

Use of graph-based knowledge organization to improve the results of Retrieval Augmented Generation for narrative texts

Michał Podstawski^{1,2,3}[0000–0003–1222–6894] and Mariusz Kamola^{2,1}[0000–0003–1550–5901]

¹ NASK National Research Institute, Warsaw, Poland
<https://nask.pl/>

² Warsaw University of Technology, Poland
{michal.podstawski.stud,mariusz.kamola}@pw.edu.pl
<https://pw.edu.pl/>

³ TCL Research Europe, Warsaw, Poland
<https://tcl-research.pl/>

Abstract. Existing Retrieval Augmented Generation solutions struggle when they deal with narrative texts. This kind of texts comes with additional levels of complexity, starting from non-linear narration, through diversity of characters, to hidden meanings carried by form and language. This paper proposes approach to overcome these obstacles by incorporating text aspects described in theory of literature into current state-of-the-art solutions. The improvement is achieved by utilizing meticulously designed graph-based knowledge layout, where each important aspect of narrative text is appropriately reflected. The layout consists of multiple layers, which empowers ability to capture inner meanings of the text on different levels of abstraction and in different aspects. This article proposes complete technical solution that enables efficient work with narrative texts. Additionally, it provides a study on the most effective configuration of the framework and the importance of each aspect in the source texts. Experiments have shown that for narrative texts our approach significantly improves current state-of-the-art results.

Keywords: RAG · narrative texts · graph-based knowledge organization

1 Introduction

Large Language Models (LLMs) have revolutionized nowadays IT. Their ability to interact with a human in natural language changed our way of thinking about human-machine collaboration, and transformed approach to build modern software. The capabilities of LLMs are even broader thanks to Retrieval Augmented Generation (RAG) framework. By using RAG, model can be provided with additional context, what enables it to use information absent in the original training dataset.

Therefore, the area of large language models is one of the fastest growing areas of the computer sciences. Almost every day comes with new ideas and solutions. Enhanced with RAG, large language models can operate on very different sources, from dispersed notes to technical documents. Recent research on RAG, including work on incorporating graphs of knowledge in the process, enabled even broader selection of sources and texts to be effectively handled.

Despite this development, there is lack of approaches that could work specifically with literary, narrative texts. This kind of texts comes with very specific set of challenges that need to be precisely addressed. The aim of this work is to reuse and improve some state-of-the-art tools and techniques, then adapt them to work with narrative texts.

The main contributions of the paper include:

- definition of main characteristics of narrative texts in the context of Retrieval Augmented Generation;
- introduction of graph-oriented data organization optimized towards narrative texts, and implementation of the tools that enable construction and use of proposed graph;
- exploration and systematization of prompt engineering capabilities to select the best prompts for summarizations used in graph construction;
- comparison of impact of input parameters on retrieval quality and cost-effectiveness;
- adaptation of existing datasets (QuALITY [21], NarrativeQA [8]) for question answering without predefined answer options and implementation of the benchmarks.

2 Related Work

The enhancements for Retrieval Augmented Generation are currently under active research. The framework firstly has been introduced by P. Lewis et. al [11]. It quickly developed from this naive RAG, through several phases of improvements, into more advanced RAG systems, and finally into many mature and sophisticated approaches called modular RAG [6]. The current efforts focus on adjusting RAG to specific needs and use cases, including dynamically tailoring retrieval process [27], mixing specialized and general models in retrieval process [30], hybrid retrieval approaches [33], enabling parallel paths of searching for information [16] or including self-correctness mechanisms [12]. There are plenty of works studying both best practices in RAG [28] and methods of testing and benchmarking it [26, 9].

Utilizing graphs for RAG is also a fast growing area [22]. Usually, it is based on graph of knowledge that contains entities described in a document or multiple documents. Also, different documents can be independent entities themselves [29]. In most cases the graph is built using large language model and detailed prompt engineering techniques, that lead to recognition of entities and relations by model itself. The applicability of this kind of solutions varies from medicine [31], through education [2], to road traffic [7] and customer services [32]. The

recent advancement in this area is Graph RAG [5]. It proposes a multi-step approach, where firstly, a graph of knowledge containing entities from the text is built, and later identified entities are clustered and post-processed (which includes graph learning and summarization techniques) to enable better understanding of the structure of the text.

The most recent work related to our study is RAPTOR [25]. It introduced the idea of multilayered tree of summarizations for RAG that captures text meaning on different levels of generalization. It differs from the previous solutions as it is not based on graph of knowledge of any kind. Our research builds on the RAPTOR components, extending their capabilities and adapting specifically towards narrative texts. In current study we reuse clustering algorithm described by RAPTOR. Also, RAPTOR proved that it is more efficient to use retrieval method called *collapsed tree*, meaning search for retrieval equally amongst all the nodes within the constructed tree (graph), instead of the tree traversals (e.g. from the most general information to more specific).

3 Proposed solution

The characteristics of narrative texts have the fundamental meaning for this study. Narrative text can be defined as a kind of text that tells a story about events or situations, usually, but not obligatorily, in chronological order. It is focused on telling the story that may be fictitious or based on real events. It includes characters, theme, setting and objects, as well as relations between them. It is told by a narrator that can be a part of the presented world or an independent observer. The narrator can have and present his/her own opinions on the presented events [3, 14, 17].

Narrative texts are especially difficult for large language models. Firstly, they are burdened with all of the difficulties related to long texts, including the fact that typically they are longer than the allowed model context. Furthermore, even if the model can digest the whole text, it significantly suffers from lost-in-the-middle problem — a phenomenon where for a long text, the model tends to overlook details described in the middle of text [13].

Secondly, there are many difficulties related specifically to narrative texts. They include intertwining threads, intended inconsistencies in narration, time jumps, foreshadowing, analepses, prolepses and discontinuities that leave space for guesswork for the reader.

The next class of problems is brought by the characters described in texts. They can have their own perspectives and ways of thinking; sometimes the very narrator is one of the characters that interact with the others, which additionally increases the difficulty of understanding the context. Also, narration can be carried out from point of view of many characters. Moreover, often characters can change their way of thinking and behavior during the story, which influences the storytelling.

Similar problems are entailed with setting of the story. It can highly impact characters and their decisions. They can be situated in real historical conditions

or in imaginary world. In some cases both approaches overlap, and fictional universe is situated inside real historical circumstances.

Another important factor is the language and style of the story. It is especially related to narrative texts, where language often plays a crucial role. It includes different types of the figures of speech, like metaphors, hyperbolas, personifications, as well as the use of symbols, allegory and irony. In many cases the used language allows us to understand the hidden meaning of the narration.

Accordingly, Robert DiYanni [3] defines following constitutive aspects of narrative texts :

- Plot and Structure
- Character
- Setting
- Point of View
- Language and Style
- Theme
- Irony and Symbol

Each of those characterizes different way of looking at and thinking about text. Identified aspects make it possible to build a graph that accounts for all of them, with increasing levels of generalization. The general structure of the designed graph is presented in Figure 1.

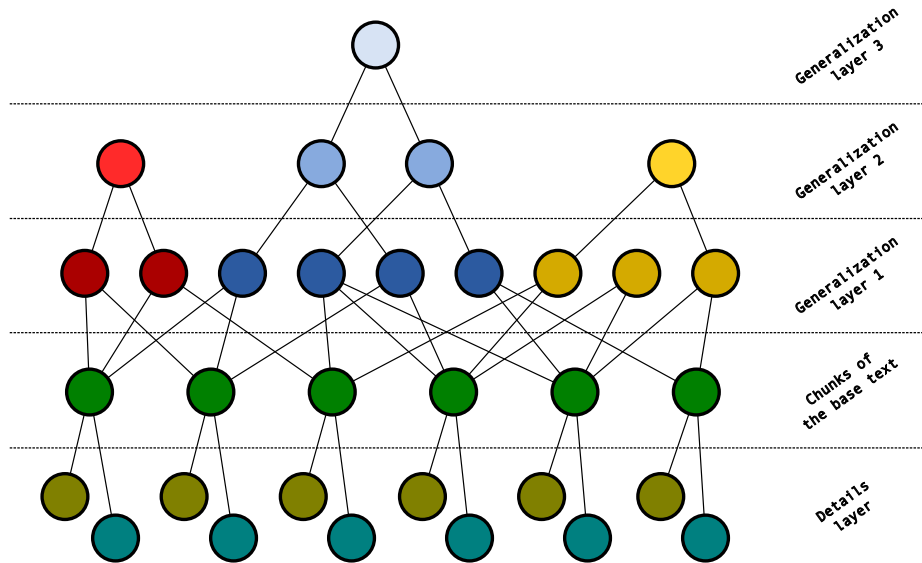


Fig. 1. The general structure of the proposed graph (different colors denote different aspects).

The graph is built in stages. The first step follows naive RAG approach, the whole text is cut into small parts called chunks, which become starting nodes for building a graph. For each chunk, an embedding is calculated using selected embedding model. The optimal chunk size for solution was a matter of experiments, the detailed analysis is provided in the evaluation chapter. Next, the created chunks are grouped into clusters. The clusters gather chunks that are semantically similar and together capture common aspect or idea. For creating clusters we utilized algorithm proposed by RAPTOR adapted for narrative texts purposes. It allows to dynamically adjust clusters size and total number of needed clusters. In order to achieve that, RAPTOR’s algorithm uses Gaussian Mixture Models (GMM) with Uniform Manifold Approximation and Projection (UMAP) for dimension reduction [15], and Bayesian Information Criterion (BIC) for determining total number of clusters [25]. In this study, we let selected large language model to assess which clusters are related to which aspects. It is done by prompting LLM to select the best matching aspects for each of previously built clusters. In rare cases (less than 5%), when it is not possible to find clusters suitable for some of the aspects, the first clustering stage is repeated.

When this stage is finished, the model is prompted to prepare summarizations for each cluster. These summarizations are aspect oriented, meaning they always request the model to focus on given aspect of the text. We experimented with different prompt engineering techniques, including few-shot prompting [1] and in-context learning [4]. The best overall results are achieved using prompts with detailed description on how we understand specific aspect, and with low generation temperature. However, overly long and detailed prompts with many examples for in-context learning generally cause deterioration of summarizations quality. Increasing the answer temperature also negatively impacts prompt following and leads to regression.

When summarizations are ready, they are added to graph as nodes of first level of generalization. Also, embeddings get calculated in this step. Next, separately for each aspect, there is a subsequent phase of clustering. It is done on newly added nodes with summarizations treated as base chunks. As captured aspects are already set up, in the consecutive phases we do not perform additional selection of matching aspect, but we reuse previous summarization strategy. The process is repeated iteratively, until it is not possible (there are not enough nodes) to create clusters for next level of generalization.

Experiments showed that also adding special nodes that extract details from chunks improved overall quality. In this case, model is prompted to get the most important details from given chunk, and the result is treated as content for additional node. While it is possible to add virtually unlimited number of such nodes, experiments revealed that adding more than two does not have further positive impact. It is additionally analyzed in experiments section.

When the process is finished, the graph is ready to be used in RAG. The retrieval process goes the same way as in naive RAG approach. There is embedding vector calculated for question, and a chosen number of nodes with the highest cosine similarity of embedding are retrieved. We conducted experiments

in order to check efficiency of utilizing graph traversals. Neither enforcing start from the most specific, nor the most general node, before further traversal across the connected layers, positively impacted the overall quality.

As the retrieval process comes with no change in comparison to vanilla RAG, it potentially can be extended by some of the techniques for retrieval improvement, such as reranking [11] or HyDE [23], in order to further increase the correctness of answers. It could be a topic for further studies.

The solution is by design model-independent and should work with any large language model. For the conducted experiments, we used two models from the GPT family. GPT models are transformer-based architectures that use self-attention and autoregressive decoding to generate coherent and context-aware text [34]. Specifically, we used GPT-3.5 Turbo for graph building and summarizations due to its efficiency and cost-effectiveness in handling structured data representations and content condensation. For tasks requiring deeper contextual understanding and nuanced classification, such as determining the character of a cluster and generating final answers, we employed GPT-4 [20]. A comparison of the models used is presented in Table 1. For embeddings, we used the *text-embedding-ada-002* model from OpenAI, which provides 1536-dimensional embeddings optimized for various tasks [19].

Table 1. Comparison of Used Models

Model	Context Length	Training Data	Parameters
<i>GPT-3.5 Turbo</i>	16,385 tokens	up to September 2021	20B (estimated)
<i>GPT-4</i>	8,192 tokens	up to December 2023	1.76T (estimated)

4 Evaluation

4.1 Tools and datasets

As baselines, we use RAPTOR and naive RAG. Naive RAG has been implemented using LangChain toolkit [10]. In experiments for all solutions we retrieve top 5 best matching nodes (chunks) to inject to context. We confirmed that it guarantees the best quality across all datasets and baselines.

As datasets we choose two sets that contain narrative texts:

- **QUALITY** [21]. It is a multiple choice question-answering dataset that consists of selection of long texts and answers prepared by human respondents. Questions have been designed to be not answerable using only small parts of the text. They need understanding of broader context scattered throughout the text in order to be properly solved.

Additionally, there is set of questions that are marked as difficult ones. These are questions that were found especially difficult by humans when time for

answer was limited. In current work we use them as a separate set. The first of the experimental sets contains all the questions, the second one only the difficult ones.

- **NarrativeQA** [8]. This dataset is prepared to assess the ability to understand texts by language models. It is build using publicly available books and movie scripts. Also, this one is designed to evaluate the ability of understanding text as a whole. Therefore, to answer most of the questions there is strong necessity of understanding the context, including theme, settings and relations between characters and objects.

Both datasets have been reworked to match the criteria of proposed solution. From both of them, we selected the examples that can be qualified as narrative texts. It has been done in the first step by selecting content that comes from Project Gutenberg [24], and in the second step by manual verification and adjustments. For QuALITY, the final extracted subset consisted of 259 documents, with 4923 total questions, among them 2606 hard questions. For NarrativeQA, we used 706 documents and total of 21894 questions. QuALITY, by default, expects selection of a matching answer from given options. It has been changed for our purposes. We treat the correct answer as ground truth, but let the model generate the full answer without guiding with predefined options.

For building benchmarks, we used RAGAS framework [26]. It is production-grade testing tool that allows testing and benchmarking wide range of LLM-based solutions. It introduces two metrics suitable for analyzed case:

- **Answer Semantic Similarity**. This metric measures how closely the model-generated answer aligns with the ground truth in meaning. This is done by converting both the generated answer and the ground truth into vector embeddings and computing their cosine similarity, which ranges from 0 to 1. A score of 1 indicates identical semantic meaning, while a score closer to 0 suggests little to no similarity. In our case, we used *text-embedding-ada-002* model to generate embeddings. This approach ensures that even if the wording differs, semantically similar answers receive high scores.
- **Answer Correctness**. This metric evaluates both the factual correctness and semantic alignment of a generated answer with the ground truth. It considers whether the response contains correct factual information while maintaining semantic similarity. The factual correctness score is computed using the F1 Score:

$$F1Score = \frac{|TP|}{|TP| + 0.5 \times (|FP| + |FN|)}$$

where TP , FP , and FN represent true positives, false positives, and false negatives in factual claims. In our case, GPT-4 was used to verify factual correctness. The final Answer Correctness score is computed as a weighted average of the acquired F1 Score (75%) and the Answer Semantic Similarity score (25%). This ensures that generated responses are both factually reliable and semantically relevant.

We present overall results for both of the metrics. For the detailed experiments we use mainly results for Answer Correctness, as it is more sensitive and provides better judgment over the real answers quality.

4.2 Results

The main experiment was conducted to quantify the general answer quality. It has been done using described metrics on datasets. The results are plotted in Figure 2.

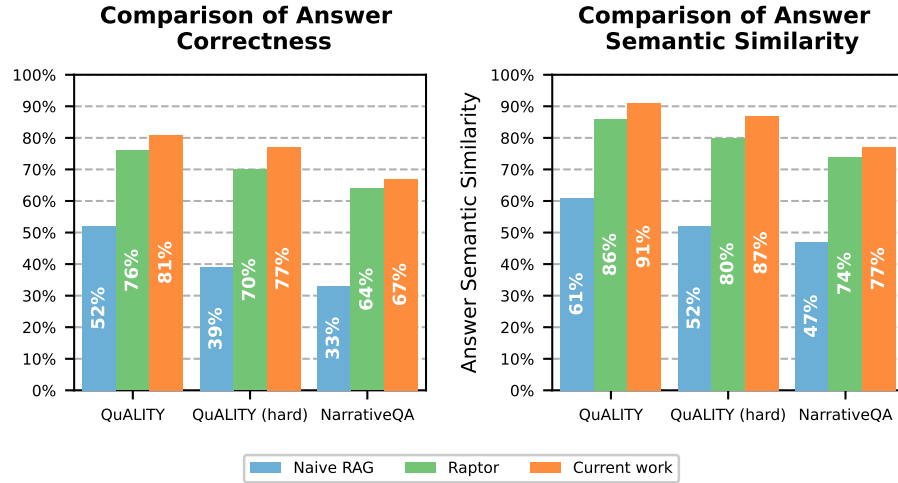


Fig. 2. Comparison of Answer Correctness and Answer Semantic Similarity across systems and datasets.

In all analyzed cases, proposed solution performs better than baselines. For Answer Correctness, the current work provides gain of 29-38% in comparison to naive RAG, and 3-7% in comparison to RAPTOR. For Answer Semantic Similarity, respective numbers are 30-35% for naive RAG, and 2-4% for RAPTOR. It is worth mentioning that as at the time of writing RAPTOR is state-of-the-art solution on QuALITY leaderboard [18].

Next, we have analyzed impact of a single aspect on overall answer quality. It is verified by gathering results both for solution with only one aspect, and for all aspects excluding selected one. It is worth noting that even with one aspect results can reach even more than 50% of correctness. It is caused by presence of base text chunks, which in some cases is enough to generate adequate answers. The combined results are plotted on Figure 3.

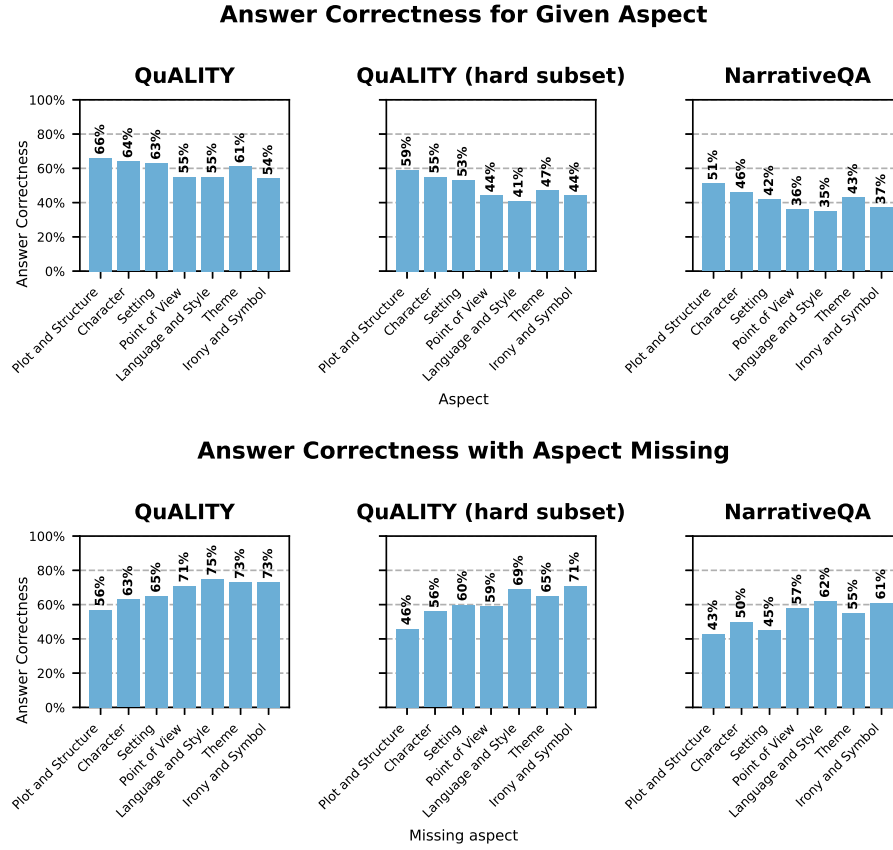


Fig. 3. Measurement of impact of one given aspect or its absence on the overall Answer Correctness.

For all the datasets, there are similar observations. The most important aspects turn out to be: Plot and Structure, Character, Setting and Theme. Aspects with the lowest impact on Answer Correctness are: Language and Style, and Irony and Symbol. However, even that last aspect has visible impact on the quality, so none of them should be discarded in order to achieve the best results.

Separate analysis was conducted to check impact of number of nodes with extracted details on the Answer Correctness. It was done by building graphs with number of such nodes ranging from 0 to 5. The results are presented in Figure 4.

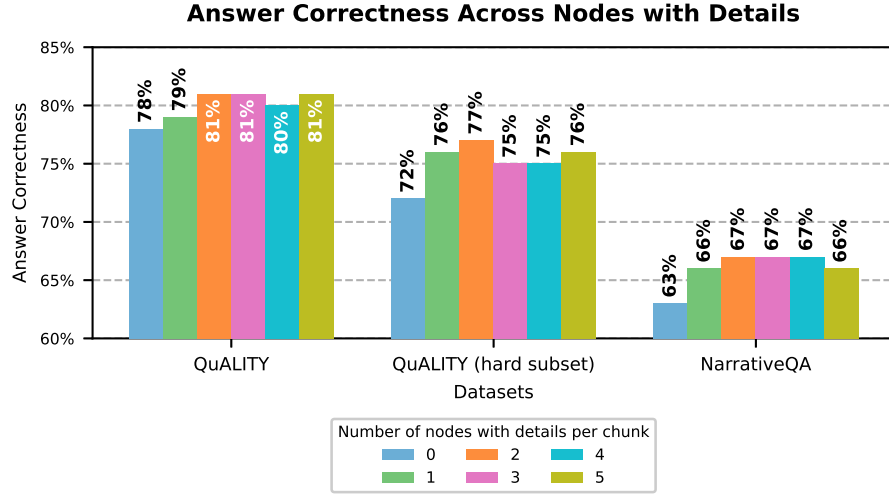


Fig. 4. Impact of number of generated nodes with details per chunk on the Answer Correctness.

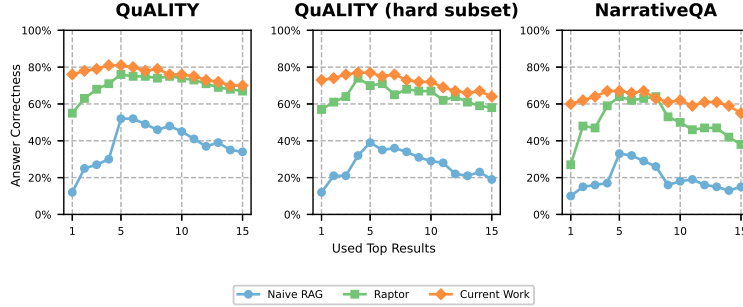
It can be observed that there is high positive impact of the first added node (improvement by 1% to 4% on Answer Correctness) and the second node (in this case the improvement is lower, from 1% to 2%). Adding further nodes does not improve correctness; in some cases it can even lead to regression.

The improvement after adding a first node can be explained by a more synthetic form of the text it provides. Introduced solution makes all the details better isolated and easier to use by the model during answer generation. The improvement that is added by a second node is the result of the freedom of the prompted model to generate content. Thanks to this, two nodes can differ by structure, method of selecting important details and ways of their presentation.

Adding a third and further nodes does not amplify this effect. Observations suggest that further nodes become too similar, and in new nodes there are not enough unique details to be cognitively valuable. Regression can also appear in the phase of retrieval. It can happen that mainly nodes from the detailed layer are selected, which limits nodes of generalization layers in the results returned to context. It affects negatively the Answer Correctness of the model.

Also, we checked how the number of graph nodes added to context impacts the Answer Correctness. It is strictly related to the overall number of tokens of a context. Results are shown in Figure 5.

Comparison of Answer Correctness For Given Top Results Number



Comparison of Answer Correctness Across Tokens Count

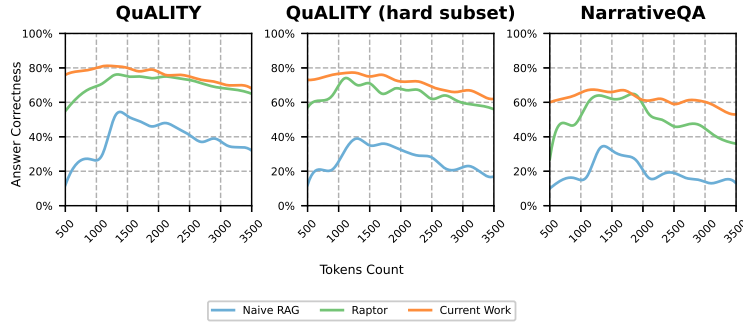


Fig. 5. Impact of number of used top matching nodes and tokens on the Answer Correctness.

The best results for all solutions were achieved while using between 3 and 6 top matching nodes, which translates to circa 1000 - 1500 tokens (context + final prompt). Further addition of nodes decreases the correctness of answer and leads to regression. This behavior can be explained by lost-in-the middle phenomenon [13]. When the number of nodes (tokens) is too high, the model is overloaded with information and loses the essence of the answer, which could have been better captured in smaller portions of text.

It is worth highlighting that proposed solution significantly outperforms baselines when using a lower number of top matching nodes, providing good results even in cases when 1 or 2 top nodes are used. In the best cases it produces results that are 21% and 64% better than baselines (in comparison to RAPTOR and to naive RAG respectively). Moreover, in current work even their results acquired with a single top node (circa 500 tokens) are of satisfying quality, ranging from 60% for NarrativeQA to 76% for QuALITY. It can be explained by multiaspectuality of the proposed approach, as there is a high chance that the gist of the answer is already captured by one of the nodes. This feature makes our research especially useful when cost-effective solution is expected, as it can

produce equally good results as baselines with three times less tokens. It is also well suited for time-sensitive solutions, as much less data needs to be retrieved.

Finally, we analyzed the impact of length of chunks and summarizations in the graph on the Answer Correctness. The results are plotted in Figure 6.

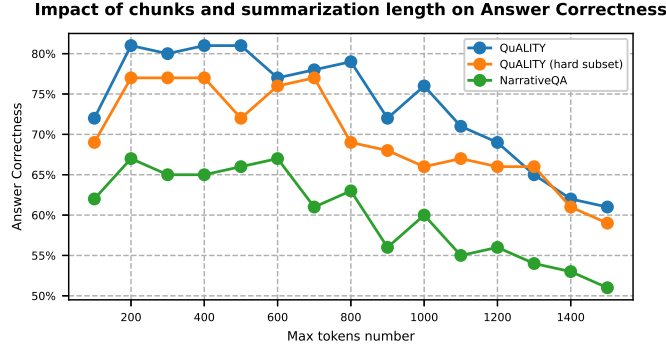


Fig. 6. Impact of length of chunks and summarization on Answer Correctness.

The best results were achieved for text of length between 200 and 600 tokens. With longer texts, the correctness is gradually decreasing. This process also can be explained by the lost-in-the-middle phenomenon. In case of excessive contexts, crucial details get diluted in the long text and are more difficult to understand and use by the model. Overall, the best Answer Correctness is obtained with max token number set to 200 (ranging from 67% for Narrative QA, through 77% for QuALITY (hard), to 81% for QuALITY). Consequently, we applied this value to other experiments.

5 Conclusions

In the paper we present a novel approach to building RAG solutions for narrative texts. It utilizes identified aspects of narrative text in order to build a graph that captures a plethora of internal meanings. Furthermore, the graph is extended with nodes that carry details from the base text, which additionally improves the overall quality of the answers. Our experiments have proven that it outperforms the current state-of-the-art solution on well-adopted datasets (e.g. QuALITY) for the task of answering questions about narrative texts. Therefore, our solution can be treated as a starting point for further exploration of this topic. The most interesting and promising future work directions include research on heuristics of retrieval from the graph and applicability of use of dedicated models that are specialized for tasks related to graph building.

Acknowledgments. This paper is based on a master’s thesis defended in March 2025 at Warsaw University of Technology.

References

1. Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language Models are Few-Shot Learners. In: *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, pp. 1877–1901. Curran Associates, Inc., Red Hook (2020). <https://arxiv.org/pdf/2005.14165>
2. Bui, T., Tran, O., Nguyen, P., Ho, B., Nguyen, L., Bui, T., Quan, T.: Cross-Data Knowledge Graph Construction for LLM-enabled Educational Question-Answering System: A Case Study at HCMUT. In: *Proceedings of AIQAM 2024*, pp. 36–43. ACM, New York (2024). <https://arxiv.org/abs/2404.09296>
3. DiYanni, R.: *Literature: Reading Fiction, Poetry, Drama, and the Essay*. McGraw-Hill, New York (1990). <https://books.google.com/books?id=fXOgtmDNuD4C>
4. Dong, Q., Dai, D., Zheng, C., Ma, J., Li, R., Xia, H., Xu, J., Wu, Z., Chang, B., Sun, X., Li, L., Sui, Z.: A Survey on In-context Learning. *arXiv preprint arXiv:2301.00234* (2022). <https://arxiv.org/abs/2301.00234>
5. Edge, D., Trinh, H., Cheng, N., Bradley, J., Chao, A., Mody, A., Truitt, S., Larson, J.: From Local to Global: A Graph RAG Approach to Query-Focused Summarization. *arXiv preprint arXiv:2404.16130* (2024). <https://arxiv.org/abs/2404.16130>
6. Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Guo, Q., Wang, M., Wang, H.: Retrieval-Augmented Generation for Large Language Models: A Survey. *arXiv preprint arXiv:2312.10997* (2023). <https://arxiv.org/abs/2312.10997>
7. Hussien, M.M., Melo, A.N., Ballardini, A.L., Maldonado, C.S., Izquierdo, R., Sotelo, M.A.: RAG-based Explainable Prediction of Road Users Behaviors for Automated Driving using Knowledge Graphs and Large Language Models. *arXiv preprint arXiv:2405.00449* (2024). <https://arxiv.org/abs/2405.00449>
8. Kočiský, T., Schwarz, J., Blunsom, P., Dyer, C., Hermann, K.M., Melis, G., Grefenstette, E.: The NarrativeQA Reading Comprehension Challenge. *Transactions of the Association for Computational Linguistics* **6**, 317–328 (2018). <https://aclanthology.org/Q18-1023>, https://doi.org/10.1162/tacl_a_00023
9. Krishna, S., Krishna, K., Mohanane, A., Schwarcz, S., Stambler, A., Upadhyay, S., Faruqi, M.: Fact, Fetch, and Reason: A Unified Evaluation of Retrieval-Augmented Generation. *arXiv preprint arXiv:2409.12941* (2024). <https://arxiv.org/abs/2409.12941>
10. LangChain Homepage, <https://www.langchain.com>, last accessed 2024/10/04
11. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., Kiela, D.: Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In: *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, pp. 9459–9474. Curran Associates, Inc., Red Hook (2020). <https://proceedings.neurips.cc/paper/2020/file/6b493230205f780e1bc26945df7481e5-Abstract.html>
12. Yan, S.Q., Gu, J.C. Gu, Zhu, Y., Ling Z.H.: Corrective Retrieval Augmented Generation *arXiv preprint arXiv:2401.15884* (2024). <https://arxiv.org/abs/2401.15884>
13. Liu, N.F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., Liang, P.: Lost in the Middle: How Language Models Use Long Contexts. *Transactions of the Association for Computational Linguistics* **12**, 157–173 (2024). <https://aclanthology.org/2024.tacl-1.9>
14. Mays, K.J.: *The Norton Introduction to Literature*. W.W. Norton & Company, New York (2015). <https://books.wwnorton.com/books/detail.aspx?id=4294990969>

15. McInnes, L., Healy, J., Melville, J.: UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *Journal of Open Source Software* **3**(29), 861 (2018). <https://doi.org/10.21105/joss.00861>
16. Merth, T., Fu, Q., Rastegari, M., Najibikohnehshahri, M.: Superposition Prompting: Improving and Accelerating Retrieval-Augmented Generation. In: *Proceedings of the 41st International Conference on Machine Learning (ICML 2024)*, pp. 1103–1125. PMLR (2024). <https://machinelearning.apple.com/research/superposition-prompting>
17. Meyer, M.: *The Bedford Introduction to Literature*. Bedford/St. Martin’s, Boston (2013). <https://www.macmillanlearning.com/college/us/product/Bedford-Introduction-to-Literature/p/1457608278>
18. NYU Machine Learning for Language: QuALITY: Question Answering with Long Input Texts. <https://nyu-ml.github.io/quality/>, last accessed 2024/09/25
19. OpenAI: Embedding Models Documentation. OpenAI (2023). <https://platform.openai.com/docs/guides/embeddings/embedding-models>, last accessed 2024/10/02
20. OpenAI: Models Documentation. OpenAI (2023). <https://platform.openai.com/docs/models>, last accessed 2023/10/04
21. Pang, R.Y., Parrish, A., Joshi, N., Nangia, N., Phang, J., Chen, A., Padmakumar, V., Ma, J., Thompson, J., He, H., Bowman, S.R.: QuALITY: Question Answering with Long Input Texts, Yes! In: Carpuat, M., de Marneffe, M.-C., Ruiz, I.V.M. (eds.) *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5336–5358. Association for Computational Linguistics, Seattle (2022). <https://aclanthology.org/2022.naacl-main.391>
22. Peng, B., Zhu, Y., Liu, Y., Bo, X., Shi, H.: Graph Retrieval-Augmented Generation: A Survey. *arXiv preprint arXiv:2408.08921* (2024). <https://arxiv.org/abs/2408.08921>
23. Gao, L., Ma, X., Lin, J., Callan, J.: Precise Zero-Shot Dense Retrieval without Relevance Labels. In: Rogers, A., Boyd-Graber, J., Okazaki, N. (eds.) *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1762–1777. Association for Computational Linguistics, Toronto (2023). <https://aclanthology.org/2023.acl-long.99>
24. Project Gutenberg: Project Gutenberg Free eBooks. <https://www.gutenberg.org/>, last accessed 2024/10/01
25. Sarthi, P., Abdullah, S., Tuli, A., Khanna, S., Goldie, A., Manning, C.D.: RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval. In: *Proceedings of the International Conference on Learning Representations (ICLR 2024)* (2024). <https://arxiv.org/abs/2401.18059>
26. Shahul, E., James, J., Espinosa-Anke, L., Schockaert, S.: 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*, pp. 162–171. Association for Computational Linguistics (2024). <https://aclanthology.org/2024.eacl-demo.16>
27. Su, W., Tang, Y., Ai, Q., Wu, Z., Liu, Y.: DRAGIN: Dynamic Retrieval Augmented Generation based on the Real-time Information Needs of Large Language Models. In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL 2024)*, pp. 12991–13013. Association for Computational Linguistics, Bangkok (2024). <https://aclanthology.org/2024.acl-long.702>

28. Wang, X., Wang, Z., Gao, X., Zhang, F., Wu, Y., Xu, Z., Shi, T., Wang, Z., Li, S., Qian, Q., Yin, R., Lv, C., Zheng, X., Huang, X.: Searching for Best Practices in Retrieval-Augmented Generation. arXiv preprint arXiv:2407.01219 (2024). <https://arxiv.org/abs/2407.01219>
29. Wang, Y., Lipka, N., Rossi, R., Siu, A., Zhang, R., Derr, T.: Knowledge Graph Prompting for Multi-Document Question Answering. In: Workshop on New Frontiers in Graph Learning, NeurIPS (2023). <https://neurips.cc/virtual/2023/82342>
30. Wang, Z., Lee, C.Y.: Speculative RAG: Enhancing Retrieval Augmented Generation through Drafting. arXiv preprint arXiv:2409.14924 (2024). <https://arxiv.org/abs/2409.14924>
31. Wu, J., Zhu, J., Qi, Y.: Medical Graph RAG: Towards Safe Medical Large Language Models via Graph Retrieval-Augmented Generation. arXiv preprint arXiv:2408.04187 (2024). <https://arxiv.org/abs/2408.04187>
32. Xu, Z., Cruz, M.J., Guevara, M., Wang, T., Deshpande, M., Wang, X., Li, Z.: Retrieval-Augmented Generation with Knowledge Graphs for Customer Service Question Answering. In: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2024), pp. 2905–2909. ACM, New York (2024). <https://arxiv.org/abs/2404.17723>
33. Sawarkar, K., Mangal, A., Solanki S.R.: Blended RAG: Improving RAG Accuracy with Semantic Search and Hybrid Query-Based Retrievers. arXiv preprint arXiv:2404.07220 (2024). <https://arxiv.org/abs/2404.07220>
34. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I.: *Improving Language Understanding by Generative Pre-Training* (2018). https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf