

# Semi-supervised Construction of Domain-specific Knowledge Graphs

Lavdim Halilaj<sup>1,\*</sup>, Thomas Richardsen<sup>1</sup>, Andreas Dittberner<sup>1</sup>, Frank Wauro<sup>1</sup> and Kim Ngan Nguyen<sup>2</sup>

<sup>1</sup>Robert Bosch GmbH, Germany

<sup>2</sup>Bosch Vietnam Co. Ltd., Vietnam

## Abstract

Current search engines are heavily optimized and excel on retrieving information based on a given set of keywords. The more sophisticated ones are extended to support searches based on the sentences or full text by calculation the similarity of the given query with the already stored entries. However, accessing the domain information using natural language queries is a challenging task. This is due to the myriad variety of "technical" terminology used to describe things. In this paper, we present a framework to automatically construct a knowledge graph based on the given domain specific information. It comprises various mechanisms to extract the relevant entities and their relations from the business cases. Graph- and learning-based methods are incorporated with the aim of improving the information retrieval. This allows users to quickly access desired result even using non-explicit terms or synonyms. Moreover, they are able to discover new links between business cases which may have not been directly encoded.

## Keywords

Informed Decisions, Knowledge Graphs, Similarity Search, Semantic Search, Information Extraction

## 1. Introduction

With the intensive use of digital technologies, the amount of data being generated on a daily basis is increasing at an exponential rate. This data are exploited to support stakeholders for making informed decisions for various domains and applications. The process of making informed decisions starts with collection, distribution, and effectively using information and knowledge that exist within an organization. As a result, individuals and organizations make better decisions, which are crucial for the future success. However, the decision-making process heavily relies on the ability to quickly and efficiently find relevant information. Traditional solutions experience challenging times when dealing with the ambiguity of the terminology as an inherent characteristic occurring in unstructured data. In this context, various automated Information Extraction (IE) techniques are used to analyse textual descriptions and extract useful information from them [1, 2]. Indispensable parts of usual IE pipelines are so-called

---


*TEXT2KG 2023: Second International Workshop on Knowledge Graph Generation from Text, May 28 - Jun 1, 2023, co-located with Extended Semantic Web Conference (ESWC), Hersonissos, Greece*

\*Corresponding author.

✉ lavdim.halilaj@de.bosch.com (L. Halilaj); thomas.richardsen@de.bosch.com (T. Richardsen); andreas.dittberner@de.bosch.com (A. Dittberner); frank.wauro@de.bosch.com (F. Wauro); ngan.nguyenkim@vn.bosch.com (K. N. Nguyen)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

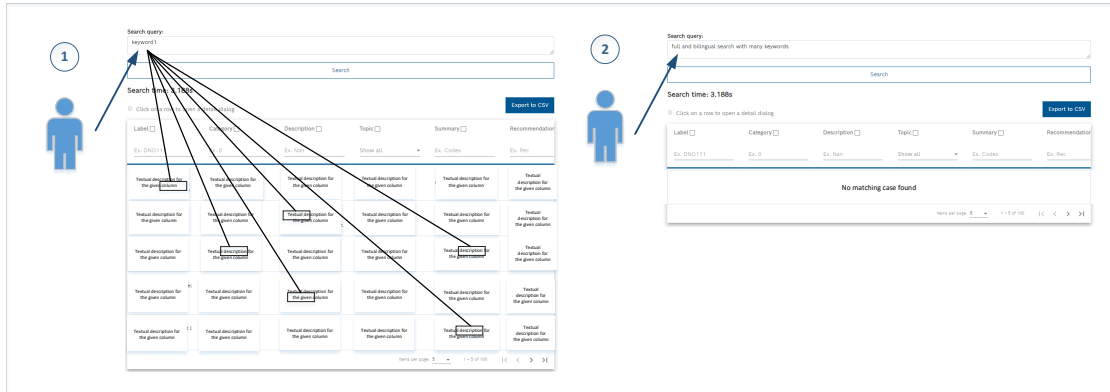
 CEUR Workshop Proceedings (CEUR-WS.org)

dedicated discriminators dealing with specific types of information. Such kind of discriminators are: 1) entity discovery or named entity recognition; 2) entity linking; 3) relation extraction and 4) topic extraction [3].

On the other hand, Knowledge graphs (KGs) are means that can capture complex relationships and interconnections between knowledge elements, e.g., entities, their attributes, and relations. Thus it possible to represent and reason about large and diverse of information in a structured way. This makes them a useful tool to represent the domain knowledge of a field of interest which is understood by practitioners in that field. KGs are used to support semantic search via improving the accuracy and relevance of the information found, which in turn enhances the quality of the decision-making process [2]. KGs can be build manually or (semi)automatically using different techniques including those from natural language processing (NLP) or rule-based methods. However, this task is very complex and heavily relies on human curation and validation of the facts represented in a KG [4].

In this paper, we present a framework to automatically generate a domain-specific knowledge graph with minimal supervision. The backbone of the framework is the ontology which encodes explicit and unique meaning for each term. This is then used to contextualise the unstructured text retrieved from the original sources and enrich them with additional context and definitions. Our focus is on improving the access to the relevant information by: 1) automatically generating the knowledge graph to represent all cases in a domain specific scenario described in free text into unambiguous and machine-readable knowledge; and 2) developing a user-friendly interface providing interactive features for exploration and visual inspection of graph data. Users can perform various searches, like looking for single terms, a combination of the terminology and topics as well as a full text search in conjunction with learned models. Regardless of the synonyms used, the search results include detailed technical descriptions of the business cases, along with an associated list of terms and topics. Further, users are able to navigate and explore related concepts and information in the knowledge graph, which helps them to discover new and relevant information. Overall, the semantic search powered with a knowledge graph provides the following benefits: 1) describing concepts and relations with semantic enriched axioms, allows for a more sophisticated graph exploration and knowledge acquisition; 2) the ability to select categorical values in advance enables performing search on the subset of results and thus reaching faster to the relevant information; and 3) adding new domain terminology and specifying their synonymical connections directly in the graph, circumvent the necessity of learning based methods for huge amount of pre-training data in order to recognize the relations between similar terms.

This paper is structured as follows: A motivating scenario is described in Section 2. A set of identified domain-specific challenges that should be tackled are derived in Section 3. Related work is outlined in Section 4. Section 5 presents our approach including the two main pipelines and their respective components. In Section 6, we provide the concrete implementation details and illustrate the user interface along with the details about the model evaluation. Section 7 concludes the paper and give an outlook of the future directions and extensions of this work.



**Figure 1: Keyword-based search.** The current search platform provides the possibility to find the business cases using keywords only. First image illustrates the scenario where the user enters a keyword and in return receives a number of matching results. The user has then to manually look for the relevant cases. In the second scenario, the user provides a longer text, which results on no found cases.

## 2. Motivation Scenario

For any product or service that is delivered to the internal divisions or customers, there can be a number of issues that can happen. In daily basis, employees have to deal with new business cases which are reported by various divisions or customers. These business cases cover issues related to the safety, emissions, electronics, etc. Typically, the first step performed after receiving the information of a new case is searching for similar ones that have been reported before. For each issue, a deep investigation of the causes and related factors should be done as well as the affected components or services. In addition, the expert should provide recommendations for solving that and measures needed to be taken into account. The ability to find such cases is crucial for preventing the duplication of the work with respect to the investigations that should be realized. Further, it also avoids the risk of missing important aspects to be considered for the similar cases, which might cause achieving divergent conclusions at the end of the workflow.

However, the process of searching is tedious, time-consuming and error prone. This is due to a number of factors related to the case representation and retrieval. While describing a given business case, employees provide a diverse terminology, i.e. some use *Term1* and others use *Term2*, both pointing to the same functionality or category. This is further exacerbated due to the fact that more than one language is usually used to describe various aspects of given cases, i.e. a combination of English with the local language. As a consequence, searching based on keywords only embodies the risk of missing similar cases and thus increasing efforts on repeating the same analysis again as well as providing different conclusions.

## 3. Domain-Specific Challenges

Data from domain-specific sources poses a number of significant linguistic challenges w.r.t. diverse terminology and the noisy phrases that need to be addressed. In the following, we emphasize some of the challenges which are crucial for an easy information access and retrieval.

**CH1 - Heterogeneity:** the structure of the raw data is typically very diverse. This hinders the ability to exchange and mutually use of information across systems or applications.

**CH2 - Multilinguality:** capturing information about the business cases is usually realized using a combination between different languages. Therefore, it is of paramount of importance to be able to handle enquires in multiple languages.

**CH3 - Ambiguity:** user queries based on the natural language can be ambiguous and involve multiple keywords. Thus, it is crucial to disambiguate mentioned entities within query in order to return the most relevant results.

**CH4 - Multimodality:** handling structured data, such as data stored in a knowledge graph, in addition to the textual descriptions, is crucial for accessing the information via sophisticated query mechanisms.

**CH5 - Evolution:** domain knowledge can change over the time because of the dynamics, or it is refined by experts. Therefore, the ability to reflect such changes quickly in the search results is very important.

## 4. Related Work

The automatic construction of domain-specific KGs has been subject of investigation for years in many domains, e.g. biomedical domain [5, 6, 7], art and culture [8, 9], and academic literatures and algorithms [10].

Some of these approaches leverage generic ontologies such as DBpedia and WikiArt or domain-specific ontologies to model relevant entities and their relations. An ontology-free approach to automatically construct a knowledge graph for art-historic documents is presented in [11]. The approach is ontology-agnostic and use open information extraction techniques to retrieve triples for the sentences. HDSKG [12] is an automatic approach that leverages the content of webpages for discovering domain specific concepts and their relations. It incorporates a rule-based dependency parser in combination with machine learning algorithms to find the candidate relations and estimate their domain relevance. To facilitate the process of information extraction for the public KGs, authors in [13] proposed PLUMBER, a framework comprising different reusable components. These components can perform various tasks, such as entity recognition and linking, relation extraction and coreference resolution and can be combined in up to 264 distinct pipelines. A framework for building a semantic knowledge bases to support advanced information analysis and intelligent inference for a specific problem domain is presented in [14]. It includes components and techniques to enable users developing customized and reusable pipelines that can be applied to different domains. A comprehensive survey of techniques that utilizes ontologies for information extraction tasks is provided in [3]. Authors group the majority of those techniques into two main categories, namely: *rule-based* - involving experts defining patterns for denoting tokens, their interrelations and semantics; and *learning-based* - which can be *supervised*, where a large annotated text corpora is used to build models; *unsupervised*, where clustering techniques are applied to find statistical similarities based on the tokens co-occurrences; and *semi-supervised*, where seed examples previously learned from a small annotated model are used to recursively learn new patterns.

In addition, there exist other approaches aiming to automate the KG construction using

declarative rules. These rules are specified via mapping languages such as R2RML<sup>1</sup> or RML [15] as well as query languages like SPARQL<sup>2</sup> and SPIN<sup>3</sup>. For instance, AutoMap4OBDA [16] and BootOX [17] receive an input ontology and generate actual mappings in R2RML language for a given relational database. Other solutions [18, 19] address the problem by employing additional methods, such as heuristic rules, fuzzy search over the KGs or knowledge graph embeddings to match the structure of tabular datasets to the given ontological concepts. Authors in [20] investigate trends of automatic KG construction declarative mapping languages. In addition, they discuss the challenges related to the maintainability, reproducibility explainability of KG construction. Particularly, they elaborate shortcomings present into the declarative approaches due to the fact they follow a *linear generation process* compared to the *iterative process* followed by the traditional approaches based on IE techniques.

## 5. Approach

With the aim of improving the accuracy of the searching process and the relevance of retrieved results, we designed a loosely-coupled framework. It allows incorporation of prior knowledge formalized in an ontology, which later is utilized to guide the information extraction and the searching process, respectively. The architecture, as shown in Figure 2, comprises two main pipelines, namely: 1) *Ingestion Pipeline*; and 2) *Consumption Pipeline*.

### 5.1. Ingestion Pipeline

This essential pipeline enables automation of the process of bringing "raw" data into useful insights and knowledge using semantic concepts. It consists of three main phases executed in sequence: 1) *Access and Preprocessing*; 2) *Knowledge Extraction*; and 3) *Integration and Enrichment*. Each phase comprises a number of components dedicated to perform specific tasks.

#### 5.1.1. Access and Preprocessing

The main task of this phase is to obtain and prepare data for subsequent analysis and processing. Next, the input data are transformed to a certain structure while avoiding parts that could impair the accuracy and relevance of the final results, thus addressing the challenge *CH1*.

**Case Retrieval** The first step is related with accessing and retrieving information from the given set of sources such as databases, or APIs. Moreover, the criteria for selecting relevant data should be defined, along with the access permission for each data source.

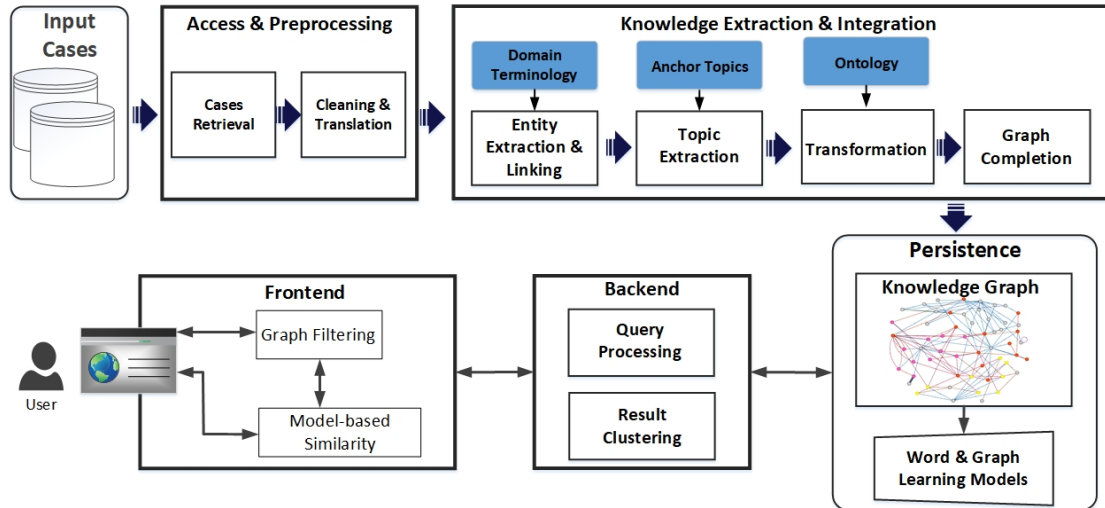
**Cleaning and Translation** Here a number of tasks related to cleaning, transforming, and formatting are performed. It may involve removing of duplicates, changing or aligning data types, as well as normalizing data, important for handling the challenge *CH1*. As the quality of the obtained data is crucial for the end results, this step is important towards ensuring the

---

<sup>1</sup><https://www.w3.org/TR/r2rml/>

<sup>2</sup><https://www.w3.org/TR/sparql11-overview/>

<sup>3</sup><https://spinrdf.org/>



**Figure 2: The Framework.** The proposed framework contains two main pipelines. The upper pipeline is comprised of three consecutive phases: 1) *Access and Preprocessing* - retrieve the data from given sources and perform necessary tasks such as cleaning up and translation of cases into a specified language; 2) *Knowledge Extraction* - extract additional knowledge using prior information as input; and 3) *Ingestion and Enrichment* - transform, enrich and ingest information in the KG. On the other hand, the user interaction is managed by separate components, i.e. *Frontend* and *Backend*, responsible for accommodating user queries and showing the results in various forms.

suitability of the data for the intended purpose. Considering the *CH2*, information is translated to one single language, i.e. *English*, while preserving the original text into respective fields. In addition, a number of predefined stop-words are removed to reduce the noise in the text with the aim of enhancing the accuracy of the learning models.

### 5.1.2. Knowledge Extraction

Automatically extracting information from various unstructured or semi-structured sources, such as text corpus, images, or audio files is an important step. In regard with challenge *CH4*, it should be possible to identify relevant entities and relationships from a given corpus of text. This information can later on be represented in a knowledge graph encapsulating the underlying structure of the input data.

**The ontology** We built the Business Cases Ontology (BCO) to formally represent the domain specific information in form of classes, attributes and relationships. It captures essential details about the business cases via the main concepts: a) *Business Case* - encapsulating metadata about the business cases. We created several subclasses to further specify the nature of a particular case, such as *Safety*, *Exhaust Emission* and *Charging*; b) *Function* - denoting the type of function mentioned in the given case; and c) *Requirement* - describing the requirements relevant to the case. To enable interlinking with other data sources several internal and external ontologies such as Schema.org and DBpedia<sup>4</sup> are reused. The BCO is developed in an iterative manner by

<sup>4</sup><https://schema.org>; <http://dbpedia.org/ontology#>



involving technical experts on phases for requirements collection and domain conceptualization. Best practices for collaborative development [21] and guidelines [22] such as quality assurance, role definition, version labeling and naming conventions are utilized. Currently, BCO contains 29 classes, 15 object, 8 datatype, and 6 annotation properties, respectively.

**Entity Linking** Recognising entities mentioned in each business case is essential for enabling domain experts to quickly reach the information they need. Therefore, this component deals with *Entity Linking*, including both recognizing and disambiguation of entities within the given text. Detecting patterns and the context in textual descriptions can be achieved via a number of techniques such as part-of-speech tagging, dependency parsing, or machine learning algorithms. Moreover, in a domain-specific scenario, a dictionary of the entities can be provided as input in advance. This dictionary is then used for examining the text w.r.t. mentioned entities. For example, for each function " $f_n$ " with the ontology identifier  $bco:Function_n$  present in a business case " $bc_m$ ", we generate a link  $\{bc_m, bco:hasFunction, bco:Function_n\}$ .

Further, the dictionary also may include the prior relations about synonyms, thus enabling the search engine to "understand" terms with similar meanings which are interchangeable in certain contexts. As a result, users can search for information using different but related terms, without having to explicitly specify each possible term in their search query.

**Topic Extraction** The textual description of business cases can in principle comprise a mixture of different topics  $T = t_1, t_2, \dots, t_n$  where each topic is composed by number of words co-occurring together [3]. This component is responsible for automatically extracting topics by identifying the underlying structures and hidden patterns. As a result, co-occurring words and phrases grouped into topics are used to later-on enrich the knowledge graph. Methods such as pattern matching, clustering, and topic modeling can be utilized to extract the most relevant words and phrases for each topic. Those methods work based on different approaches: 1) *unsupervised*, typically generative models extracting the topics solely on correlation of words; and 2) *(semi)supervised* with minimal human intervention, where a set of anchor topics and their respective words are given as input. Here, we use a semi-unsupervised approach which takes as input topics defined in the BCO ontology and the produces correlations of the textual descriptions with those topics. As a result, each business case is linked to one or multiple topics, allowing users to explore the knowledge and group the results based on different criteria.

### 5.1.3. Integration and Enrichment

This component is responsible for transformation of the results generated from the previous steps into a knowledge graph. The objective is to convert received data into actionable insights that later are used to filter and refine the relevance of the results and facilitate an informed decision-making process. Representing the original data into a knowledge graph is realized according the BCO ontology.

The extracted entities and topics are then linked to the concepts in the BCO ontology and enhanced with semantic definitions of concepts and their relationships. Finally, various techniques such as inference based on rules or link prediction with machine learning can be used to complete the graph and make explicit the implicit information.

## 5.2. Persistence

The persistence component provides necessary mechanisms to store and manage with the information represented in a knowledge graph over time as specified in the challenge *CH5*. A KG represents a set of facts in form of triples,  $KG = H, R, T$ , where  $H$  denotes entities,  $T \subseteq E \times L$  entities or literal values and  $R$ , a set of relation between elements in  $H$  and  $T$ . It also provides accessing interfaces for later usage as well as the ability to perform more advanced operations related to querying, indexing, and transaction execution. In addition, this component has dedicated modules for learning latent representations and capturing the syntactic and semantic properties. Various information modalities such as graph or text can be retrieved from the storage and mapped to a high-dimensional vector space, where similar information are embedded closely. Next, trained models may be used to provide the results based on calculating the similarity between the given input and the learned vector representation.

## 5.3. Consumption Pipeline

After the raw data is fully transformed into a knowledge graph according the semantic concepts, they are ready for further usage. This pipeline comprises a number of modules performing various tasks to enable an efficient and effective information consumption, i.e. allowing users to quickly retrieve relevant results.

### 5.3.1. Backend - Application Logic

The backend works in conjunction with the frontend and the persistence component. It handles user requests towards the knowledge graph as well as to the machine learning models. In turn, after the queries are executed, this component offers results for further processing.

**Result Clustering** Techniques such as graph-based or learning-based can be combined to refine the search space and avoid irrelevant content. The results from this combination of search techniques should be synthesised to best match business cases. Depending on the relevance, various methods are available to rank results retrieved from the given input. The rankings of these individual methods can be summarized into an overall ranking and thus (according to the ensemble principle) synergies from the specific advantages of the individual methods can be maximized. The arithmetic, median or the harmonic mean value (or even more sophisticated strategies) can be used as classification criteria for this superordinate ranking.

### 5.3.2. Frontend - User Interaction

Allowing users to interact with the system can be realized in many forms, including submitted queries, feedback about the relevance of search results, as well as other kinds of graph exploration and filtering features.

One of the main objectives is to enable users to quickly and effectively find the relevant information they are looking for. Therefore, adequate mechanisms for browsing and filtering are essential to navigate more efficiently over large amounts of information, thus helping to



priority narrow down the search space using various categorical or numeric criteria. Further, users are able to easily incorporate their provide feedback according the search results.

The search results will contain detailed information of use cases along with a list of the associated semantic concepts and their description. It enable users to navigate and explore in a graph-based representation and having a quick overview of similar cases.

## 6. Implementation

We implemented our solution based on the architecture described in the previous section. In the following, we give technical details about each component.

### 6.1. Access and Preprocessing

Accessing the given datasource is realized using the following libraries: `requests==2.28.1` and `requests-toolbelt==0.10.0`. The relevant columns to be selected from the given source are pre-defined in advance. Original data are retrieved in CSV format via a process scheduled to run once per week. The preprocessing step is performed using `pyparsing==3.0.7` and includes data cleaning tasks like deleting special characters and expressions, fixing white spaces and punctuation. Further, as there might be a mixture of English, German and Japanese languages, data are translated to English via specific models, e.g. BERT and libraries such as `transformers==4.21.3`, `sentence-transformers==2.2.2` and `sentencepiece==0.1.96`.

### 6.2. Knowledge Extraction

The *Entity Extraction* component is implemented based on the *spaCy*<sup>5</sup> framework v.2.3.5 using the rule-based extraction feature. The rule-based matcher engine allows for finding the words and phrases in form of tokens within present business cases corpus. As input, we received a dictionary with 270 unique functions and their respective descriptions. Next, a link for each identified function in our scenario and the given business case is established.

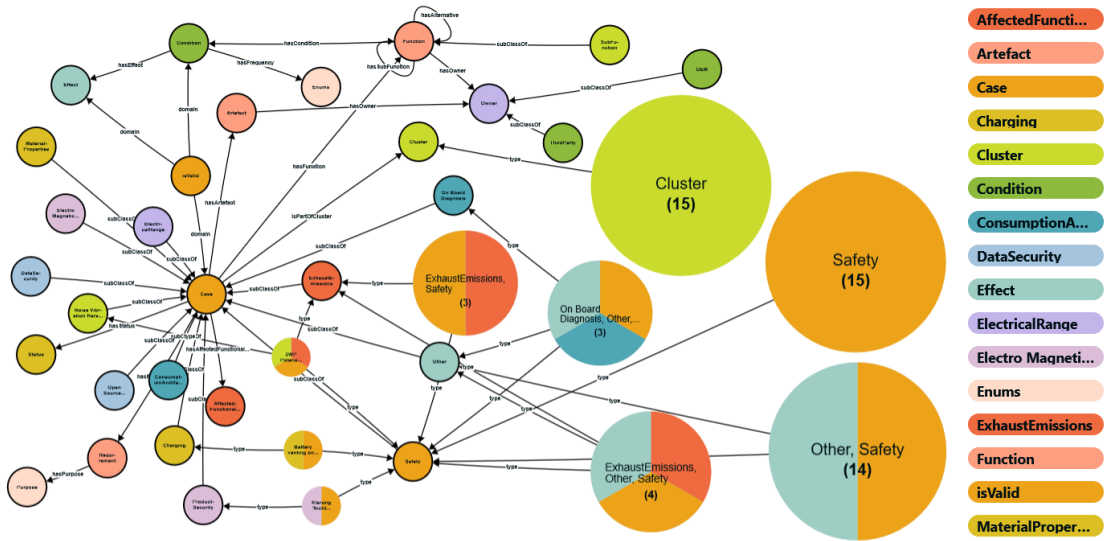
The *Topic Extraction* component is based on a semi-supervised method, namely Correlation Explanation (CorEx) [23]. From domain experts, we received a list with 15 topics, each associated with different number of words, ranging from a minimum of 3 to a maximum of 30. These predefined topics are then incorporated as input to CorEx in the form of *anchor* words. After the method execution, a total of 3,227 topics are identified for all business cases, with an average of 3 topics per case.

### 6.3. Business Case Knowledge Graph

Data retrieved from the data source is structured in the CSV format. Using the defined pipeline this data is then enriched based on the domain ontology. As a result, the Business Case Knowledge Graph (BCKG) is generated and contains over 22 thousand triples. There are more than 1000 business cases belonging to one or multiple categories. The BCKG is stored in Stardog

---

<sup>5</sup><https://spacy.io/>

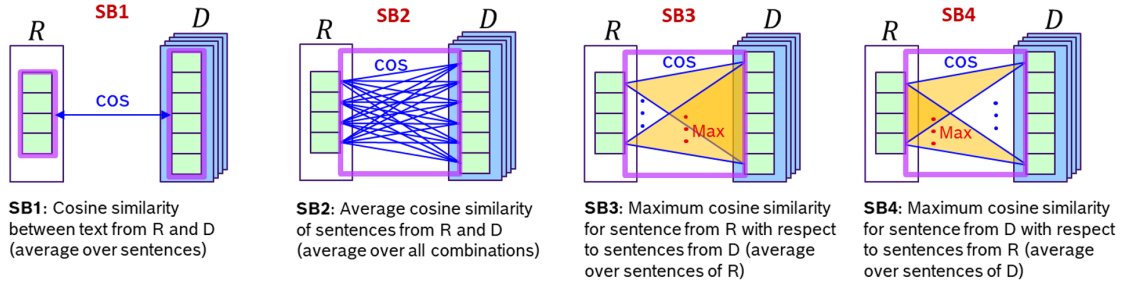


**Figure 3: Business Cases Knowledge Graph (BCKG).** An excerpt of BCKG depicting classes, their taxonomic- and inter- relations. The encoded business cases are of different types, such as *Safety*, *Board Diagnosis* and *Exhaust Emissions*.

v.7.9.2. In addition, several built-in functions of Stardog for performing basic string similarity such as *cosine* or *levenshtein* distance, respectively, are used.

#### 6.4. Learning Component

The knowledge graph-based search is supported by further parallel search methods based only on the textual entries such as TF-IDF and S-BERT [24]. In this scenario, knowledge graph acts as a refining mechanism to narrow down the search space. The retrieved subset of results is then used as input to the learning-based methods for further comparison, i.e. TF-IDF and S-BERT. TF-IDF is a plain bag-of-words method, which exploits the occurrence and the relative frequency of the terms from the search entry. For this reason, it is able to deal with respective domain-specific terminology, but has the disadvantage that different synonyms in search text and business cases text cannot be related. In order to tackle this, we retrieve the synonyms from the BCKG, so the user has the possibility to incorporate into the search query. The second assisting method, the text-based similarity comparison via S-BERT, is characterized by the fact that the meaning of sentences or text sections is encoded via embedding vectors, where the similarity relationship between synonyms can be recognized, and ambiguities in the meaning can be resolved from the textual context. The content of texts is not reduced to the presence of terms, but is derived from the word combination under consideration of the structure and grammar of sentences. However, this capability of similarity decoding is reduced to the plain word comparison or comparison of tokens if the used terms are not represented in the underlying language model. The S-BERT similarity comparison is integrated into an overarching aggregation method, that converts the similarity relationships between individual sentences into similarity relationships between larger text sections.



**Figure 4: Aggregation methods.** Aggregation similarity score is based on cosine of embedding vectors between subunits of the reference search text  $R$  and the base set  $D$ . Individual scores for unit pairings are aggregated to overall score.

### 6.4.1. Aggregation Methods

Different methods are developed to derive indicators for the quality of the obtained search results. For this aim, two main sets  $X$  are defined:

- the base set  $D$  - represents any business case from database comprising the text from the different attributes (e.g. *case description*, *technical description*, *case conclusion*, etc.) and
- the reference set  $R$  - represents one specific business case, and contains a textual summary of this case, which is used as search entry (e.g. the separately stored attribute *technical assessment* is used for this purpose).

The expectation is that a business case in  $D$  related to the chosen summary from  $R$  should be located in the top ranks (i.e. with low position index) of the search results. Further, for topic clusters of existing business cases, their individual text descriptions are used as search input. Here, we investigate to what extent other business cases of the same cluster are represented with low position index in the search results. Based on these indicators, the subsequently introduced aggregation methods are compared to decide for the most suitable ones.

#### Definition:

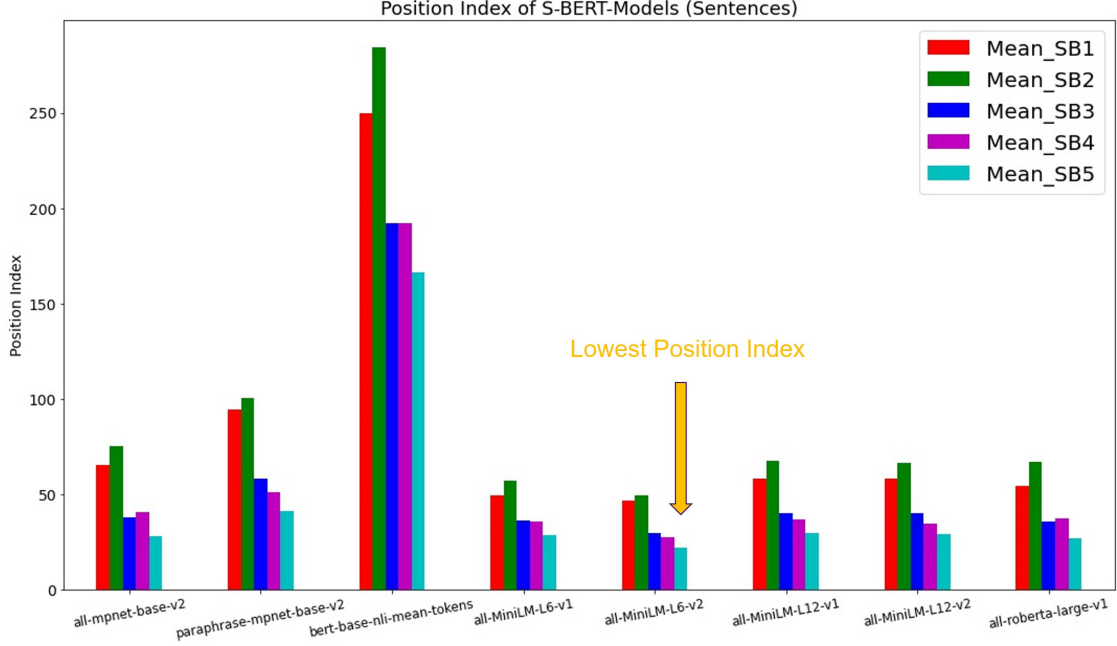
The set  $X$  (representing either  $D$  or  $R$ ) is a set of sentences a.k.a. text units of the respective text parts (see above) of the related business case. The number of text units in the set  $X$  is denoted by  $|X|$ , whereas  $v_i^{(X)}$  is the embedding vector of the  $i$ -th text unit. The  $l_i$  is the corresponding unit lengths in words and  $\cos(v_i, v_j)$  is the cosine of the intermediate angle between two vectors  $v_i$  and  $v_j$ . Figure 4 illustrates each aggregation method defined as follows:

- **SB1:** Individual cosine similarity of average embedding vectors (averaged separately over all sentences of  $R$  respectively  $D$ ):

$$SB1(R, D) = \cos(v^{(R)}, v^{(D)}); \quad v^{(X)} = \frac{1}{|X|} \sum_{i \in X} v_i^{(X)}, \quad X \in R, D \quad (1)$$

- **SB2:** Average cosine similarity of all pairs of individual sentences from sets  $R$  and  $D$ :

$$SB2(R, D) = \frac{\sum_{i \in R} l_i \sum_{j \in D} (l_j \cos(v_i^{(R)}, v_j^{(D)}))}{\sum_{i \in R} \sum_{j \in D} l_i l_j} \quad (2)$$



**Figure 5: Model Selection based on Position Index.** Position index of the database cases to be searched on average over a larger number of cases for different S-BERT models (all-mpnet-base-v2, paraphrase-mpnet-base-v2, bert-base-nli-mean-tokens, all-MiniLM-L6-v1, all-MiniLM-L6-v2, all-MiniLM-L12-v1, all-MiniLM-L12-v2 and all-roberta-large-v1) and different aggregation methods. The lower the position index, the more reliably the detection of the database cases related to the search texts.

- **SB3:** Maximum cosine similarity for sentence from  $R$  with respect to sentences from  $D$  (average over sentences of  $R$ ):

$$SB3(R, D) = \sum_{i \in R} l_i \max_{j \in D} (l_j \cos(v_i^{(R)}, v_j^{(D)})) / l_{j_{Max}} \sum_{i \in R} l_i \quad (3)$$

- **SB4:** Maximum cosine similarity for sentence from  $D$  with respect to sentences from  $R$  (average over sentences of  $D$ ):

$$SB4(R, D) = \sum_{i \in D} l_i \max_{j \in R} (l_j \cos(v_i^{(D)}, v_j^{(R)})) / l_{j_{Max}} \sum_{i \in D} l_i \quad (4)$$

- **SB5:** Arithmetic average of similarity between SB3 and SB4:

$$SB5(R, D) = (SB3(R, D) + SB4(R, D)) / 2 \quad (5)$$

#### 6.4.2. Model Selection

Apart from the aggregation method also the selected S-BERT model has a significant impact on the quality of the obtained results. Hence different pretrained S-BERT models should be investigated based on these indicators. The preselection (see Figure 5) of these models is based on the performance scores for sentence embeddings and performance semantic search on <sup>6</sup>.

<sup>6</sup>[https://www.sbert.net/docs/pretrained\\_models.html](https://www.sbert.net/docs/pretrained_models.html)

For the comparison, a selection of reference texts in  $R$  is used as search input and the received cases from  $D$  are sorted according to their respective similarity. The obtained position indexes correspond to the ranking of the business cases in  $D$  related to the respective reference text  $R$  within the search results (averaged over all selected reference texts  $R$ ). This procedure is repeated for all preselected S-BERT models jointly with the aggregation methods described as above. As can be seen from Figure 5, the lowest position indexes result for the model (all-MiniLM-L6-v2) together with the aggregation method SB5.

The evaluation of the quality indexes also revealed the fact, that especially for very short search text entries (consisting of only one sentence) the aggregation method SB3 can perform even better than method SB5. Additionally, it should be mentioned here, that intended search results may be based on more abstract aspects which are often very difficult to implement with a plain text-based similarity comparison. More advanced investigations to improve the search results in this respect are ongoing.

#### 6.4.3. Backend

The backend is implemented using python v.3.9 and various libraries dedicated for interacting with the knowledge graph as well as handling the application logic and processing. Communicating with the triple store is realized using SPARQLWrapper v.2.0. On the other hand the application logic is implemented using rdflib v.5.0 and pandas v.1.2.5.

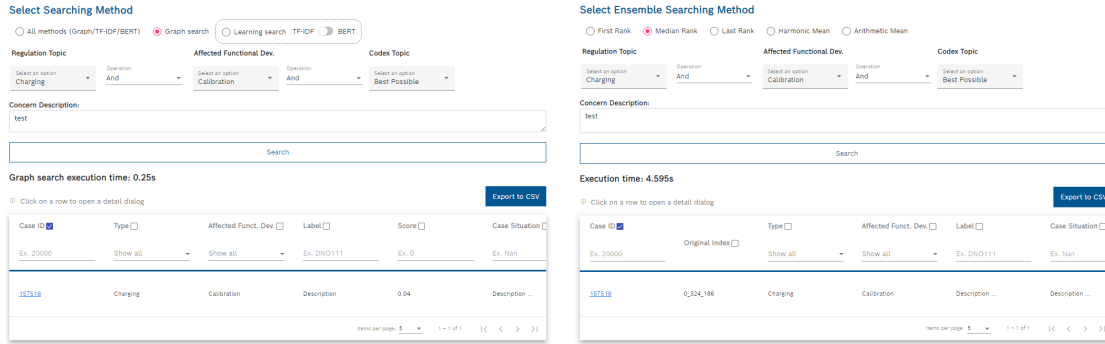
#### 6.4.4. Frontend

The frontend is implemented using Angular 9.1.13, vis-data 7.1.4 and vis-network: 9.1.2. It offers various user-friendly forms for interaction and displaying the results. The user can select between different searching methods in combination with various criteria and the operators between them i.e. *OR*, *AND*. These searching methods depicted in Figure 6a are as follows:

**Graph-based** - use the selected categories, the operators, and the given text. These are then posted to the knowledge graph platform in form of queries with the selected values and the text to calculate the similarity with the other cases. A further advanced feature enables leveraging the subclasses defined in the ontology in combination with Boolean operators, thereby merging categories and tokens with *OR* or *NOT* to allow alternatives or exclude certain terms. One particular extension is including synonyms or special categories of business cases.

**Learning-based** - first, the categories from the knowledge graph are used to refine or narrow down the search space. Then the subset of results is sent to the chosen method either TF-IDF or a BERT variant as described above, where the similarity calculation is performed.

Moreover, users may post simultaneous queries against all searching methods, i.e. graph, TF-IDF, or Bert. Therefore, as shown in Figure 6b, we implemented five different techniques to allow ranking of the retrieved results: 1) First Rank; 2) Median Rank; 3) Last Rank; 4) Harmonic Mean; and 5) Arithmetic Mean. The results coming from the selected methods are finally ordered according to their respective similarity score and the selected technique.



(a) Various searching methods

(b) Various ensemble methods

**Figure 6: User interface.** Dedicated forms providing sophisticated mechanisms for retrieving relevant results: a) different search methods; b) various means for ranking can be chosen.

## 7. Conclusions

This article presents a framework to automatically construct a KG for the given industrial domain. The input data are provided in textual descriptions for all business cases. The approach comprises a number of components to support information extraction, like entities and topics. Next, the extracted entities are linked with the specific concepts of the developed ontology. At the end of the pipeline, information is transformed into a knowledge graph. Further, in order to support users for accessing relevant information, we build a specific application. It contains two main components, namely backend, and frontend. Users are able to easily use various methods for searching and reaching to the relevant results. They can combine different criteria to restrict the search scope against the knowledge graph. Then, on the subset of the returned results, it is possible to apply different methods like cosine similarity, TF-IDF, or BERT. Further, it is also possible to leverage the synonyms which are manually defined by domain experts. In summary, using a knowledge graph to power semantic search provides several advantages, such as: 1) semantic enrichment of concepts and relationships enhances knowledge exploration and acquisition; 2) pre-selection of categorical values enables more efficient search and quickly access relevant information; and 3) incorporating priory new domain terminology along with their synonyms helps on tackling the need for vast amounts of pre-training data to recognize relationships between similar terms.

The ambiguity and impreciseness present in the natural language text make the automatic construction of a KG a very challenging task. Therefore, our future direction is to exploit the power of the large language models for extracting the pieces of information in form of subgraphs before integrating them into the domain knowledge graph. Further, we plan to further extend our solution with another type of similarity based on graph topology and structure. As currently the source dataset is rather small, the actual ingestion pipeline pulls once per week all data and convert into a knowledge graph. However, as the size of the dataset is expected to grow over time, we plan to work on techniques that allows retrieving and processing of the deltas only. We also plan to use event-based mechanisms that would allow for putting data into the KG as they arrive or after each change. As result, the KG will always reflect the latest state of the business cases and any synchronization issue will be avoided.

## References

- [1] R. C. Bunescu, R. J. Mooney, A. K. Ramani, E. M. Marcotte, Integrating co-occurrence statistics with information extraction for robust retrieval of protein interactions from medline, in: Proceedings of the Workshop on Linking Natural Language and Biology, BioNLP@NAACL-HLT, Association for Computational Linguistics, 2006, pp. 49–56.
- [2] A. Aljamel, T. Osman, D. Thakker, A semantic knowledge-based framework for information extraction and exploration, International Journal of Decision Support System Technology (IJDSST) 13 (2021) 85–109.
- [3] J. Martínez-Rodríguez, A. Hogan, I. López-Arévalo, Information extraction meets the semantic web: A survey, Semantic Web 11 (2020) 255–335.
- [4] H. Ye, N. Zhang, H. Chen, H. Chen, Generative knowledge graph construction: A review, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP, Association for Computational Linguistics, 2022, pp. 1–17.
- [5] H.-M. Müller, E. E. Kenny, P. W. Sternberg, Textpresso: An ontology-based information retrieval and extraction system for biological literature, PLOS Biology 2 (2004).
- [6] J. Yuan, Z. Jin, H. Guo, H. Jin, X. Zhang, T. H. Smith, J. Luo, Constructing biomedical domain-specific knowledge graph with minimum supervision, Knowl. Inf. Syst. 62 (2020) 317–336.
- [7] P. Ernst, A. Siu, G. Weikum, Knowlife: a versatile approach for constructing a large knowledge graph for biomedical sciences, BMC Bioinform. 16 (2015) 157:1–157:13.
- [8] H. Wu, S. Y. Liu, W. Zheng, Y. Yang, H. Gao, Paintkg: the painting knowledge graph using bilstm-crf, in: 2020 International Conference on Information Science and Education (ICISE-IE), 2020, pp. 412–417. doi:10.1109/ICISE51755.2020.00094.
- [9] J. Hunter, S. Odat, Building a semantic knowledge-base for painting conservators, in: IEEE 7th International Conference on E-Science, e-Science, IEEE Computer Society, 2011, pp. 173–180.
- [10] J. Patel, B. Dutta, An approach for automatic construction of an algorithmic knowledge graph from textual resources, in: Proceedings of the 1st International Workshop on Knowledge Graph Generation From Text and the 1st International Workshop on Modular Knowledge co-located with 19th Extended Semantic Conference (ESWC), volume 3184 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2022, pp. 109–122.
- [11] N. Jain, A. S. Múnera, M. Lomaeva, J. Streit, S. Thormeyer, P. Schmidt, R. Krestel, Generating domain-specific knowledge graphs: Challenges with open information extraction, in: Proceedings of the 1st International Workshop on Knowledge Graph Generation From Text co-located with 19th Extended Semantic Conference (ESWC), volume 3184 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2022, pp. 52–69.
- [12] X. Zhao, Z. Xing, M. A. Kabir, N. Sawada, J. Li, S. Lin, HDSKG: harvesting domain specific knowledge graph from content of webpages, in: IEEE 24th International Conference on Software Analysis, Evolution and Reengineering, SANER, 2017, pp. 56–67.
- [13] M. Y. Jaradeh, K. Singh, M. Stocker, A. Both, S. Auer, Better call the plumber: Orchestrating dynamic information extraction pipelines, in: Web Engineering - 21st International Conference, ICWE Proceedings, volume 12706 of *Lecture Notes in Computer Science*, Springer, 2021, pp. 240–254.



- [14] A. Aljamel, A knowledge-based framework for information extraction and exploration, Ph.D. thesis, Nottingham Trent University, UK, 2018.
- [15] A. Dimou, M. V. Sande, P. Colpaert, R. Verborgh, E. Mannens, R. V. de Walle, RML: A generic language for integrated RDF mappings of heterogeneous data, in: Proceedings of the Workshop on Linked Data on the Web co-located with the 23rd International World Wide Web Conference (WWW), CEUR Workshop Proceedings, 2014.
- [16] Á. Sicilia, G. Nemirovski, Automap4obda: Automated generation of R2RML mappings for OBDA, in: Knowledge Engineering and Knowledge Management - 20th International Conference, EKAW Proceedings, volume 10024 of *Lecture Notes in Computer Science*, 2016, pp. 577–592.
- [17] E. Jiménez-Ruiz, E. Kharlamov, D. Zheleznyakov, I. Horrocks, C. Pinkel, M. G. Skjæveland, E. Thorstensen, J. Mora, Bootox: Bootstrapping OWL 2 ontologies and R2RML mappings from relational databases, in: Proceedings of the ISWC Posters & Demonstrations Track co-located with the 14th International Semantic Web Conference (ISWC), volume 1486 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2015.
- [18] P. Nguyen, I. Yamada, N. Kertkeidkachorn, R. Ichise, H. Takeda, Semtab 2021: Tabular data annotation with mtab tool, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching co-located with the 20th International Semantic Web Conference (ISWC), CEUR Workshop Proceedings, CEUR-WS.org, 2021, pp. 92–101.
- [19] V. Huynh, J. Liu, Y. Chabot, F. Deuzé, T. Labbé, P. Monnin, R. Troncy, DAGOBAN: table and graph contexts for efficient semantic annotation of tabular data, in: Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching co-located with the 20th International Semantic Web Conference (ISWC), CEUR Workshop Proceedings, 2021, pp. 19–31.
- [20] D. Chaves-Fraga, A. Dimou, Declarative description of knowledge graphs construction automation: Status & challenges, in: Proceedings of the 3rd International Workshop on Knowledge Graph Construction (KGCW) co-located with 19th Extended Semantic Web Conference (ESWC), CEUR Workshop Proceedings, 2022.
- [21] L. Halilaj, I. Grangel-González, G. Coskun, S. Lohmann, S. Auer, Git4voc: Collaborative vocabulary development based on git, *International Journal on Semantic Computing* 10 (2016) 167–192.
- [22] I. Grangel-González, L. Halilaj, G. Coskun, S. Auer, Towards vocabulary development by convention, in: KEOD - Proceedings of the International Conference on Knowledge Engineering and Ontology Development, part of the 7th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K), Volume 2, SciTePress, 2015, pp. 334–343.
- [23] R. J. Gallagher, K. Reing, D. C. Kale, G. V. Steeg, Anchored correlation explanation: Topic modeling with minimal domain knowledge, *Trans. Assoc. Comput. Linguistics* 5 (2017) 529–542.
- [24] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP, Association for Computational Linguistics, 2019, pp. 3980–3990.