MeGraS: An Open-Source Store for Multimodal Knowledge Graphs

Luca Rossetto
Dublin City University
Dublin, Ireland
luca.rossetto@dcu.ie

Florian Ruosch
FPR Consulting
Zurich, Switzerland
ruosch@ifi.uzh.ch

Abstract

Multimodal knowledge graphs often separate easily represented information (text) from that which is not (multimedia documents like images, videos, or audio). This severely limits query expressiveness, as the engines lack access to the node contents stored externally. We present **MeGraS**, the **Me**dia**Gra**ph **S**tore, a novel storage and query engine for multimodal knowledge graphs. By storing multimedia documents directly in the graph, MeGraS allows the query engine to leverage their content for enhanced capabilities, making it natively capable of performing operations such as k-NN, segmentation, or deriving non-materialized relations based on visual features. To demonstrate this, we incorporate and extend the pattern-matching query language SPARQL, resulting in a unified framework for storing and managing multimodal knowledge graphs with advanced expressiveness. MeGraS is available as open-source software: http://megras.org

CCS Concepts

• **Information systems** → *Users and interactive retrieval*; **Graph-based database models**; **Multimedia and multimodal retrieval**.

Keywords

Multimodal Knowledge Graphs, Graph Store, Multimodal Media Segmentation

ACM Reference Format:
Luca Rossetto and Florian Ruosch. 2025. MeGraS: An Open-Source Store for Multimodal Knowledge Graphs. In *Proceedings of the 33rd ACM International Conference on Multimedia (MM '25), October 27–31, 2025, Dublin, Ireland.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3746027.3756872

1 Introduction

With the rise of multimodal data, traditional knowledge graphs increasingly exhibit limitations in effectively representing and querying multimedia information. While they remain powerful tools to represent the seemingly unending floods of knowledge in a structured way, they struggle to incorporate modalities other than text. Even though multimodal knowledge graphs offer methods to incorporate multimedia documents into the graph, they often do so by using external URI references or storing media in unstructured blobs. Consequently, they deny the underlying query engine access to their contents, severely limiting expressiveness.

MM '25, Dublin, Ireland

ACM ISBN 979-8-4007-2035-2/2025/10
https://doi.org/10.1145/3746027.3756872

To address this issue, we present **MeGraS**, short for **Me**dia**Gra**ph **S**tore, and aim to elevate multimedia documents to first-class citizens in multimodal knowledge graphs. MeGraS consists of an integrated data storage and a custom query processing engine, handling the graph and its media components with the Resource Description Framework (RDF).[1] Therefore, the entire directed graph consists of triple statements (subject, predicate, object), the parts of which are all identified by a URI. While the data storage separates the triples and the multimedia content, the latter remains linked by being kept in a dedicated file store and the former in the database.

MeGraS not only provides means for querying but also for manipulating multimedia documents. In keeping with the Unified Media Segmentation [9], they can be segmented in temporal, spatial, or both dimensions, further enhancing the data management aspect. This also addresses limitations in existing approaches by avoiding rigid constraints on media and segmentations. Instead, it offers a formal and content-agnostic model combined with a practical URI scheme for representing segments, contributing to granular and, hence, more expressive querying.

Our contributions with MeGraS are as follows:

(1) Direct storage of multimodal documents within the graph
(2) Consistent URI scheme for media addressing
(3) Support for spatio-temporal segmentation of media
(4) Extended SPARQL capabilities
(5) An open-source framework for multimodal knowledge graphs

MeGraS is available to compile from source[2] or using a Docker container[3] for convenience.

2 MeGraS: MediaGraph Store

The key features of MeGraS include integrated multimodal storage, which allows documents to be directly stored within the graph, and consistent URI-addressing, enabling direct addressing of media and segments. It also offers spatiotemporal segmentation to support fine-grained media segment access, and extended SPARQL querying for multimodal-aware capabilities.

As an overview, a high-level system architecture diagram of MeGraS can be found in Figure 1. It shows the system at the center, the three interaction methods on the left, and the components it interacts with. The remainder of this section describes them all in order.

2.1 Storage

The heart of MeGraS is the data management: the storage of the knowledge graph and, explicitly, the linked multimedia documents.

[1] https://www.w3.org/RDF
[2] http://megras.org
[3] http://megras.org/docker

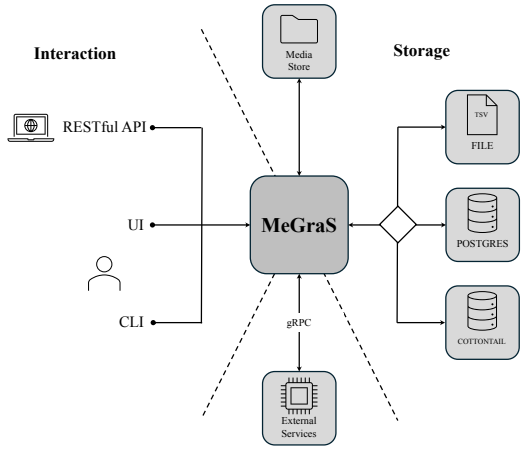

Figure 1: High-level system architecture diagram of MeGraS.

As displayed in Figure 1, the two are separated, but the engine still retains access to the content of the documents. When adding an item to MeGraS, it is assigned a URI for reference, its graph triples are written to the configured backend, and the document contents are replicated to the media store.

The nature of the triple store depends on the configuration, with three options being available. First, the `FILE` method wherein the graph is held in memory and saved to a tab-separated values file, suitable for smaller graphs and testing purposes. Second, `COTTONTAIL` uses Cottontail DB [2] as the backend to store the knowledge graph. Third, `POSTGRES` employs the open-source object-relational database PostgreSQL to persist triples. Furthermore, we leverage the pgvector[4] extension for more efficient vector operations, enabling faster similarity search or k-nearest-neighbor queries. While the first option supports knowledge graphs with triple numbers up to the six-digit range, the two databases are aimed at production use with large graphs. Cottontail can be used for knowledge graphs with up to a few million triples, whereas PostgreSQL can sustain several tens of millions of triples.

To address the multimedia documents in the knowledge graph, a URI is created upon adding them to the graph. This URI consists of the configured hostname, port, and a generated document identifier, making it a local value that indicates to the query engine that the document contents are available, as opposed to external URIs. To make the URI scheme consistent, interoperable, and human-readable, we employ multihash.[5] The identifiers for the documents, which are part of the URI, are produced by hashing their contents as a file stream with SHA-256, then encoding the resulting digests into `base64` and removing the padding. With this process, the string representation adheres to common URI requirements, being composed of uppercase and lowercase letters (A-Z and a-z), digits (0-9), and the symbols + and /. In contrast, the "raw ID," used for placing the document into media storage, is the raw digest encoded in `base32`, thus avoiding issues with operating system compatibilities and relying only on capital letters and digits. At the same time, building a multi-level folder tree based on this identifier distributes the files uniformly and maintains fast file access. Finally, multimedia segments created from documents through the Unified Media Segmentation [9] are also assigned URIs. To easily distinguish them from proper documents, their identifier is a composite of the ID of their parent and the hash of the string of the document ID and a description of the segmentation. Hence, they are also URI-addressable.

[4] https://github.com/pgvector/pgvector
[5] https://github.com/multiformats/multihash

2.2 Queries

The brain of MeGraS is its query engine: a sophisticated component designed to provide comprehensive and flexible access to the multimodal knowledge graph. Using a layered approach allows for efficient filtering of the information, making not only the media content available but also its inherent semantics. The robust underlying storage manages the defined RDF triples and provides means for precise retrieval. It enables lookups based on subject, predicate, or object, as well as lists and combinations thereof. Queries against these assertions form the foundation of the engine's capabilities, with its performance depending on the configured triple store and the associated indexes.

Beyond the explicitly stored information, MeGraS extends its query power to non-materialized triples. To this end, we distinguish two types of relations: implicit and derived. Implicit relations are derived from facts already in the existing knowledge graph based on the respective implemented logic of its reasoner and are always between two nodes. Spatial or temporal relations [3] are natural candidates for implicit relations as they can be inferred from already attached properties. For example, there may be an `after`-relation between two documents, indicating that one event takes place later in time than another. Similarly, the information for k-nearest neighbors is also already available, but the exact results need to be computed. Implicit relations are never materialized in the knowledge graph as they might change based on the addition or removal of data.

Meanwhile, derived relations can be but are not necessarily part of the graph. If the query engine does not find them for a given subject, they are computed and persistently added to reduce computational load. Derived relations are the results of predefined functions evaluated at query time, mostly dependent on the contents of the multimedia documents. Embedding vectors and extracted text are examples of derived relations.

Consequently, MeGraS executes complex and expressive queries that seamlessly combine explicit facts with dynamically generated insights from multimodal data. Its sophisticated, layered architecture supports advanced operations that can process properties, literals, and multimedia content, resulting in a powerful interface for exploring and reasoning over multimodal knowledge graphs.

2.3 Interaction and Manipulation

The interaction with MeGraS happens, as shown in Figure 1, through three different endpoints. First, a simple command-line interface (CLI) is available for adding multimedia files and importing RDF triples with their respective instructions.

Second, the RESTful API, designed with ease of use and interoperability in mind, is aimed at client applications. It offers a wider

range of functionalities. Endpoints are available for data ingestion, allowing the addition of media files to the store and inserting RDF triples for semantic descriptions. Furthermore, users can retrieve various representations of objects, including raw media, previews, and metadata. The most extensive features are found in the query interfaces. They enable filtering for subjects, predicates, and objects, as described above. Moreover, a dedicated API is available for queries using SPARQL and its extended functionality.

Another important aspect of MeGraS's API is the Unified Media Segmentation [9], which it enables. This is central to elevating multimodal documents to first-class citizens within the knowledge graph. Moving away from viewing them as atomic blobs, MeGraS allows for the identification and precise addressing of specific parts or regions within them. These can be defined in various ways, such as dimensional (spatio, temporal) or content-based (channels, intervals). Every newly generated segment is added to the knowledge graph and, while linked to a parent, treated as its independent document with the accompanying properties. As described above, relationships between these segments can be leveraged for more fine-grained insights, giving semantic meaning to details that would otherwise remain inaccessible.

Finally, there are user interfaces (UI) for some core functionalities of MeGraS. These include displaying the contents of the multimedia documents, their previews, as well as further information as described in the knowledge graph, showing the triples that form the neighborhood subgraph. There are also UIs for adding files and triples to the knowledge graph, as well as one for segmenting documents with previews.

2.4 External Services

MeGraS integrates external, but locally run, services using gRPC, a high-performance Remote Procedure Calls framework. This approach allows for leveraging the rich ecosystem of available pre-trained models in a different language than the core system. With gRPC, we use an industry-standard interface to connect the two parts of a distributed system reliably and comprehensibly, one being the MeGraS core and the other one computationally intensive and rapidly evolving tasks covered by outsourced services. Currently, we have two of those implemented.

The first is an API to compute the embeddings of texts and images using a Contrastive Language-Image Pre-Training (CLIP) model [4]. Since we use the same model for the two modalities, it allows for comparing the resulting vectors in the same space. This enables multimodal semantic similarity search and to query for images with a text description as well as for nearest neighbors based on visual features. The computed embeddings can be materialized in the knowledge graph, preventing the need to repeat the expensive calculations.

The second service extracts text from images, performing optical character recognition with a pre-trained model. Likewise, this data is added to the graph after having been computed. This allows searching for text found within images, even if this information has not been pre-computed or annotated, and can be extracted on the fly.

3 Applications and Use Cases

MeGraS is aimed at researchers, data engineers, and developers working with large-scale multimodal datasets and knowledge graphs. Hence, we describe applications and use cases for it below.

Enhancing Semantic Search in Multimedia Archives. Several iterations of LifeGraph [6, 7] using MeGraS as a backend have and continue to participate in the Lifelog Search Challenge.[6] Therein, the participants are given a dataset consisting of lifelog events of a first-person view over the span of 18 months, represented with over 700,000 images. The task is then to retrieve these images according to varying criteria from this multimodal dataset. Built for this purpose, MeGraS leverages not only features of the dataset (time, location, visual concepts) but also the visual content of the images. This makes it possible to retrieve images based on query expressions that would not be possible in traditional graph stores. For example, MeGraS natively offers a k-nearest-neighbor operation based on CLIP embeddings [1] of either image to image or text to image. Furthermore, using the semantic expansion of annotations, super- or subclasses can be included in the search.

Analytical Queries Across Disparate Media Types. Even though the LSC use case is multimodal in nature, it only includes media documents of a single type. However, MeGraS's strength lies in its ability to transcend such limitations. It enables complex analytical queries across disparate media types, giving semantics to hitherto undefined or non-existent relations, opening up new avenues for knowledge discovery, proving valuable when dealing with large multimodal datasets such as CASTLE [5]. Unifying the processing of different multimedia types, MeGraS facilitates tasks such as multimodal event search, co-occurrence analysis of objects in precise spatial regions of an image, or retrieving 3D model segments based on textual descriptions from a related document. This holistic understanding is enabled by the deep integration of media types as well as their content.

Listing 1: Querying for the three nearest neighbors of an image based on CLIP embeddings [4].

```
SELECT ?o
WHERE {
  <img> implicit:clip3nn ?o .
}
```

Listing 2: Finding images that fit a textual description based on the cosine similarity of their CLIP embeddings [4].

```
SELECT ?img
WHERE {
  BIND (megras:CLIP_TEXT("description") as ?textVec)
  ?img derived:clipEmbedding ?imgVec .
  FILTER (megras:COSINE_SIM(?textVec, ?imgVec) > 0.8)
}
```

[6] http://lifelogsearch.org/lsc

Example SPARQL Queries. As the focus of the current development is mainly on extending the query capabilities of MeGraS, leveraging SPARQL and Apache Jena, we give some examples inspired by the use cases above. They are shown in Listings 1 and 2.

4 Implementation

MeGraS is implemented as a modular and extensible open-source software system. It is developed using modern and widely adopted technologies for easy integration and to facilitate reproducibility.

4.1 Technology Stack

MeGraS is written in Kotlin and runs on the Java Virtual Machine, combining concise syntax with Java interoperability. It leverages the following main technology stack:

- **Build Tool**: Gradle (Groovy DSL)
- **Web Framework**: Javalin, with OpenAPI and Swagger integration for API documentation
- **CLI**: Clikt for command-line interface support
- **Backend**:
  - PostgreSQL with pgvector and JetBrains Exposed ORM
  - Cottontail DB [2]
- **Remote Services**: gRPC for remote procedure calls and Python (transformers) for local model inference
- **Semantic Processing**: Apache Jena for SPARQL handling

4.2 Availability and Installation

MeGraS is freely available under the MIT License and can be built and run either from source[2] or using a Docker container[3] for ease of deployment and reproducibility. Prebuilt artifacts for MeGraS are automatically created and published through the GitHub CI pipeline. The repository includes installation instructions, configuration examples, and build scripts.

4.3 Documentation

MeGraS provides comprehensive documentation, including:

- **README**: A README file to get started
- **API Documentation**: Automatically generated by OpenAPI annotations and accessible through Swagger UI
- **Command-Line Help**: Built-in support via Clikt for CLI usage and parameter hints
- **User Guides**: Instructions in the getting-started guide

5 Conclusion

In this paper, we introduced **MeGraS**, the **Me**dia**Gra**ph **S**tore, a novel storage and query engine for multimodal knowledge graphs. By unifying the contents and annotations, such as metadata, of multimedia documents, we extend the capabilities for querying them and elevate them to first-class citizens of the graph. MeGraS also allows for the segmentation of multimedia documents according to the Unified Media Segmentation [9], further improving query expressiveness.

MeGraS's query engine is custom-built to handle sophisticated extensions and allow it to reason over content-based features as well. Expanding on SPARQL functionalities, we also account for implicit (computed at run time) and derived (non-materialized) relations, which contribute to enhancing what query expressions can capture.

For future work, we plan on extending the incorporated functionalities, further growing what MeGraS is already capable of, such as including more implicit and derived relations. Optimizations for the query planner are also yet to be explored. Moreover, interaction with the system may be facilitated—especially for non-expert users—by employing a natural-language-to-SPARQL transformer such as NLQxform [8]. Ultimately, MeGraS aims to serve as a foundational tool, fostering new avenues for research and application in the increasingly multimodal landscape of knowledge management and retrieval.

Acknowledgments

This work was partially funded by the Swiss National Science Foundation through Project "MediaGraph" (Grant Number 202125).

References

[1] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. 2022. Reproducible scaling laws for contrastive language-image learning. *CoRR* abs/2212.07143 (2022). doi:10.48550/arXiv.2212.07143 arXiv:2212.07143

[2] Ralph Gasser, Luca Rossetto, Silvan Heller, and Heiko Schuldt. 2020. Cottontail DB: An Open Source Database System for Multimedia Retrieval and Analysis. In *MM '20: The 28th ACM International Conference on Multimedia, Virtual Event / Seattle, WA, USA, October 12-16, 2020*, Chang Wen Chen, Rita Cucchiara, Xian-Sheng Hua, Guo-Jun Qi, Elisa Ricci, Zhengyou Zhang, and Roger Zimmermann (Eds.). ACM, 4465–4468. doi:10.1145/3394171.3414538

[3] Thomas Kurz, Sebastian Schaffert, Kai Schlegel, Florian Stegmaier, and Harald Kosch. 2014. SPARQL-MM - Extending SPARQL to Media Fragments. In *The Semantic Web: ESWC 2014 Satellite Events - ESWC 2014 Satellite Events, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers (Lecture Notes in Computer Science, Vol. 8798)*, Valentina Presutti, Eva Blomqvist, Raphaël Troncy, Harald Sack, Ioannis Papadakis, and Anna Tordai (Eds.). Springer, 236–240. doi:10.1007/978-3-319-11955-7_26

[4] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Proceedings of the 38th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 139)*, Marina Meila and Tong Zhang (Eds.). PMLR, 8748–8763. https://proceedings.mlr.press/v139/radford21a.html

[5] Luca Rossetto, Werner Bailer, Duc-Tien Dang-Nguyen, Graham Healy, Björn Þór Jónsson, Onanong Kongmeesub, Hoang-Bao Le, Stevan Rudinac, Klaus Schöffmann, Florian Spiess, Allie Tran, Minh-Triet Tran, Quang-Linh Tran, and Cathal Gurrin. 2025. The CASTLE 2024 Dataset: Advancing the Art of Multimodal Understanding. *CoRR* abs/2503.17116 (2025). doi:10.48550/ARXIV.2503.17116 arXiv:2503.17116

[6] Luca Rossetto, Oana Inel, Svenja Lange, Florian Ruosch, Ruijie Wang, and Abraham Bernstein. 2023. Multi-Mode Clustering for Graph-Based Lifelog Retrieval. In *Proceedings of the 6th Annual ACM Lifelog Search Challenge, LSC 2023, Thessaloniki, Greece, June 12-15, 2023*. ACM, 36–40. doi:10.1145/3592573.3593102

[7] Luca Rossetto, Athina Kyriakou, Svenja Lange, Florian Ruosch, Ruijie Wang, Kathrin Wardatzky, and Abraham Bernstein. 2024. LifeGraph 4 - Lifelog Retrieval using Multimodal Knowledge Graphs and Vision-Language Models. In *Proceedings of the 7th Annual ACM Workshop on the Lifelog Search Challenge* (Phuket, Thailand) *(LSC '24)*. Association for Computing Machinery, New York, NY, USA, 88–92. doi:10.1145/3643489.3661127

[8] Ruijie Wang, Zhiruo Zhang, Luca Rossetto, Florian Ruosch, and Abraham Bernstein. 2023. NLQxform: A Language Model-based Question to SPARQL Transformer. In *Joint Proceedings of Scholarly QALD 2023 and SemREC 2023 co-located with 22nd International Semantic Web Conference ISWC 2023, Athens, Greece, November 6-10, 2023 (CEUR Workshop Proceedings, Vol. 3592)*, Debayan Banerjee, Ricardo Usbeck, Nandana Mihindukulasooriya, Gunjan Singh, Raghava Mutharaju, and Pavan Kapanipathi (Eds.). CEUR-WS.org. https://ceur-ws.org/Vol-3592/paper2.pdf

[9] Jan Willi, Abraham Bernstein, and Luca Rossetto. 2024. Unified Multimedia Segmentation - A Comprehensive Model for URI-based Media Segment Representation. *TGDK* 2, 3 (2024), 1:1–1:34. doi:10.4230/TGDK.2.3.1