Towards Reliable Compositional Behavior in QALD Systems

David Maria Schmidt[1,*], Raoul Schubert[1] and Philipp Cimiano[1]

[1]Semantic Computing Group, CITEC, Technical Faculty, Bielefeld University, Bielefeld, Germany

Abstract

Accompanying the Research Track paper "CompoST: A Benchmark for Analyzing the Ability of LLMs To Compositionally Interpret Questions in a QALD Setting", we investigate how compositionality is approached in our compositional question answering over linked data (QALD) pipeline "NeoDUDES". This way, we point out how some of the limitations of large language models (LLMs) w.r.t. compositional interpretation of QALD questions can be dealt with by combining LLMs with symbolic methods. In our demo, we show detailed intermediate results from the NeoDUDES pipeline, underlining how the strengths of neural and symbolic approaches can be combined in a fine-grained, compositional pipeline to tackle compositional tasks in a more reliable fashion.

Keywords

Compositionality, Question Answering over Linked Data, Large Language Models, Semantic Web

1. Introduction

The reasoning abilities of large language models (LLMs) and especially the abilities of LLMs to work and reason in a compositional way have been investigated by numerous related works in recent years, either by directly targeting compositionality [1, 2, 3], or indirectly through various multi-step reasoning tasks [4, 5, 6, 7]. Other works also investigated the abilities of LLMs w.r.t. compositionality from a (complexity-) theoretical perspective [8, 9, 10, 11, 12]. However, in contrast to our accompanying Research Track paper "CompoST: A Benchmark for Analyzing the Ability of LLMs To Compositionally Interpret Questions in a QALD Setting" [13], most related work does not deal with or focus on QALD specifically. Therefore, we adapt the compositionality term of Zoltán G. Szabó [14] to the QALD domain and generate a corresponding benchmark dataset *CompoST* ("Compositional Systematicity Test") to test the abilities of LLMs to systematically recombine known parts to new SPARQL queries. Our evaluation, summarizing over 400 experiments, raises substantial concerns w.r.t. the ability of LLMs to interpret QALD questions in a systematic, compositional way, even when all necessary information to interpret a question is given in the input. In line with, e.g., Dziri et al. [1], this may indicate fundamental limitations of LLMs when it comes to truly compositional tasks.

In this paper, we further analyze the issues of LLMs with compositional tasks that have been raised in [13]. Furthermore, we propose first solutions to those problems by demonstrating how we deal with compositionality in our compositional question answering over linked data (QALD) pipeline "NeoDUDES"[1] [15, 16]. This way, we show new avenues for future work to ensure reliable compositional behavior by combining the strengths of symbolic and LLM-based approaches in a single QALD pipeline. Finally, we leverage the fine-grained nature of our pipeline to present its intermediate results in a corresponding demo[2] and thus allow deep insights into the inner workings of the pipeline.

ISWC 2025 Companion Volume, November 2–6, 2025, Nara, Japan

*Corresponding author.

daschmidt@techfak.uni-bielefeld.de (D. M. Schmidt); raoul.schubert@uni-bielefeld.de (R. Schubert); cimiano@techfak.uni-bielefeld.de (P. Cimiano)

https://davidmschmidt.de/ (D. M. Schmidt); http://cimiano.de (P. Cimiano)

0000-0001-7728-2884 (D. M. Schmidt); 0009-0009-7743-5401 (R. Schubert); 0000-0002-4771-441X (P. Cimiano)

[1]NeoDUDES repository: https://github.com/ag-sc/neodudes Zenodo: https://doi.org/10.5281/zenodo.12610054

[2]Demo video: https://doi.org/10.5281/zenodo.16531345. Although the video only shows one example, the demo supports generating these illustrations for any given input question.

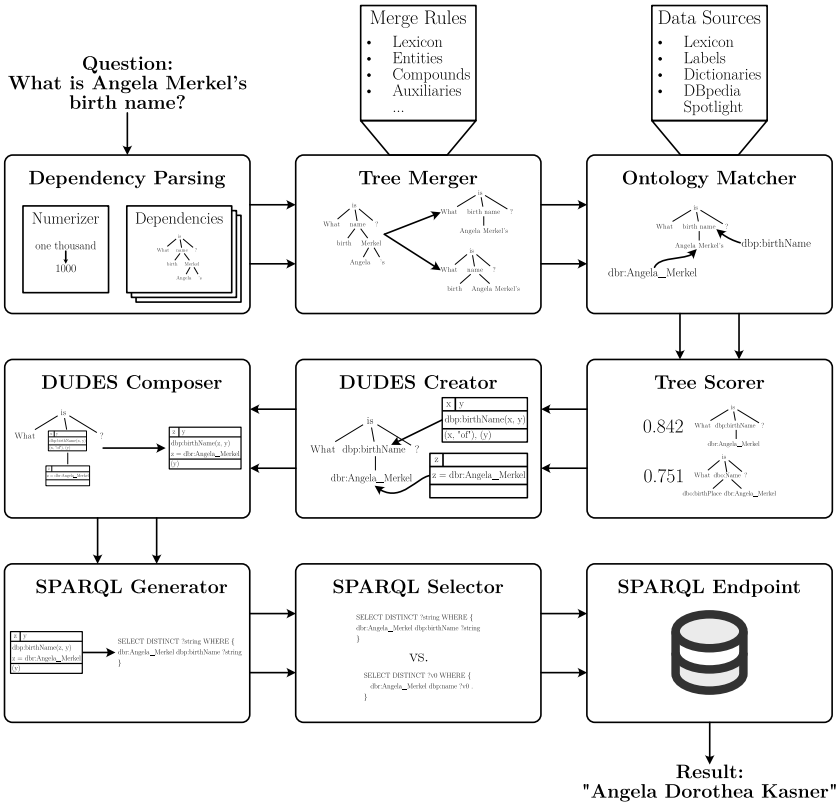


Figure 1: Schema of the compositional NeoDUDES QALD pipeline (taken over from [15]).

This paper only highlights the parts of the NeoDUDES pipeline most relevant for compositionality. For more detailed information about the pipeline, we refer the interested reader to [15, 16].

2. Methods

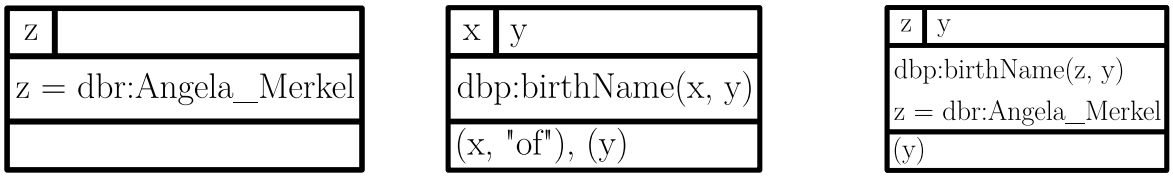
In *CompoST* [13], we focus on a sub-property of compositionality, namely *systematicity*. That means, citing a classic example from Zoltán G. Szabó [14], a compositional system understanding both "brown dog" and "black cat" should understand "brown cat" as well. Adapted to the QALD domain, we thus expect a compositional system which correctly generates SPARQL queries for "What is the birth name of Angela Merkel?" and "What is the birth place of Barack Obama?" to also generate a correct query for, e.g., "What is the birth place of Angela Merkel?". As we show in [13], this property is violated frequently by current LLMs. Thus, in this section, we focus on the question how one can approach compositionality in a different, more robust way by the example of the NeoDUDES pipeline.

An overview of the whole pipeline is given in Figure 1. In this paper, we mainly focus on the DUDES composition as well as the way ambiguities are handled by the pipeline.

The core of the NeoDUDES pipeline as well as its compositional backbone are *Dependency-based Underspecified Discourse Representation Structures (DUDES)*, which are used to represent the meaning of a question or parts of it. They are defined as follows:

Definition 1 (Dependency-based Underspecified Discourse Representation Structure [17, 15]). *A Dependency-based Underspecified Discourse Representation Structure (DUDES) is a triple* (v, D, S) *where:*

- $v \in U \cup \{\epsilon\}$ *is the main variable (also called referent marker or distinguished variable) where* ϵ *represents the absence of a main variable*
- $D = (U, C)$ *is a Discourse Representation Structure (DRS) [17, 18, 19] with*



(a) Entity DUDES for `dbr:Angela_Merkel`

(b) Property DUDES for `dbo:birthName` with selection pairs $(x, "of")$ and (y, ϵ)

(c) Composition of 2a and 2b using selection pair $(x, "of")$.

Figure 2: Illustration of exemplary DUDES and their composition (taken over from [15]).

  - *set of variables U (also called discourse universe or referent markers)*
  - *set of conditions C over variables U*
- *S is a set of selection pairs of the form (v, m) with v being a variable from U and m being a marker word for that variable with ϵ representing the empty marker, i.e., no marker being connected to that variable. Instead of writing ϵ, the second tuple component can also just be left out.*

Two example DUDES are given in Figures 2a and 2b. Thus, DUDES represent the meaning of (parts of) a question or sentence through logical formulas that roughly correspond to SPARQL triple patterns in most cases. Additionally, the main variable and selection pairs are what makes this representation compositional, as they are used for the composition operation of two DUDES. This operation has the goal to compose the meaning of two DUDES, and thus two parts of a question, into one combined meaning representation. This composition operation is defined as follows:

Definition 2 (DUDES Composition [17, 15]). *Let $d_1 = (v_1, D_1 = (U_1, C_1), S_1)$, $d_2 = (v_2, D_2 = (U_2, C_2), S_2)$ be two DUDES with disjoint variable sets, i.e. $U_1 \cap U_2 = \emptyset$. The DUDES composition operation $\odot$ for substituting d_1 into d_2 using selection pair $p = (x \in U_2, m) \in S_2$ and resulting in a composed DUDES $d_c = (v_c, D_c = (U_c, C_c), S_c)$, written $d_c = d_1 \overset{p}{\odot} d_2$, is defined as follows:*

$$U_c = U_2[x := v_1] \cup U_1 \qquad S_c = (S_2 \cup S_1) \setminus p \qquad v_c = \begin{cases} v_1 & \text{if } x = v_2 \\ v_2 & \text{else} \end{cases}$$
$$C_c = C_2[x := v_1] \cup C_1$$

An example of the result of such a composition operation is given in Figure 2c. In practice, most questions consist of more than two parts. Therefore, the composition operation is applied bottom-up along a (slightly compacted) dependency tree in order to create a single DUDES representation from multiple parts of a question. Both the original DUDES for all parts of the question as well as the final resulting DUDES are presented in our demo for the respective given question.

However, as the result of the above composition operation depends on both the used selection pair as well as the direction in which the operation is applied, there may arise ambiguities when there are multiple possibilities that cannot be further disambiguated. In those cases, we both want to avoid the problem of state space explosion as well as losing potentially correct combinations by deciding for one option too early and discarding the others.

Therefore, the NeoDUDES pipeline applies an iterative approach for most steps, assembling one query at a time without explicitly storing all possible combinations of, e.g., DUDES compositions in memory. However, even though this limits the memory consumption, the number of possible combinations remains large. In order to still get results in a reasonable time, a decision has to be made which combinations are tried first and to which extent to tradeoff memory for runtime.

In the pipeline, this is mainly done in the Tree Scorer and SPARQL Selector components. The Tree Scorer gets a set of trees to which different node merging heuristics have been applied. For example, in the original dependency tree, the entity `dbr:Angela_Merkel` is split into two nodes corresponding to “Angela” and “Merkel”, respectively. To improve those correspondences and facilitate ontology

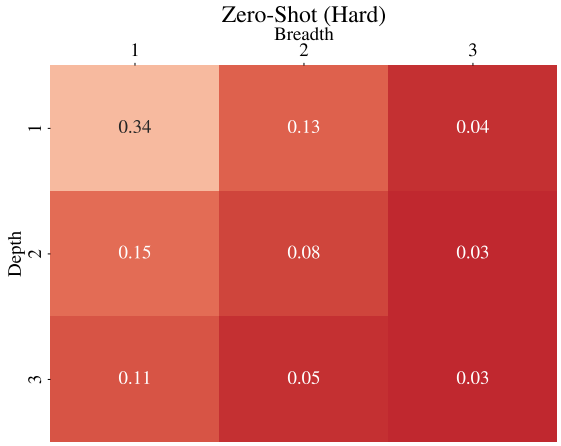


Figure 3: Macro F_1 scores of best zero-shot approach for the *hard* dataset, grouped by graph pattern depth and breadth. Prompt optimization data comprised samples up to two edges, i.e., up to breadth 2, depth 1 and depth 2, breadth 1. For few-shot and fine-tuning results see [13].

matching, various merging heuristics are applied. However, this gives us a number of candidate trees that typically cannot all be processed at the same time. Therefore, the Tree Scorer assigns each tree a score, aiming to measure how promising that tree is in terms of size and matched ontology resources and thus determining an order for those trees in which they are further processed. These scores together with the corresponding trees are as well part of the demo.

Similarly, the iterative approach which follows after the Tree Scorer produces candidate SPARQL queries one by one while avoiding to store multiple possible combinations at the same time to limit memory usage. From these queries, one has to be chosen as the final output. As there are no clear rules for what a good SPARQL query for a specific question is, we train an LLM to compare two candidate queries w.r.t. a given question and choose the "better" one. These single comparisons of the LLM-based SPARQL selection are then aggregated in different ways for different strategies to arrive at a final decision. This way, the symbolic part of the approach shows its strength by trying different possibilities in a structured and reliable way while an LLM-based approach deals with the more "fuzzy" task of selecting a final query from a set of candidates. The SPARQL selection is also illustrated in the demo.

All in all, this underlines how symbolic and neural components can work together to provide both, reliable compositional behavior without losing possible combinations of the compositional parts, as well as LLM-based optimizations and trained heuristics for scenarios where all available rules have been applied but still some decisions need to be made. This shows promising avenues for future research.

3. Results and Discussion

Revisiting the scores achieved by current LLMs in [13], this underlines the need for new methods that deal with compositionality in a more robust way. This gets especially clear when considering the scores of the best-performing zero-shot approach on the *hard* CompoST dataset, presented in Figure 3, together with the scores of the few-shot and fine-tuning approaches shown in [13]. These F_1 scores, grouped by breadth and depth of the respective SPARQL graph pattern, were achieved by Llama 3.3 [20], using MIPRO prompt optimization with the *heavy* preset in combination with Chain of Thought prompting. All experiments were conducted using the DSPy framework [21, 22]. Overall, a broad set of models has been tested, namely Llama 3.3 (70B) [20], Phi-4 (14B) [23], Qwen2.5-Coder (7B) [24, 25], OLMo 2 (7B) [26] and GPT-4o-mini [27]. The prompting techniques included plain prompting, COPRO prompt optimization as well as MIPRO prompt optimization, each tested with and without Chain of Thought prompting. Further information on the conducted experiments as well as heatmaps for few-shot and fine-tuning can be found in the accompanying Research Track paper [13].

In general, the experimental results of Schmidt et al. [13] show that LLMs struggle with compositional tasks, especially as the size of the questions gets further away from the data observed during training - although it was ensured that the training data contained all relevant information and "building blocks" to construct the answer for the questions in the validation and test splits of the dataset. Even for "self-contained" experiments, containing all necessary information to solve the task in the input, the achieved scores did not exceed 0.57 in terms of test macro F_1 scores (achieved on the *easy* CompoST dataset with Llama 3.3 using few-shot prompting together with MIPRO prompt and shot optimization with a *medium* preset). This shows additional effort is needed whenever reliable compositional interpretation of QALD questions is necessary. Some possibilities on how to achieve this have been outlined above.

However, there are also limitations of the presented NeoDUDES pipeline. First, the pipeline relies on the availability of a Lemon lexicon [28], covering all relevant verbalizations of used properties. Similarly, e.g., `rdfs:label` data for our trie-based entity matcher or some other entity matcher has to be available. Second, depending on how many combinations have to be tested before a suitable candidate is found, the runtime of the NeoDUDES pipeline can be much longer than typical inference times of current LLMs. Finally, the initial implementation effort of the pipeline was higher than the effort typically necessary for, e.g., fine-tuning or prompt optimization for the QALD task.

Nevertheless, the existing pipeline can now be easily adapted to new datasets or knowledge graphs. This can be especially useful for small or domain-specific datasets which are not sufficient for purely LLM-based approaches either due to their size or because the respective knowledge graph or the style of the questions deviates too much from the LLM training data. Moreover, an open modular pipeline like the NeoDUDES approach provides a whole new level in terms of explainability and possibilities to justify answers or fix errors that a purely LLM-based approach typically cannot offer.

In future work, we aim to test different ways to generate the required Lemon lexicon automatically, using combinations of existing data sources (e.g., WordNet, Wikidata alias entries, inflection tools, etc.) as well as LLM-based generation. Additionally, as the goal of the pipeline is to use symbolic and neural approaches where they each work best, we plan to replace different parts of the pipeline with LLMs for that specific sub-task and investigate how this compares to the performance of the symbolic pipeline in terms of compositionality. Although preliminary results show promising performance of the NeoDUDES pipeline on the CompoST dataset, we aim to provide a full evaluation in the future. Finally, various performance optimizations and further parallelization is planned to improve the responsiveness and runtime of the pipeline.

4. Conclusion

To summarize, in this paper, we revisited the results of the accompanying Research Track paper [13], highlighting the weaknesses and limitations of LLMs when it comes to truly compositional tasks. Motivated by these findings, we investigated how the NeoDUDES pipeline, a compositional approach by design that combines the strengths of both symbolic and neural methods in a transparent modular pipeline, approaches compositionality. An illustration of these aspects and advantages is also part of the corresponding demo[3].

Acknowledgments

This work is partially funded by the Ministry of Culture and Science of the State of North Rhine-Westphalia under grant no NW21-059A (SAIL).

Declaration on Generative AI

The authors have not employed any Generative AI tools.

[3]Demo video: https://doi.org/10.5281/zenodo.16531345. Although the video only shows one example, the demo supports generating these illustrations for any given input question.

References

[1] N. Dziri, X. Lu, M. Sclar, X. L. Li, L. Jiang, B. Y. Lin, P. West, C. Bhagavatula, R. Le Bras, J. D. Hwang, S. Sanyal, S. Welleck, X. Ren, A. Ettinger, Z. Harchaoui, Y. Choi, Faith and fate: limits of transformers on compositionality, in: Proceedings of the 37th international conference on neural information processing systems, Nips '23, Curran Associates Inc., Red Hook, NY, USA, 2023. Number of pages: 40 Place: New Orleans, LA, USA tex.articleno: 3081.

[2] O. Press, M. Zhang, S. Min, L. Schmidt, N. Smith, M. Lewis, Measuring and Narrowing the Compositionality Gap in Language Models, in: H. Bouamor, J. Pino, K. Bali (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2023, Association for Computational Linguistics, Singapore, 2023, pp. 5687–5711. URL: https://aclanthology.org/2023.findings-emnlp.378/. doi:10.18653/v1/2023.findings-emnlp.378.

[3] D. Hupkes, V. Dankers, M. Mul, E. Bruni, Compositionality Decomposed: How do Neural Networks Generalise?, Journal of Artificial Intelligence Research 67 (2020) 757–795. URL: https://jair.org/index.php/jair/article/view/11674. doi:10.1613/jair.1.11674.

[4] Y. Zhang, A. Backurs, S. Bubeck, R. Eldan, S. Gunasekar, T. Wagner, Unveiling Transformers with LEGO: a synthetic reasoning task, 2023. URL: http://arxiv.org/abs/2206.04301. doi:10.48550/arXiv.2206.04301, arXiv:2206.04301 [cs].

[5] M. Nye, A. J. Andreassen, G. Gur-Ari, H. Michalewski, J. Austin, D. Bieber, D. Dohan, A. Lewkowycz, M. Bosma, D. Luan, C. Sutton, A. Odena, Show Your Work: Scratchpads for Intermediate Computation with Language Models, 2021. URL: http://arxiv.org/abs/2112.00114. doi:10.48550/arXiv.2112.00114, arXiv:2112.00114 [cs].

[6] S. Welleck, J. Liu, X. Lu, H. Hajishirzi, Y. Choi, NaturalProver: Grounded Mathematical Proof Generation with Language Models, Advances in Neural Information Processing Systems 35 (2022) 4913–4927. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/1fc548a8243ad06616eee731e0572927-Abstract-Conference.html.

[7] A. Saparov, H. He, Language Models Are Greedy Reasoners: A Systematic Formal Analysis of Chain-of-Thought, 2023. URL: http://arxiv.org/abs/2210.01240. doi:10.48550/arXiv.2210.01240, arXiv:2210.01240 [cs].

[8] W. Merrill, A. Sabharwal, The parallelism tradeoff: Limitations of log-precision transformers, Transactions of the Association for Computational Linguistics 11 (2023) 531–545. URL: https://doi.org/10.1162/tacl_a_00562. doi:10.1162/tacl_a_00562, tex.eprint: https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00562/2131191/tacl_a_00562.pdf.

[9] L. Chen, B. Peng, H. Wu, Theoretical limitations of multi-layer Transformer, 2024. URL: https://arxiv.org/abs/2412.02975, arXiv: 2412.02975 [cs.LG].

[10] N. Zubić, F. Soldá, A. Sulser, D. Scaramuzza, Limits of deep learning: Sequence modeling through the lens of complexity theory, 2025. URL: https://arxiv.org/abs/2405.16674, arXiv: 2405.16674 [cs.LG].

[11] G. Feng, B. Zhang, Y. Gu, H. Ye, D. He, L. Wang, Towards revealing the mystery behind chain of thought: a theoretical perspective, in: A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, S. Levine (Eds.), Advances in neural information processing systems, volume 36, Curran Associates, Inc., 2023, pp. 70757–70798. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/dfc310e81992d2e4cedc09ac47eff13e-Paper-Conference.pdf.

[12] L. Strobl, W. Merrill, G. Weiss, D. Chiang, D. Angluin, What formal languages can transformers express? A survey, Transactions of the Association for Computational Linguistics 12 (2024) 543–561. URL: http://dx.doi.org/10.1162/tacl_a_00663. doi:10.1162/tacl_a_00663, publisher: MIT Press.

[13] D. M. Schmidt, R. Schubert, P. Cimiano, Compost: A benchmark for analyzing the ability of llms to compositionally interpret questions in a qald setting, in: The Semantic Web – ISWC 2025, Springer Nature Switzerland, Cham, 2025. doi:10.48550/arXiv.2507.21257, (in press).

[14] Z. G. Szabó, The case for compositionality, in: M. Werning, W. Hinzen, E. Machery (Eds.), The oxford handbook of compositionality, Oxford University Press, 2012.

[15] D. M. Schmidt, M. F. Elahi, P. Cimiano, Lexicalization is all you need: Examining the impact of

lexical knowledge in a compositional QALD system, in: M. Alam, M. Rospocher, M. van Erp, L. Hollink, G. A. Gesese (Eds.), Knowledge engineering and knowledge management, Springer Nature Switzerland, Cham, 2025, pp. 102–122.

[16] D. M. Schmidt, M. F. Elahi, P. Cimiano, Lexicalization Is All You Need: Examining the Impact of Lexical Knowledge in a Compositional QALD System, in: C. Badenes-Olmedo, I. Novalija, E. Daga, L. Stork, R. G. Pillai, L. Dierickx, B. Kruit, V. Degeler, J. Moreira, B. Zhang, R. Alharbi, Y. He, A. Graciotti, A. M. Tirado, V. Presutti, E. Motta (Eds.), Joint Proceedings of Posters, Demos, Workshops, and Tutorials of the 24th International Conference on Knowledge Engineering and Knowledge Management (EKAW-PDWT 2024), volume 3967 of *CEUR Workshop Proceedings*, CEUR, Amsterdam, Netherlands, 2024.

[17] P. Cimiano, C. Unger, J. P. McCrae, Ontology-Based Interpretation of Natural Language, Synthesis Lectures on Human Language Technologies, Morgan & Claypool Publishers, 2014.

[18] K. Hans, A theory of truth and semantic representation, Formal Methods in the Study of language (1981).

[19] H. Kamp, U. Reyle, From discourse to logic: Introduction to modeltheoretic semantics of natural language, formal logic and discourse representation theory, volume 42, Springer Science & Business Media, 2013.

[20] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, et al., The Llama 3 Herd of Models, 2024. URL: http://arxiv.org/abs/2407.21783. doi:10.48550/arXiv.2407.21783, arXiv:2407.21783 [cs].

[21] O. Khattab, A. Singhvi, P. Maheshwari, Z. Zhang, K. Santhanam, S. Vardhamanan, S. Haq, A. Sharma, T. T. Joshi, H. Moazam, H. Miller, M. Zaharia, C. Potts, DSPy: Compiling declarative language model calls into self-improving pipelines, The Twelfth International Conference on Learning Representations, 2024.

[22] O. Khattab, K. Santhanam, X. L. Li, D. Hall, P. Liang, C. Potts, M. Zaharia, Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive NLP, arXiv preprint arXiv:2212.14024 (2022).

[23] M. Abdin, J. Aneja, H. Behl, S. Bubeck, R. Eldan, S. Gunasekar, M. Harrison, R. J. Hewett, M. Javaheripi, P. Kauffmann, J. R. Lee, Y. T. Lee, Y. Li, W. Liu, C. C. T. Mendes, A. Nguyen, E. Price, G. d. Rosa, O. Saarikivi, A. Salim, S. Shah, X. Wang, R. Ward, Y. Wu, D. Yu, C. Zhang, Y. Zhang, Phi-4 Technical Report, 2024. URL: http://arxiv.org/abs/2412.08905. doi:10.48550/arXiv.2412.08905, arXiv:2412.08905 [cs].

[24] A. Yang, B. Yang, B. Hui, B. Zheng, B. Yu, C. Zhou, C. Li, C. Li, D. Liu, F. Huang, others, Qwen2 technical report, arXiv preprint arXiv:2407.10671 (2024).

[25] B. Hui, J. Yang, Z. Cui, J. Yang, D. Liu, L. Zhang, T. Liu, J. Zhang, B. Yu, K. Dang, others, Qwen2. 5-coder technical report, arXiv preprint arXiv:2409.12186 (2024).

[26] T. OLMo, P. Walsh, L. Soldaini, D. Groeneveld, K. Lo, S. Arora, A. Bhagia, Y. Gu, S. Huang, M. Jordan, N. Lambert, D. Schwenk, O. Tafjord, T. Anderson, D. Atkinson, F. Brahman, C. Clark, P. Dasigi, N. Dziri, M. Guerquin, H. Ivison, P. W. Koh, J. Liu, S. Malik, W. Merrill, L. J. V. Miranda, J. Morrison, T. Murray, C. Nam, V. Pyatkin, A. Rangapur, M. Schmitz, S. Skjonsberg, D. Wadden, C. Wilhelm, M. Wilson, L. Zettlemoyer, A. Farhadi, N. A. Smith, H. Hajishirzi, 2 OLMo 2 Furious, 2025. URL: http://arxiv.org/abs/2501.00656. doi:10.48550/arXiv.2501.00656, arXiv:2501.00656 [cs].

[27] OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al., GPT-4 Technical Report, 2024. URL: http://arxiv.org/abs/2303.08774. doi:10.48550/arXiv.2303.08774, arXiv:2303.08774 [cs].

[28] J. P. McCrae, D. Spohr, P. Cimiano, Linking lexical resources and ontologies on the semantic web with lemon, in: Proceedings of the 8th extended semantic web conference on The semantic web: research and applications (ESWC), volume 6643, 2011, pp. 245–259.