AGGILE: Automated Graph Generation for Inference and Language Exploration

Victoria Firsanova[1,*,†], Yana Khlusova[1,†]

[1]Higher School of Economics (HSE University), Griboedova emb. 123, St Petersburg, 190068, Russian Federation

Abstract

Knowledge graphs are widely used in Retrieval Augmented Generation (RAG) and Explainable AI (XAI), since they can illustrate semantic relationships generated by Large Language Models (LLMs). Recent studies focus on generating knowledge graphs from unstructured data to improve RAG performance; however, they do not explain the underlying graph structure. The analysis of synthetic graphs behind modern graph-based RAG systems shows that the structures generated by LLMs fail to capture semantic relations represented in the unstructured texts. The paper addresses these limitations by proposing a novel approach that enhances explainability and reduces hallucinations in knowledge graph generation for RAG systems. The paper introduces AGGILE, a tool for LLM-based graph generation from unstructured data and their visualization for enhanced explainability. Our approach extracts keywords and related concepts from unstructured data and generates predicates that denote semantic relations between the extracted entities. We evaluated our framework using qualitative linguistic criteria on a sample of unstructured Wikipedia data. The results of our qualitative analysis show that the AGGILE tool captures complex semantic relationships from unstructured data, allowing for in-depth LLMs' knowledge exploration. The method is scalable and domain-agnostic; however, a small rate of hallucinations occurs in the generated structures, and there are limitations in the input text size for proper graph modeling. Overall, this work contributes to improving the transparency and controllability of graph RAG. The AGGILE tool is open source: demo versions, source code, and software documentation are available in the project repository at https://github.com/vifirsanova/AGGILE/.

Keywords

Large Language Models (LLMs), Automated Graph Generation, Knowledge Graphs, Explainable AI (XAI)

1. Introduction

A knowledge graph is a data model that represents objects and relations between them with nodes and edges. In Artificial Intelligence (AI), knowledge engineering, and Natural Language Processing (NLP), knowledge graphs are often used to depict semantic and logical relations between concepts to represent data, cognitive structures, or common knowledge. Graphs are often applied to enhance search engines and information retrieval systems [1]. In recent years, knowledge graphs has been widely applied in Retrieval Augmented Generation (RAG).

Retrieval Augmented Generation (RAG) [2] is a type of NLP architecture that perform language generation conditioned by data retrieved from an external source, such as a vector database or knowledge graph [3]. RAGs are often applied in knowledge-intensive tasks, such as question-answering, and their performance is heavily dependent on the quality of content of the external knowledge source. The data structure determines the RAG performance, and knowledge graphs are popular choice as an external source for the retriever, because graphs allow extracting entities and their relations directly. Subgraphs extracted through graph-based RAG signify facts as subsets of entities consisting of interrelated items.

While there are various open-domain knowledge bases, such as Wikidata [4] and DBPedia [5], that use specific formats, like Resource Description Framework (RDF) or Extensible Markup Language (XML), they cannot be used for building a RAG system serving specific closed-domain field (e.g. a RAG

XAI-KG@ESWC2025: ESWC 2025 - 1st International Workshop on eXplainable AI and Knowledge Graphs

*Corresponding author.

†These authors contributed equally.

vfirsanova@hse.ru (V. Firsanova); yakkhlusova@edu.hse.ru (Y. Khlusova)

https://vifirsanova.github.io/ (V. Firsanova)

0000-0002-8474-0262 (V. Firsanova); 0009-0004-3619-8191 (Y. Khlusova)

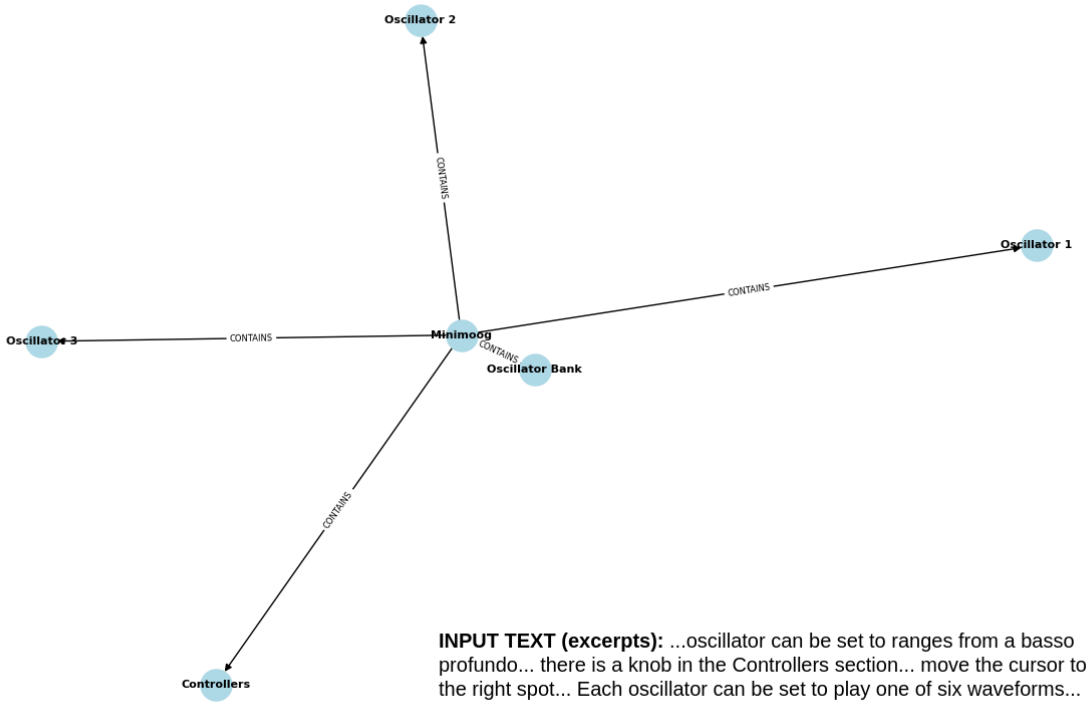

Figure 1: A GraphRAG visualization for a passage about Minimoog.

system for a company FAQ). Building a custom closed-domain knowledge graph is a resource-intensive task that requires complex data structuring and manual data handling. Some RAG approaches [6] propose automated graph generation from unstructured data. Automated graph generation improves RAG performance and simplifies model explainability through graph visualization.

From our empirical findings and observations made with GraphRAG API [7] and a GraphRAG visualization Gradio application [8], while automatically generated graphs successfully handle common relations, such as locations or addresses ('X is located in Y'), family relations ('X is a brother / sister of Y'), and hierarchical relations (e.g., see examples in LlamaIndex tutorial [9]), they often fail in capturing professional terms. For example, in the text describing Minimoog, an analog synthesizer, the model provides poor and uniformative graphs (see Fig. 1). In the provided example, the model repetitively extracts relation 'CONTAINS' for the node 'Minimoog', focusing on the most frequent terms related to synthesizers ('oscillator' and 'controllers'), while ignoring other important technical terms, such as 'knob', 'cursor', or 'waveforms'.

We suggest that the limitation is caused by the fact the graph structure is generated end-to-end, i.e. the graph is resulted from the input data directly without fragmenting the graph into its structural elements (nodes and edges). The study hypothesis is that knowledge graph generation can become more controllable, providing more advanced explainability options through visualization tools, if the graph synthesis implies sequential generation of a graph components, i.e. nodes and edges. The study proposes a novel approach called Automated Graph Generation for Inference and Language Exploration (AGGILE) that uses prompt-tuning to build knowledge graphs from unstructured data sequentially using the following algorithm:

- Extract entities from the unstructured data to form graph nodes
- Extract related items for the extracted entities (related nodes)
- Generate predicated between the extracted entities and the related items (edges)
- Form triplets from two nodes and edges between them

- Connect the triplets to a graph
- Set weights to edges according to the number of connections with other nodes

The resulting structure is a weighted graph, where the nodes with more relations get higher weights. This model allows for extracting topics, finding clusters and searching through subgraphs, as well as visualizing weighted graphs allowing for fine-grained RAG explainability analysis. The AGGILE tool provides the following functions:

- Graph structure generation from unstructured data
- Interactive graph visualization
- Graph-based RAG function

The approach is evaluated using linguistic qualitative analysis based on criteria involving searching for noise in the generated data, assessing nodes' and edges' accuracy and relevance to the data topic, as well as evaluating graph scalability. The methods from linguistics are used to explore the semantic relations built with the AGGILE tool, analyze the word forms and parts-of-speech used by the model to denote nodes and edges, and identify LLM hallucinations and logical errors.

The AGGILE tool is fully compatible with the HuggingFace infrastructure. The tool is open source and available in the project repository at https://github.com/vifirsanova/AGGILE/. The tool generates JSON object with graph structure, HTML file with interactive graph built with Plotly, and answer to user question. The tool supports a wide range of LLMs, while the experiments described in this paper are based on DeepSeek-R1-Distill-Qwen-32B model, a distilled model from DeepSeek-R1 based on Qwen.

The study contributions are the following:

- A novel tool for RAG explainability based on automated knowledge graph generation and visualization
- A novel method for generating knowledge graphs from unstructured data based on sequential nodes and edges synthesis for RAG and XAI
- A novel approach to qualitative knowledge graph assessment for XAI based on linguistic theory

Overall, AGGILE provides a toolkit for automated graph generation, explainability analysis and RAG implementation. The study findings show that sequential generation of graph components from unstructured data allows for building scalable RAG models with high explainability potential, enabling transparent and controllable language generation.

The AGGILE tool demonstrate the ability to highlight target domain and topical clusters, as well as filter irrelevant information. The model can produce weighted relations, allowing to extract subgraphs and learning how LLMs prioritize the information during the processing. Also, the model can extract diverse semantic links, for example, hierarchical and associative relations.

However, the proposed method has several limitations. The model implies several queries to the LLM, increasing the time required to generate the graph. This limits the AGGILE efficiency, however, we plan to solve the issue by using vector representations directly instead of using text-based prompt-tuning. Also, the model shows some logical errors in forming edges between the nodes, and tends to generate relations between identical nodes. The latter problem was fixed by applying a simple rule for filtering triplets with duplicating nodes.

In perspective, we plan to test the AGGILE performance on popular RAG benchmarks, expland the tool interface with tablular representation of the generated nodes and edges, and present an analytical framework for LLMs' assessment based on the developed linguistic criteria for graph-based XAI.

2. Related Work

A knowledge graph (KG) is a structured representation of real-world entities and the relationships between them [10]. In the context of our work, entities are understood as keywords and their semantically related terms. Formally, a knowledge graph can be expressed as follows: $KG = ((kw), (rw), (l))$,

where kw represents the set of keywords, rw denotes the set of related words, and l signifies the set of semantic links connecting these concepts.

The semantic linking methodology employed in this study is inspired by WordNet [11], a well-known lexical database. WordNet organizes synonyms into sets interconnected via various semantic relations such as meronymy/holonymy, antonymy, hyponymy/hypernymy, troponymy, and entailment. A target word is understood as a keyword. The keyword has some related words, e.g. particular synonyms or antonyms. We rely on this terminology.

A triplet (or triple) is a part of KG (subgraph), consisting of a keyword (*subject*), a related word (*object*) and the semantic relation (*predicate*) between them: $(s, p, o) = (subject, predicate, object)$. Some examples of such triplets include: Mathematics (s) is a branch of (p) Number Theory (o), Geometry (s) studies (p) Shapes (o), Analysis (s) is a branch of (p) Mathematic Logic (o).

Retrieval-augmented generation (RAG) is a technique designed to enhance the performance of large language models (LLMs) by integrating them with an external knowledge base (KB) [2]. In its simplest form, a RAG system processes a user query by vectorizing it, retrieving the most relevant information from a pre-vectorized KB via semantic search, and generating a response based on the retrieved content.

An advanced variant of RAG, known as GraphRAG [6], retrieves data from a KG. In this approach, LLM is employed to extract entities and relationships from source documents to construct a KG. Then the communities of the KG are summarized to generate response to user query. This method has demonstrated notable improvements in question-answering (QA) task compared to conventional semantic search-based RAG systems.

SubgraphRAG [12] introduces a method for answer generation based on subgraphs extracted from the main KG using a lightweight multilayer perceptron (MLP) combined with parallel triple-scoring. The authors argue that LLMs should focus on interpreting data rather than constructing graphs. Explanations are produced from relevant triplets, and the method achieves results that are either better or comparable with state-of-the-art (SOTA) approaches.

G-Retriever [13] similarly works with subgraphs but uses language model for indexing and data retrieval. It further assesses the relevance and optimal size of subgraphs using the Prize-Collecting Steiner Tree algorithm, returning the most pertinent subgraph to explain the LLM's reasoning process. The system architecture was evaluated in three ways: inference with a frozen LLM, prompt-tuning, and fine-tuning with LoRA. The highest evaluation scores were observed in the fine-tuning scenario. As a result, G-Retriever was able to decrease tokens, nodes and training time. Additionally, the study highlights the challenges posed by KG complexity and data noise, both of which can negatively affect answer quality.

While the aforementioned works primarily focus on KG-based information retrieval, the following studies concentrate on KG generation.

In the first one [14] LLM predicts road users behavior and provides explanations using RAG. Video content is converted into linguistic features, entities are extracted through deep learning techniques. The resulting KG is constructed using the Ampligraph 2.0.0 library [15] in the form of triplets describing the scene from the video. Explanations are provided in two ways: using fuzzy-logic and retrieved linguistic features. This approach demonstrated notable improvements in prediction accuracy compared to conventional methods.

The second study is devoted to explanation generation for learning recommendations [16, 17] based on the custom KG. The KG is constructed from structured data comprising learning object titles (short text headings) and their extended textual descriptions. Titles are embedded using Sentence-BERT and SpaCy. Keywords are extracted from descriptions using KeyBERT and then embedded. The overall quality of explanations provided by LLM was improved; however, learning experts noted issues related to the phrasing of generated responses and a lack of high-level abstraction in the explanations.

None of the aforementioned works explicitly addresses the problem of explainability in graph generation itself. They are aimed to assess LLM's outputs and reasoning without identifying the specific data components contributing most to KG construction. Our work is devoted to exploration of this issue. Moreover, it is designed to handle unstructured user data and introduces a more controllable graph generation method through structured prompt engineering.

3. Method

The AGGILE is a Python-based tool designed to generate knowledge graphs from unstructured data for the purposes of XAI. The graphs are generated from plain text using LLMs.

The tool generate triplets, which is a set of 2 nodes and 1 edge denoting a relation between them. Each triplet has the following form: subject-predicate-object, where subject and object are nodes, and predicate is an edge.

The tool is based on prompt-tuning methods, prompts are used to guide LLMs in identifying key lexical entities to form the graph. The AGGILE Python class consists of methods for extracting entities, identifying relations between them and forming triplets, as well as constructing and visualizing graph in HTML-format.

3.1. Algorithm

The AGGILE Python class is built around three core functionalities:

- Entity extraction: identifying key concepts, named entities, and keywords from the input text
- Relation extraction: determining relationships between extracted entities
- Graph construction: visualizing the extracted entities and relationships as a knowledge graph

First, the tool extracts N keywords, named entities or produces semantic concepts consisting of one word or a phrase based on the provided text. N is the number of instances defined by user (10 by default). The process is done through LLM prompt-tuning (see Appendix A "Subject extraction"). This list is forms a list subject, and it will contain a list of nodes for the graph.

Next, the tool extracts 5-10 related words for each primary node (subject) and produces a predicate between them. Predicates define the relationship between subjects and objects as a verb or a phrase describing it. The list of secondary nodes denoting related words forms the list of objects, and a list of predicates form the edges of the graph. See Appendix A "Object extraction" and "Predicate generation".

This way of generating graphs is more controllable and predictable due to the ordered prompt structure. LLM inference is implemented through HuggingFace Hub's InferenceClient, see the project repository for more details.

As a result, triplets are combined into a JSON object used to construct a graph. Visualization is made in HTML-format using Plotly library. Nodes' colour depends on their degree: the more relations has the node, the higher weight is assigned to it. The weight defines the nodes' degree.

The AGGILE class also provides a method for implementing RAG based on the generated graphs. However, this study focuses on the explainability potential of the developed tool.

3.2. Model parameters

The LLMs in AGGILE are set with the following parameters by default:

- max_tokens = 1024 for subjects' extraction,
- max_tokens = 512 for objects' extraction and predicates' generation,
- temperature = 0.5 in all cases,
- top_p = 0.1 in all cases.

Either of three lists (subjects, objects, predicates) are generated as JSON objects. The resulting set of prompt templates may be found in Appendix A.

4. Evaluation Setup

The method was evaluated on several texts accessed through Wikipedia API. The experiments were conducted using Deepseek-ai/DeepSeek-R1-Distill-Qwen-32B [18]. The aim of the procedure was to assess the generated graphs qualitatively according to the following criteria:

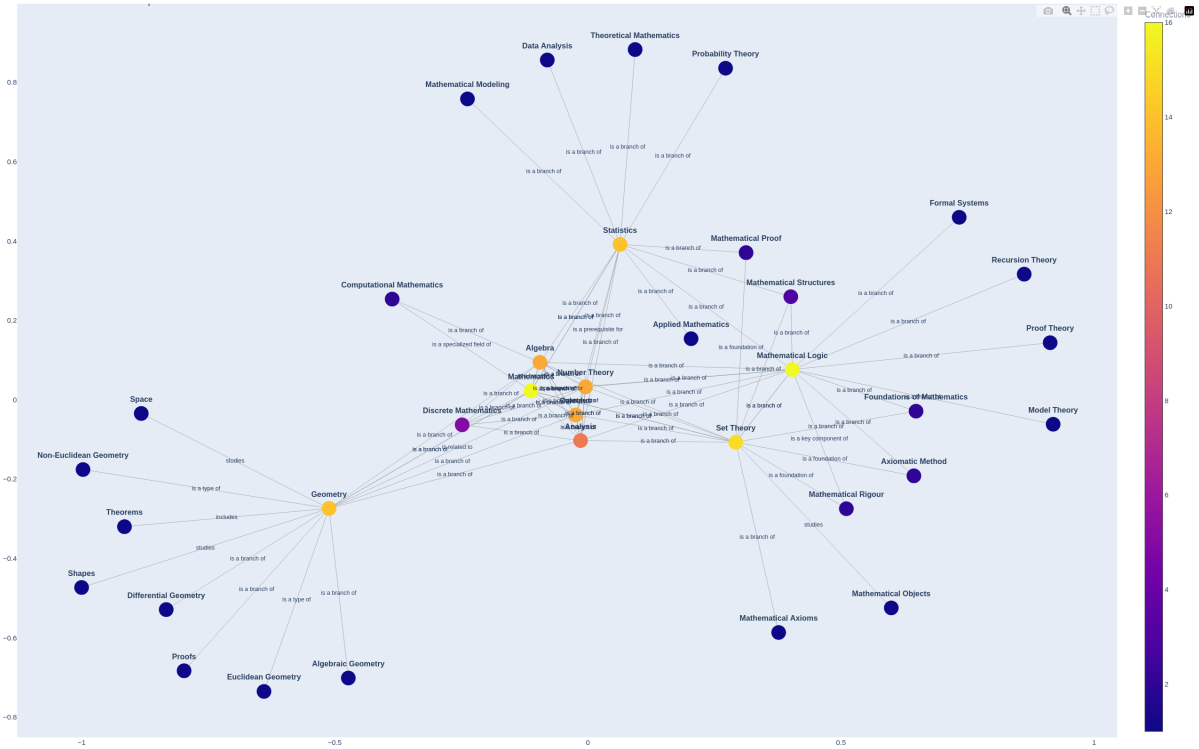

Figure 2: The example of graph visualization based on Wikipedia article about mathematics.

- noise in data,
- accuracy of data representation,
- hallucinations in comparison to source text,
- relationships and their relevancy,
- nodes representation (which words are used for this purpose),
- comprehensiveness,
- interpretability,
- scalability.

5. Results and Discussion

5.1. Qualitative Analysis

The first graph was (see Fig. 2) constructed using the Wikipedia article *Mathematics*. Two dominants are observed: Mathematics itself and Mathematical Logic. These nodes exhibit the highest number of related nodes, thereby achieving the maximum degree score. All other nodes represent objects related to them, although the source text contains more entities from other disciplines such as engineering, medicine, natural sciences, etc. This fact suggests that the model labeled these concepts as the most significant during the graph generation process.

As for relations, mainly the triplet 'S is a branch of O' is used, however, there are also some other predicate types including *studies*, *is a type of*, *includes*, *is related to*, *is a specialized field of*, *is a prerequisite for*, *is a key component of*, *is a foundation of*. Semantically, most of these relations represent meronymic or holonymic structures (i.e., part–whole relationships), with the exception of *studies*, which reflects agent-object relationships.

Figure 3 illustrates the graph constructed from the Wikipedia article *Pug*. This graph demonstrates greater granularity due to the structure of the source text, which contains not only a description of the breed's physical characteristics but also historical information regarding its origins. Following the

Figure 3: The example of graph visualization based on Wikipedia article about pugs.

provided source, the graph is organized into two thematic clusters, connected by the most significant node, *Pug*. The nodes above Pug represent the breed's physical traits. Notably, the term *Pekingese* appears within this cluster, despite the fact that there is no other direct relationship between pekingeses and pugs apart from the fact that both 'are the breeds of dogs', as presented in the graph. Most likely the model included Pekingese because it has the similar appearance as Pug, according to Wikipedia page.

The 'historical' cluster includes the Second Opium War, its events, and information about the pug's ancestor, the Happa or Hap-pah dog, as well as the role of pugs in the lives of Tibetan monks.

The nodes are denoted with the key entities including the target Pug, its ancestors and Pekingese, the Second Opium War, Chinese emperors, geographical places, monks and monasteries. An inaccuracy in nodes extraction is observed: there is the node *Pug* as well as the node *Pugs*. Another cases include *Hap-pah dog* and its variations: *Happa*, *Happa dog* and *Happa or Hap-pah dog*; *Lo-sze*, *Lo-Sze*, *Lo-Chiang-Sze*; *Old Summer Palace*, *Old Summer Palace (Yuanmingyuan)* and *Yuanmingyuan*. The model does not focus on clustering different names of the same objects, probably because it is not queried in the prompt (see Appendix A).

Relations in the graph are represented in a variety of ways:

- using a verb: *train*, *exhibits*;
- using a verb with preposition or particle: *is associated with*, *lived in*, *unrelated to*;
- using a verb, a preposition and a noun: *has no direct relationship with*;
- using a noun and a preposition: *ancestor of* ;
- sometimes, the whitespace between words is replaced with an underscore: *involved_participants*, *has_short_coat*.

It is not clear at this moment how LLM decides between whitespace and underscore, or how it determines whether to use a particle or preposition: *unrelated* VS. *unrelated to*. The grammatical features of predicates depend on the subjects and objects (see 'has' or 'have'). There are past and present forms of verbs, passive constructions, the difference between singular and plural entities.

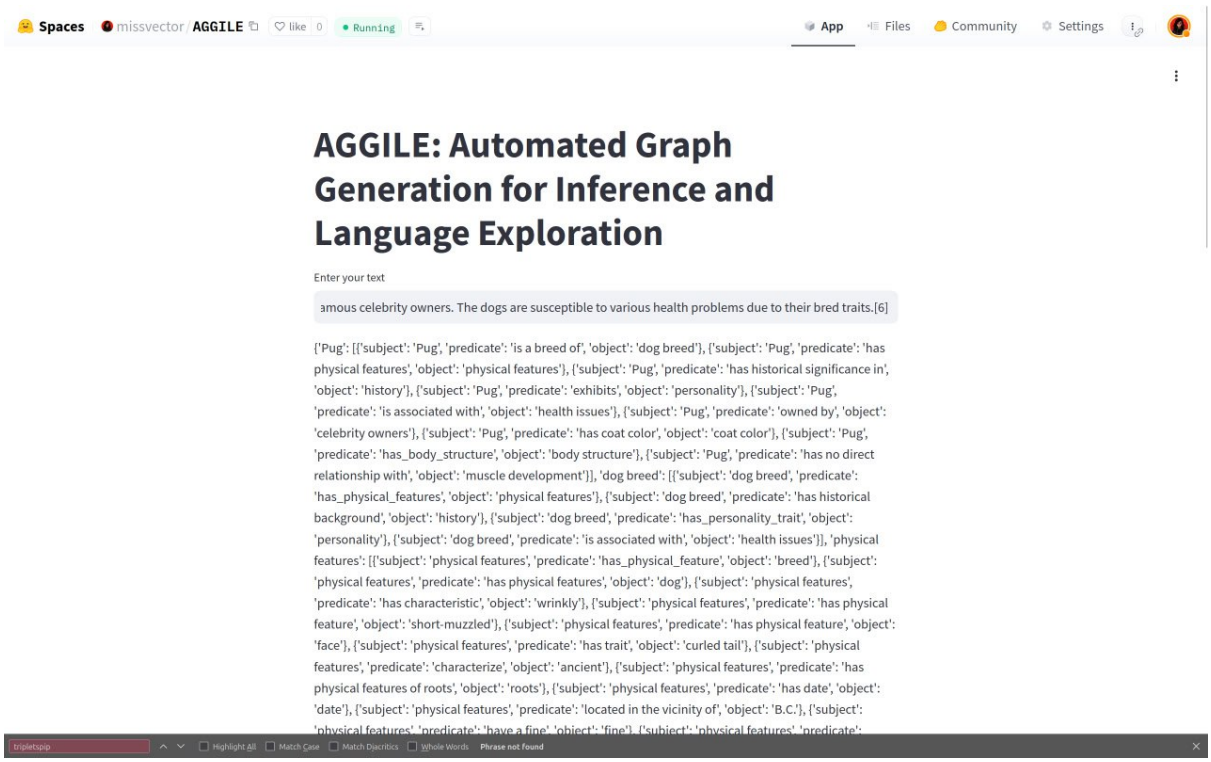

Figure 4: AGGILE graphical user interface, part 1.

The list of inaccuracies in generating predicates includes the following. Firstly, there is an instance of predicate where the node is mentioned. The relation between *monasteries* and *Pugs* is expressed as *monasteries often have*. This forms a triplet *monasteries-monasteries often have-Pugs*. Secondly, an illogical relation is used — *has no direct relationship with*. On the one side, this may be considered as a hallucination since the model is asked to extract related concepts. On the other side, the fact that a subject and an object have nothing in common suggests that this is indeed the relationship between them. Lastly, once incorrect information is given: *Tibet is a type of royal dogs*.

It remains unclear why, despite the comparable length of the Wikipedia articles on Mathematics and Pug, the former resulted in a smaller graph containing fewer concepts. The possible explanation is that the article about Pug contains a broader range of topics, while Mathematics page is devoted only to discipline and its elements. This interpretation is supported by the relations generated by LLM; as was already mentioned, the majority of predicates in the Mathematics graph are expressed 'as a branch of', reflecting a more hierarchical, discipline-centered structure.

Talking about scalability, the method has demonstrated adaptability to various topics, including animals, history, and science, as concise graphs were generated for all of them. This fact proves that our method is applicable to texts across different domains.

Discussing the limitations of the method, the criteria by which the model selects entities for triplets are not clear. In some cases, the subject and object coincide, or the hierarchical relationship between them lacks clarity. Moreover, as previously mentioned, the model does not cluster the names of the same object. Secondly, the actual version of the instrument requires three queries to LLM's reasoning, which increases computational time. Addressing these limitations will be a focus of our future research.

5.2. User Interface

A graphical user interface was built, using Plotly, Streamlit, and HuggingFace. It is shown on Figures 4 and 5.

User is asked to provide a text into the window and press the Enter button. Then the text is processed and HTML with the visualization is returned. Visualized graph may be zoomed. The color of nodes

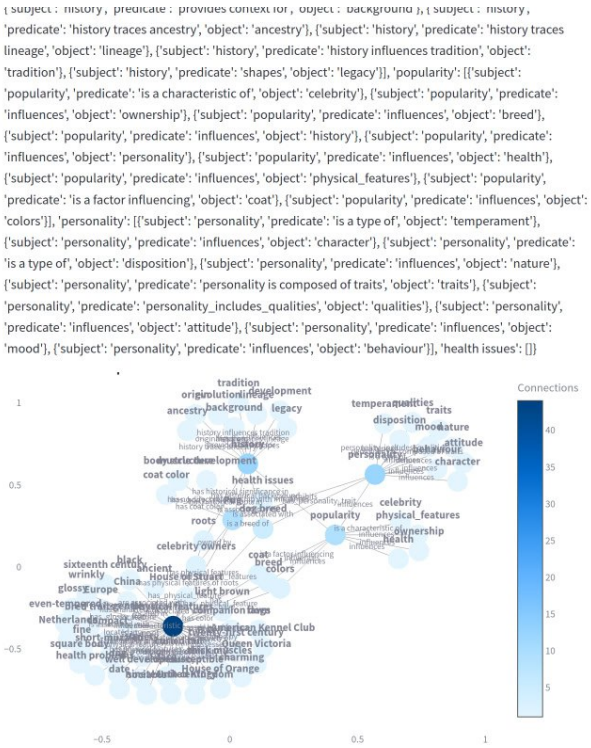

Figure 5: AGGILE graphical user interface, part 2.

represents their degree. This functionality was achieved with Plotly.

Additionally, our app supports the function for graphRAG. An answer to user query is provided on the basis of both provided source and generated triplets. However, it is not evaluated yet and is a next step of our work.

The app is hosted on HuggingFace Spaces and built with Streamlit.

6. Conclusion

The paper presents a novel tool for RAG explainability, AGGILE. The model generates knowledge graphs from unstructured data and visualizes the results using Plotly. The study describes primary qualitative analysis based on linguistic criteria developed by the authors of the paper. The results of qualitative evaluation show that the method is scalable, suggesting that AGGILE is a general-purpose explainability instrument suitable for wide range of domains.

We have proven that shifting from end-to-end graph generated to a more fine-grained procedure, where the process of graph generation is split into separate stages (sequential node extraction and edge generation) lead to improved controllability, enhanced transparency and decreased hallucination ratio.

The proposed approach have notable advantages. The model demonstrate the ability to focus on key concepts, define target domain, and filtering irrelevant entities, ensuring disambiguation (contextual understanding) and the possibility of finding the root concepts from the point of view of the LLMs' explainability.

The approach can produce fine-grained graph structures, producing subgraphs, clusters, and complex relations. The weighting technique introduced in the paper allows for using the AGGILE tool for assessing concepts that gain more focus by the LLM during the processing.

The model is capable of extracting diverse relations, including hierarchical (e.g. 'a branch of'), associative ('is associated with'), and descriptive ('exhibits') meanings. The method can be applied to a wide range of topics, suggesting that the proposed tool is a general-purpose explainability instrument.

The method has several limitations. For example, the model execution time is significant; the model

generates a graph structure for approximately 3 minutes, since it queries LLM 3 times to generate 2 sets of nodes and 1 set of edges. In perspective, we plan to use vector storage for prompt-tuning to perform all intermediate processes (generation of nodes and edges) directly through vectors, reducing the time required for vectorization to optimize the model processing time.

Also, the model struggles with recognizing word forms, synonyms, and variations of the same lexical entity (e.g., 'Hap-pah dog' and 'Happa dog'). The model might produce inconsistent redundant edges, e.g. 'is a branch of' or 'studies' for the same semantic relation. This indicates that a study towards improving the control over nodes and edges generation is required.

Despite the decreasing amount of hallucinations in our graphs, some generated graphs still have logical errors and hallucinations, especially in forming edges between the nodes. This can misrepresent the source data.

In some cases, the model tends to generate relation between one and the same entity, however, this issue was fixed by a simple heuristics. We added a rule that ensures that 2 nodes in a triplet are always different.

The method allows for building representative graphs for with considerably long and contextually-rich texts. The produced graphs reflect both semantical and grammatical features of the input data, as well as indicate, which parts of input data target the model output, contributing to XAI methodology.

In perspective, we plan to develop methods for enhancing the AGGILE control over LLM outputs that would result in reducing redundancies, filtering hallucination and improving the model consistency. A possible solution is to define a standardized set of relations and semantic hierarchy in a form of rules for prompt-tuning and response formatting through formal grammars and conditional text generation.

Next, we plan to perform the model optimization through switching to intermediate vector representations from text-based prompt-tuning in generating nodes and edges. This step should reduce the model execution time and enhance user experience and the model scalability.

Finally, we plan to develop a multilingual version of the AGGILE tool, and add such features as generating tables with nodes and edges for improved explainability.

Declaration on Generative AI

During the preparation of this work, the authors used GPT-4 in order to: Grammar and spelling check. After using these tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] L. Ehrlinger, W. Wöß, Towards a definition of knowledge graphs., SEMANTiCS (Posters, Demos, SuCCESS) 48 (2016) 2.

[2] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021. URL: https://arxiv.org/abs/2005.11401. arXiv:2005.11401.

[3] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, H. Wang, H. Wang, Retrieval-augmented generation for large language models: A survey, arXiv preprint arXiv:2312.10997 2 (2023).

[4] F. Erxleben, M. Günther, M. Krötzsch, J. Mendez, D. Vrandečić, Introducing wikidata to the linked data web, in: The Semantic Web–ISWC 2014: 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I 13, Springer, 2014, pp. 50–65.

[5] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, S. Hellmann, Dbpedia-a crystallization point for the web of data, Journal of web semantics 7 (2009) 154–165.

[6] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, D. Metropolitansky, R. O. Ness, J. Larson, From local to global: A graph rag approach to query-focused summarization, 2025. URL: https://arxiv.org/abs/2404.16130. arXiv:2404.16130.

[7] GitHub - noworneverev/graphrag-api: GraphRAG Server — github.com, https://github.com/noworneverev/graphrag-api, 2024. [Accessed 15-03-2025].
[8] GitHub - GirishWangikar/GraphRAG-Visualization-System — github.com, https://github.com/GirishWangikar/GraphRAG-Visualization-System, 2024. [Accessed 15-03-2025].
[9] Knowledge Graph RAG Query Engine - LlamaIndex — docs.llamaindex.ai, https://docs.llamaindex.ai/en/stable/examples/query_engine/knowledge_graph_rag_query_engine/, ???? [Accessed 15-03-2025].
[10] C. Peng, F. Xia, M. Naseriparsa, F. Osborne, Knowledge graphs: Opportunities and challenges, 2023. URL: https://arxiv.org/abs/2303.13948. arXiv:2303.13948.
[11] G. A. Miller, Wordnet: a lexical database for english, Commun. ACM 38 (1995) 39–41. URL: https://doi.org/10.1145/219717.219748. doi:10.1145/219717.219748.
[12] M. Li, S. Miao, P. Li, Simple is effective: The roles of graphs and large language models in knowledge-graph-based retrieval-augmented generation, in: The Thirteenth International Conference on Learning Representations, 2025. URL: https://openreview.net/forum?id=JvkuZZ04O7.
[13] X. He, Y. Tian, Y. Sun, N. Chawla, T. Laurent, Y. LeCun, X. Bresson, B. Hooi, G-retriever: Retrieval-augmented generation for textual graph understanding and question answering, in: A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, C. Zhang (Eds.), Advances in Neural Information Processing Systems, volume 37, Curran Associates, Inc., 2024, pp. 132876–132907. URL: https://proceedings.neurips.cc/paper_files/paper/2024/file/efaf1c9726648c8ba363a5c927440529-Paper-Conference.pdf.
[14] M. M. Hussien, A. N. Melo, A. L. Ballardini, C. S. Maldonado, R. Izquierdo, M. Ángel Sotelo, Rag-based explainable prediction of road users behaviors for automated driving using knowledge graphs and large language models, Expert Systems with Applications 265 (2025) 125914. URL: https://www.sciencedirect.com/science/article/pii/S0957417424027817. doi:https://doi.org/10.1016/j.eswa.2024.125914.
[15] L. Costabello, A. Bernardi, A. Janik, A. Creo, S. Pai, C. L. Van, R. McGrath, N. McCarthy, P. Tabacof, AmpliGraph: a Library for Representation Learning on Knowledge Graphs, 2019. URL: https://doi.org/10.5281/zenodo.2595043. doi:10.5281/zenodo.2595043.
[16] H. Abu-Rasheed, M. Dornhöfer, C. Weber, G. Kismihók, U. Buchmann, M. Fathi, Building contextual knowledge graphs for personalized learning recommendations using text mining and semantic graph completion, in: 2023 IEEE International Conference on Advanced Learning Technologies (ICALT), IEEE, 2023, p. 36–40. URL: http://dx.doi.org/10.1109/ICALT58122.2023.00016. doi:10.1109/icalt58122.2023.00016.
[17] H. Abu-Rasheed, C. Weber, M. Fathi, Knowledge graphs as context sources for llm-based explanations of learning recommendations, in: 2024 IEEE Global Engineering Education Conference (EDUCON), IEEE, 2024, p. 1–5. URL: http://dx.doi.org/10.1109/EDUCON60312.2024.10578654. doi:10.1109/educon60312.2024.10578654.
[18] DeepSeek-AI, Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL: https://arxiv.org/abs/2501.12948. arXiv:2501.12948.

A. System Prompt Templates

The AGGILE class is initialized with the following prompts:

Subjects extraction:

```
"""
extract {n} collocations describing key concepts,
keywords, named entities from the provided source
"""
```

Objects extraction:

```
"""
extract 5-10 most representative collocations
from the provided source that are related to the provided concept
"""
```

Predicates generation:

```
"""
define the relationship between two words:
generate a verb or a phrase describing a relationship between two entities;
return a predicate for a knowledge graph triplet
"""
```

Graph-based question-answering (not tested):

```
"""
answer the question using graph triplets and provided source
"""
```