4 AI Service for Module 4 in the Service Life Cycle: On-site processing

QUESTION ANSWERING OVER KNOWLEDGE GRAPHS (KGQA)

Access to knowledge in a company can be very fragmented. The information can be in the form of relational databases, text notes, web tables and so on. A question answering over knowledge graphs (KGQA) system can provide a single point of access to all relevant information stored in an organisation's knowledge graph without requiring technical knowledge (for example, SQL skills). KGQA systems can make it easier and more efficient to find information in large volumes of data.

FOR THE FOLLOWING CHALLENGES

- Finding factual information from the company's knowledge graphs
- Quick access to multiple sources of information (for example, information about customers, deliveries, personnel, parts, and service requests)
- Locating information about previous operations (such as: when was a particular customer machine serviced, with what parts, who performed the repair, when was the repair performed, what problems occurred, etc.).

THE USE CASE

- An employee needs information, for example: "When was the XYZ machine last repaired?"
- They enter this question into the system.
- The system identifies the machine in question in the company's knowledge graph and searches for the relationship corresponding to repairs on a machine.
- It finds five repairs logged for the machine (the relationship in the knowledge graph is "serviced_on") and returns the most recent, by date.
- Similar questions can be asked about any machine from the company's knowledge graph.
- Other information needs can also be answered - for example. "Who performed the repair on machine XYZ?", "What spare parts were used for the repair on machine XYZ?"
- As long as the relevant information is present in the knowledge graph, the system is able to learn how to answer it correctly.

THE SOLUTION IN DETAIL

- For the KGQA system, a trained GNN-based model is used, which is able to provide answers from the knowledge graph to the questions formulated by the user in natural language.
- The system first identifies the entities in the question (based on an entity linkage tool specific to the company's knowledge graph).
- The identified entities are used to identify the relationships to which the question refers.
- For more complex questions, the step of identifying the relationships requires multiple iterations.
- The response predicted by the system is then transmitted to the user.

PROJECT STATUS

The current prototype system answers questions in English about a general knowledge graph (Wikidata)[1]. It achieves a rate of 70% correct answers for simple questions.

REQUIREMENTS

- The system requires an existing knowledge graph to function.
- The system requires an existing entity linking tool capable of recognising entity names specific to the enterprise knowledge graph.
- The system can only answer questions about entities and relationships that are part of a particular knowledge graph.
- The system can only answer factoid questions - that is, questions that have a precise answer in the knowledge graph.
- The system must be trained in advance. As a rule, the more questions it sees, the better its performance. Depending on the size of the knowledge graph, the minimum number of examples can lie between 1,000 and 10,000 questions.

[1] https://www.wikidata.org/wiki/Wikidata:Main_Page

CONTACT PERSON:
University of Stuttgart, Analytic Computing Department

AVAILABILITY

- The code for the prototype system will be made available as an open-source package via GitHub in late 2022.
- The models trained are language and domain specific – for example, a model trained only on the general-purpose knowledge graph will perform worse on a completely different knowledge graph (say, with different relations and entity types).
- Models trained for English are not usable for input in German. These models have to be trained again for the German language.

SPECIFICATION

	Input data	Preprocessing	Data storage	Algorithms	Interfaces
High-level description	For training: questions in natural language + the corresponding answer from the KB For the application: a question in natural language	Identification of entities from the knowledge graph in the question and their associated relationships (requires a suitable entity linker)	SPARQL-based storage for the knowledge graph (e.g. Blazegraph) Space for storing entity indices	GCNs (Graph Convolutional Networks) to find the answer to the question using the background data; SentenceBERT for the representation of questions/ entities	REST-API, question input (JSON), answer and context output (JSON)
Configurability	Connection settings for Knowledge Graph, Entity Linker API	Questions, entities and associated relatationships represented with SentenceBERT	Ideally, the data should be accessible for REST-API enquiries	The hyperparameters of the model (number of GCN layers, size of the layers) can be adapted to best fit the current knowledge graph.	The answer can be converted into a simple text format or other formats.
Technical Implementation	Input from JSON files (Train) or directly from a text field (Use)	REST-API for linking entities REST API for querying the knowledge graph for entities and relationships.	Blazegraph for the knowledge graph, possibly SQLite for other indices	Python, PyTorch, DGL, access to GPUs required for training and use.	Flask API
Specific example from the speedboat project	Company-specific knowledge graph and questions about the knowledge stored therein	User data is preprocessed	User data is saved with suitable storage methods	Question answering over knowledge graphs	Simple interface to enter the question and to list the answers

CONTACT PERSON:
University of Stuttgart, Analytic Computing Department