Gathering and Combining Semantic Concepts from Multiple Knowledge Bases

Alexander Paulus, André Pomp, Lucian Poth, Johannes Lipp and Tobias Meisen
Institute of Information Management in Mechanical Engineering, RWTH Aachen University, Aachen, Germany

Keywords: Semantic Computing, Semantic Model, Knowledge Graph, Ontologies, Semantic Similarity.

Abstract: In the context of the Industrial Internet of Things, annotating data sets with semantic models enables the automatic interpretability and processing of data values and their context. However, finding meaningful semantic concepts for data attributes cannot be done fully automated as background information, such as expert knowledge, is often required. In this paper, we propose a novel modular recommendation framework for semantic concepts. To identify the best fitting concepts for a given set of labels, our approach queries, weights and aggregates the results of arbitrary pluggable knowledge bases. The framework design is based on an intensive review of labels that were used in real-world data sets. We evaluate our current approach regarding correctness and speed as well as stating the problems we found.

1 INTRODUCTION

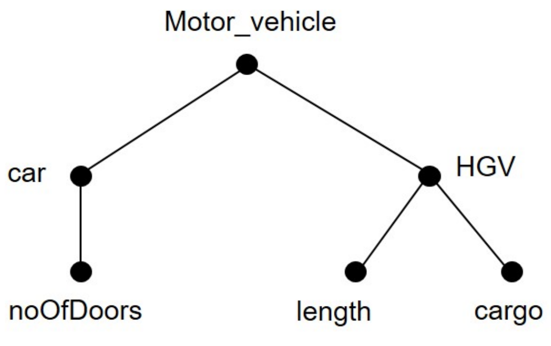
Enabling semantics in data processing allows data users to understand data sets without detailed system knowledge and extensive research. It also mitigates interpretation errors or missing units of measurement and enables systems to store meta information alongside the original data set. Semantics are usually expressed using annotations to one or more data attributes of a data set containing an arbitrary but homogeneous number of data points. These annotations set each label of the data set into a semantic context to create a domain specific meaning. Without those additional information, the user (e.g., data analyst or broker) has to interpret the meaning of certain attributes only by the attribute name, the actual values of such attributes, or by using a possibly existing documentation. Unfortunately, most data sets are created by persons who assume implicit domain knowledge or use identifiers which only they can understand, e.g., names or identifiers which have been agreed upon on company site, but which are otherwise not as easy to understand. Furthermore, if the used format does not allow any hierarchy, such as in CSV files, relations between attributes are not expressed.

In the context of the Industrial Internet of Things and Big Data approaches in todays companies, this leads to unforeseen problems when data sets from multiple sites are combined, e.g., while using a data lake. Analysts mostly do not possess sufficient detailed domain knowledge of each site and each process to fully understand implicit conventions in naming or interpreting cryptic data attribute names such as ’a24d7ff-2’. In the wake of big data technologies, data is gathered from all available sources without considering the compatibility and usability of random collected data sets from different branches, subcompanies, production sites and countries.

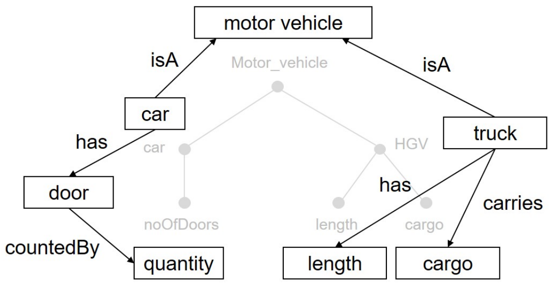
Using semantic annotations allows data providers (persons who own/create data) to annotate their data sets such that other persons can understand the data without the provider’s explicit knowledge. By additionally defining relations between the concepts, one can describe the data set in more detail. This is called semantic modeling. Figure 1 shows an example data set in raw format and with its semantic model. In the annotated version, meta concepts (i.e., concepts without any data attached to them), such as ’motor vehicle’ can also be added to indicate the linkage of two or more attributes. Semantic models usually also provide information about units of measurement, relations to other attributes or domain information that helps to understand the context of the acquired data. The semantic model is not limited and depends on the provider’s view of the world to fully understand the information contained in the data set. Usually, semantic models are based on an underlying ontology describing all possible or valid relations and entities.

When a data set is about to be shared, the provider has to create such a semantic model. Therefore, the

Paulus, A., Pomp, A., Poth, L., Lipp, J. and Meisen, T.
Gathering and Combining Semantic Concepts from Multiple Knowledge Bases.
DOI: 10.5220/0006800700690080
In *Proceedings of the 20th International Conference on Enterprise Information Systems (ICEIS 2018)*, pages 69-80
ISBN: 978-989-758-298-1



(a) Data model



(b) Semantic model

Figure 1: Exemplary data set before and after semantic modeling. During the process, each attribute has been assigned a semantic concept which describes the attribute. In addition, relations allow to refine the meaning of the used concepts.

provider annotates each data attribute of the data set with an entity that is available in the ontology. In addition, he can define relations between the defined entities. As manually selecting and creating concepts for each attribute can be a time consuming task, our goal is to automate this step as far as possible.

In this paper, we present our approach of a concept recommendation framework, which generates a selection of possible semantic concepts from which the user can choose. Our framework provides an abstraction layer for external sources, such as BabelNet, WordNet or specific domain ontologies. This layer enables users to also add additional knowledge bases, such as Wikipedia, to the recommendation process. Results from all sources are combined to form a single list of recommendations, relieving the user of searching for concepts and allowing an automated annotation in the best case. The framework design is based on an intensive review of the labels observed in real-world data sets that are either available on Open Data platforms or were obtained from local companies. We implemented and evaluated the framework in the context of the semantic data platform ESKAPE (Pomp et al., 2017a), (Pomp et al., 2017b). Hence, ESKAPE's knowledge graph will be one of the knowledge bases used by our recommendation framework.

The remainder of this paper is organized as follows: Section 2 motivates the necessity to develop a generic framework for gathering semantic concepts. Afterwards, Section 3 provides an overview of related approaches and Section 4 defines the problem classes which we identified when reviewing real-world data sets. Based on these classes, we present our approach in Section 5 consisting of querying, weighting and aggregating results from multiple knowledge bases. Section 6 gives an evaluation of our approach before we conclude the paper with a summary and a short outlook in Section 7.

2 MOTIVATING EXAMPLE

In this section, we provide a motivating example illustrating the necessity for annotating data with semantic models and supporting the modeling process with a framework that enables semantic concept gathering from multiple knowledge bases.

The scenario consists of a simplified production process of a large global enterprise with multiple sites in different countries. The enterprise is specializing in manufacturing products that need to be deformed and painted. Example products may be bicycles or cars. On the production sites, different versions of the good are produced in multiple steps by using different production lines and machines.

During the production process, different parts of the involved systems generate different kinds of data. As a global strategy for improving the manufacturing process based on data science, the enterprise decided to collect all these data in a centralized data lake. This lake collects all kinds of batch and streaming data produced by any production line and machine. To additionally support the process of finding exactly the data required by the data analysts, the management decided to set up an enterprise ontology with the help of two external experts. Based on this ontology, the employees who are responsible for a data source (e.g., the data produced by the sensors of one machine) have to create semantic models. We consider these employees to be domain experts knowing the meaning (semantics) of the data, calling them *data providers*.

For example, the data provider DP_1 creates the semantic model for the data set illustrated in Table 1. Please note that this is just an example data set. Industrial real-world data sets usually contain hundreds of columns. For defining the semantic model, DP_1 is forced to use all the entities and relations (vocabulary) that are available in the enterprise ontology. Figure 2

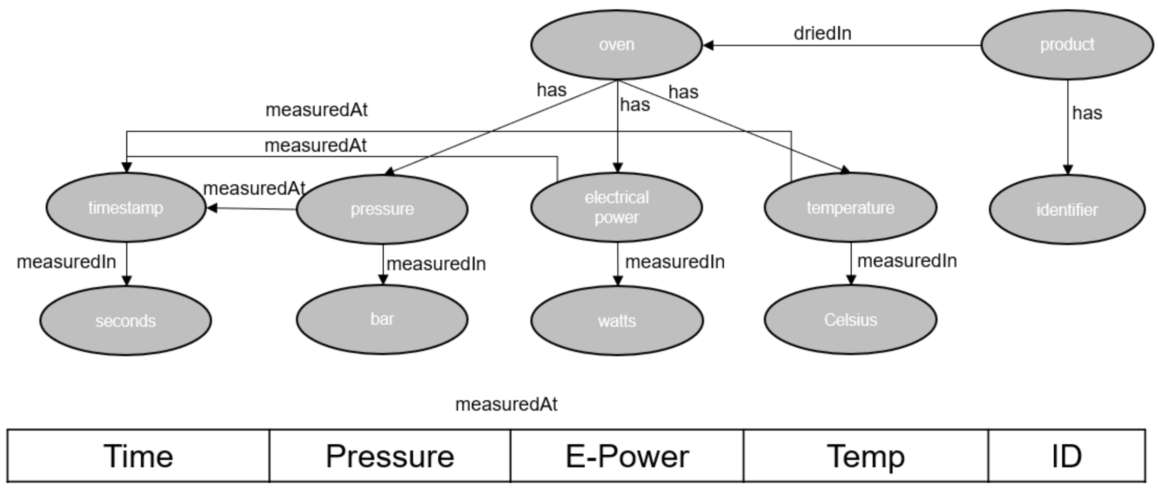


Figure 2: Exemplary semantic model for the data set provided in Table 1. The concepts used in the semantic model are obtained from the underlying ontology of the enterprise.

Table 1: Exemplary data set for which DP_1 has to create a semantic model.

Time	Pressure	E-Power	Temp	ID
1476835200	0.50	1400	312	4
1476845200	0.49	1350	311	5
1476855200	0.50	1445	314	6
...	...	...	...	...

illustrates the example semantic model that was created by DP_1. Assuming that all data sets of the enterprise are annotated with semantic models, a data scientist DS_1 can now search for data sets using the vocabulary defined in the enterprise's ontology. In addition, the provided semantic model helps the data scientist to understand the meaning of each data value.

When examining this scenario, we identify different drawbacks of the common solution for the current process of data providing as well as data consuming. First, both the data provider DP_1 and the data scientist DS_1 are limited to the vocabulary provided by the underlying ontology. Terms such as synonyms, that are not present in the ontology but may refer to the same entity cannot be used for creating semantic models or searching data sources. As ontologies can become very large and unhandy, it will be challenging for both parties to get an overview about the available vocabulary. Hence, a system which is capable of using a more advanced vocabulary without adjusting the enterprise's ontology would be desirable.

Another drawback of the current scenario is that data providers have to create the semantic models manually. To support a user in creating semantic models, it will be necessary for a system to automatically identify the concepts that map best to a specific column. This is especially important if we consider large data sets containing hundreds of columns. However, comparing the data labels with the concepts available in the enterprise's ontology will also lead to different challenges. In the example illustrated above we mapped the concept *Temperature* to the column *Temp*. Here, a simple comparison of the label *Temp* with the concept *Temperature* will not yield any result. As data labels in the real-world are very heterogeneous and are rarely annotated with meaningful terms (cf. Section 4), a simple comparison against an enterprise ontology would fail.

Finally, when using an enterprise ontology designed by experts, the ontology would need to cover any concept and relation that will be available in data sets that are added in the future. Otherwise, the data publisher could not map any concept from the ontology to the new data attribute. While defining a comprehensive ontology may be realizable in a closed and controllable environment (e.g., inside a single department of a company), it is very unrealistic for a global company with multiple sites in different countries or in an open scenario where data sources are added over a long time period by multiple independent actors, such as local companies, city administration or private end users.

Hence, it would be desirable to develop a system that is capable of combining the knowledge of an enterprise ontology with additional external knowledge bases and pre-processing steps to identify exactly those concepts that are relevant for creating or querying semantic models.

3 RELATED WORK

The problem of finding a correct or at least meaningful semantic concept for a given label or search term is well-known in research areas like ontology matching and alignment, schema matching, semantic tagging, semantic annotation or semantic modeling. Researchers working in these areas came up with different solutions for generating meaningful labels that help them to solve their problem.

In the areas of schema matching as well as ontology matching and alignment, the goal is to align a given ontology A to another ontology B that differ in number and kind of used relations or entity labels. To target the problem of finding the correct label in ontology B for a given entity of ontology A, approaches like structure-based, instance-based or lexical-based methods are frequently used strategies (M. H. Khan et al., 2015). For lexical-based as well as structure-based methods, the use of upper ontologies (cf. (Mascardi et al., 2010)) or external sources, such as Wikipedia, DBPedia or WordNet is very common (cf. (M. H. Khan et al., 2015), (Smirnov et al., 2011), (Maedche et al., 2002)).

Compared to ontology matching, the research area of semantic annotation targets the problem of finding an appropriate semantic concept for a given label of a data attribute (e.g., the label of a column in a table) (Goel et al., 2012), (Ramnandan et al., 2015). The field of semantic modeling goes one step further by additionally generating meaningful semantic relations and more abstract concepts (e.g., combining the concepts *name*, *age* and *hair color* to the abstract concept of a *person*) (Taheriyan et al., 2016), (Taheriyan et al., 2015). To perform the task of semantic annotation, Goel et al. (Goel et al., 2012) use the data provided by previous data sets to train a machine learning model that learns the representation of the data enabling to suggest semantic concepts without considering the label. As opposed to this, approaches like (Syed et al., 2010) or (Wang et al., 2012) use external knowledge bases, such as WordNet, Wikipedia or Probase to suggest concepts based on the present data labels. To create full-fledged semantic models, Taheriyan et al. (Taheriyan et al., 2016), (Taheriyan et al., 2015) additionally use specific domain ontologies and frequently occurring relations in Linked Data sources.

Beside the presented research fields, the area of semantic tagging focuses on either automatically generating or suggesting accurate tags for unstructured and binary data, such as documents, images and audio or video files (Du H et al., 2012). Proposed solutions use tag ontologies (e.g., MOAT and SCOT) (Kim et al., 2008), search engines such as Google (Singhal and Srivastava, 2014) or external knowledge bases such as WordNet, DBPedia, OpenCyc or Wikipedia (Du H et al., 2012), (M. Kalender et al., 2010), (Hong et al., 2015). Beside the already mentioned external knowledge bases WordNet, Wikipedia, OpenCyc or DBPedia, we identified additional useful sources. BabelNet (Navigli and Ponzetto, 2012) is a fully automatic generated multilingual semantic network combining knowledge from multiple other sources (e.g., WordNet, DBPedia or Wikipedia). However, BabelNet lacks the possibility to be extended by third-parties, which results in, e.g., missing domain knowledge. In addition, the automatic gathering of the existing knowledge bases may lead to inconsistencies. Another external knowledge base is Wolfram Alpha (Wolfram Alpha, 2017), a search engine that allows to retrieve knowledge about topics, such as physics, materials or people and history. Hence, this knowledge base covers very specific knowledge which may be missing in the other knowledge bases.

The developed approaches of the discussed research areas use different strategies to solve their problems. Here, most of the presented solutions rely on the use of one or more preset external knowledge bases to find the best matching concept. Hence, compared to the presented approaches, our work focuses on developing a single semantic search framework, which is capable of returning a set of best matching concepts. Instead of predefining a set of possible external knowledge bases, our work offers the user the possibility to add more knowledge bases and prioritize them. In comparison to BabelNet, our approach does not store the results of the connected sources. It queries, merges and sorts the results on-demand. Furthermore, it enables enterprises to connect their own external knowledge bases (e.g., a unified company-wide ontology) without leaking this internal knowledge to the public.

Compared to the discussed broader research fields, domain specific approaches like the NCBO Ontology (Jonquet et al., 2010) are a first attempt in recommending the best fitting ontology for a predefined use case. Therefore, users can enter keywords or a full-text and the recommender identifies the best fitting ontologies. However, compared to our work, the Ontology recommender aims in identifying the best fitting ontology whereas the goal of our approach is not to identify the best fitting knowledge base. Instead, our approach tries to identify the best fitting concept based on all concepts that are available in the connected knowledge bases. The underlying system can then add this concept to its ontology or knowledge graph if it is not present yet.

To identify the most fitting concept, semantic relatedness is a prominent measure to use. It helps identifying clusters of possible candidates by comparing the semantic concepts to each other and computes a measure of how likely those elements are. (Banerjee and Pedersen, 2003) propose the use of gloss overlaps to compute the measure. However, we extend and refine this approach by extracting keywords from the text first, thus generalizing the approach and releasing it from WordNet relations to allow general comparisons between concepts.

4 REVIEWING REAL-WORLD DATA SETS

Before we started to design and implement our search framework, we performed an intensive research on the way people label columns and data attributes in real-world data sets. We collected the data from multiple Open Data platforms such as San Francisco, New York, Los Angeles, Berlin, Aachen, Munich and multiple other cities as well as from OpenGov, mCloud and from multiple local partner companies. We limited our review to five common structured data formats (CSV, XML, XLSX, JSON and GJSON) and collected a total of 272 files. Afterwards, we manually reviewed each data set to identify problems that will occur when trying to identify a label for the corresponding column or data attribute. We used the gained insights to formulate problem classes for data labels as well as data sets. A label can belong to none, exactly one or multiple problem classes. However, some of the problems may occur together whereas other problems directly exclude others. Depending on the classes a label or a data set belongs to, it will be more or less difficult to identify the correct semantic concept. Altogether, we identified the following ten problem classes for labels:

- **Abbreviations:** Instead of writing the complete label, people tend to abbreviate labels. Prominent examples we found are *lat*, *lon*, *www*, etc.
- **Different Languages:** Depending on the source of the data, labels may be in different languages. Moreover, it may occur that languages are mixed within data sets.
- **Natural Language:** Instead of labeling data with a concrete concept or a single word, people tend to describe data attributes in more detail. One example is *'Number of accidents where persons got injured'*.
- **Time Labels:** Depending on the data, people tend to label columns or data attributes with points in time or time spans *(e.g., Monday, 1-2PM, etc.)*.
- **Misspelling:** The person who labeled the data made a simple mistake. Examples are *Acess Point, Telehphone Number,* etc.
- **Splitting Characters:** Beside white spaces in labels, some labels contained special characters *(e.g., '_' or '-')* to split words. One example we found is *telephone_number*.
- **Camel Case Input:** Similar to splitting characters, some persons tend to split words using the camel case syntax *(e.g., StreetNumber)*.
- **Additional Information:** In a few data sets, we identified labels that contained additional valuable semantic knowledge that can be used to specify the semantic concept in more detail (*e.g., temperature in* °*C*).
- **Plural:** In multiple data sets, people labeled columns using the plural instead of the singular *(e.g., names vs. name or street vs. streets)*.
- **Random Labels:** Labels which do not follow any meaning and are just random generated strings, such as hash codes or increasing numbers. Examples are *1, 2, 8321b319b1781,* etc.

Beside the challenges that we identified for single labels, we also identified a problem class for a complete data set. We observed that identifying the correct concept for humans becomes much simpler and unambiguous if the labels within the data set belong to the same domain. Hence, we created the class domain context.

- **Domain Context:** The labels within data sets can either belong to the same domain or they may belong to different domains.

Depending on the classes a label belongs to, finding appropriate concepts that specify the information that is present in the data set can be more or less challenging. While classes like *Abbreviations*, *Different Languages*, *Misspelling*, *Splitting Characters*, *Camel Case Input* and *Plural* can be tackled by suitable preprocessing steps, classes like *Natural Language*, *Time Labels* and *Additional Information* or *Random Labels* require more sophisticated strategies. For instance, one will never be capable of identifying appropriate concepts for labels that belong to the *Random Label* class by just considering labels. Here, we require strategies that consider the data within the column to achieve appropriate results. On the other hand, the class of *Additional Information* requires the search framework to consider and model the additional information. Hence, the result for a label can be more than a single concept. It can also be a small graph describing the semantics of the label.

5 SEMANTIC CONCEPT GATHERING

In this section, we describe the main concept and prototype of our recommendation framework mainly focusing on the *Domain Context* problem class. We give necessary background information on semantic networks and present the main approach afterwards.

5.1 Semantic Networks

Semantic networks help modeling information by defining concepts (e.g., *engine*, *boat*) and relations (e.g., *has*, *isA*, *partOf*) to represent knowledge that is available in the world, similar to an ontology. Those networks exist to help machines gain meta information to their data by modeling this knowledge in a standardized way. Semantic networks can either model domain specific or general purpose knowledge. Most of them are available online with a public API, which can have metered access in some cases.

However, when receiving a result from a semantic network, those replies may contain multiple similar concept suggestions, usually from different domains. Identifying the one concept that can be aligned with concepts found for other labels in the data set can be a challenging task. Some semantic networks offer a way of querying multiple concepts at the same time, yielding a combined result for all labels (e.g., the publicly available Babelfy).

For example, Table 2 shows the first resulting concept from BabelNet for the search term 'car'. In addition to the main label 'car', several synonyms are given in combination with a short glossary description of the concept. This glossary can help to distinguish between multiple similar concepts from different domains. The result also contains meta information like other related concepts (e.g., car *isA* motor vehicle). All elements obtained from a source also have a unique identifier which helps identifying equal concepts, thus allowing us to combine several search results from the same data source.

When receiving a response from a semantic network, multiple possible concepts can be included, e.g., *IP address* or *postal address* for the label *'address'*. Hence, the resulting challenge is the selection of the most fitting concepts for our queried label. By additionally considering the concepts obtained for the other labels (context) and by querying multiple semantic networks for the complete set of labels, the challenge is getting bigger. However, not all semantic networks may return a concept for a queried label. Hence, the concepts returned for other labels might originate from other semantic networks thus not allowing a comparison based on relations available in a single semantic network. In the following, we present our semantic concept recommendation framework to mitigate this problem.

5.2 Concept

Detecting suitable semantic concepts for data attributes is a multi-layered task. First, a set of possible matches has to be gathered for each label, followed by an evaluation of those candidates. From all possible candidates, the best candidates have to be selected for presentation to the user. However, those recommendations can vary heavily if we consider different domains in which our semantic model is built. A single label can have multiple meanings in different domains or in rare cases even in the same domain if combined with specific other concepts of that domain. This implies that, in order to present a matching candidate, the recommendation framework has to detect the domain and consider not only the expression but also parent or sibling elements. The framework's task is to provide as many suitable matches for semantic concepts as possible for each input label.

To achieve this goal, our framework is comprised of different parts. Before the knowledge bases are queried, the label is pre-processed in multiple steps to tackle the defined problem classes (e.g., Different Languages, Misspelling etc.) and prepare it for the connectors. Next, the communication to semantic sources is realized using connectors, which are proxy modules helping to retrieve a standardized result from the different and independent knowledge bases. All connectors provide their results to the recommendation module which computes the most likely matches from all candidates and returns the result to the user.

5.2.1 Querying Semantic Networks

As a first step, using the connectors, a query is sent to selected semantic networks (cf. Section 5.1) which return a list of possible concepts for this expression. New semantic networks can be added by implementing connectors for those sources and registering them in our framework. A schematic view of a connector for a data source is given in Figure 3. The label is prepared for each connector such that it closely matches the naming structure of the connected source. Next, each connector runs the query and transforms the result to a homogeneous representation which can be handled by the framework. Based on the type of network queried, the returned set of concepts might include unique IDs, glossaries and additional linked concepts like hyper- and hyponyms. Those additional linked concepts can range from zero or a few entries

Table 2: Example query result for 'car' from BabelNet. Linked concepts are underlined.

Field	**Content**
Label	automobile, car, auto, machine, motorcar
Glossary	A <u>motor vehicle</u> with four <u>wheels</u>; <u>usually</u> <u>propelled</u> by an internal <u>combustion engine</u>
Is A	<u>motor vehicle</u>
Category	Automobiles, Wheeled vehicles

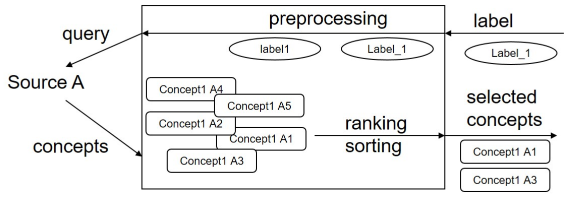


Figure 3: Querying a semantic source by pre-processing the input, pre-selecting a fixed amount of resulting concepts and returning the result to the recommendation module.

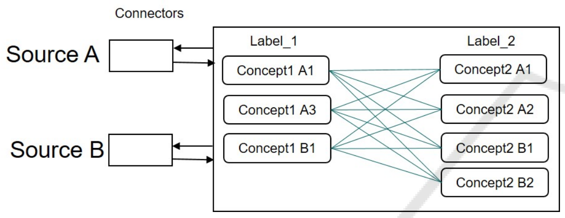


Figure 4: Comparing all semantic concepts to candidate concepts for other labels. Each similarity boosts this concept's score.

to more than a thousand. Therefore, all queries are filtered by a set of considered relations (e.g., *isA* or *partOf*) to reduce the complexity of the following steps. As our framework can currently only consider a limited amount of five relations, all source-specific relation types have to be mapped to a set of preset relations or be dropped during the implementation of the connector.

5.2.2 Intra Source Selection

Beside the mapping of relations, a pre-selection of concepts is done inside the connector. The metric which determines the candidates selected is currently chosen by the connector's developer, but could be configurable later. Depending on the connector, results can be filtered and sorted by specified metrics (e.g., sort the concepts by the number of edges they have to others and take the first *n*). However, any other metric is possible. To reduce the implementation overhead for each connector, the pre-selection on the concepts is solely done for one label. The relations between the queried label and the other available labels are not considered at this point. This is currently done in the *Inter Source Scoring* step (cf. Section 5.2.3).

After all query results have been obtained and pre-selected, all candidates from different sources for a given label are combined to a single list (cf. Figure 4). Duplicates that are obtained from the same data source are eliminated per list (e.g., BabelNet returning WordNet concepts).

Hence, formally, we have a set K of m knowledge bases and a data set with n labels. For each label l_i where $i \in \{1...n\}$, we query each knowledge base K_j where $j \in \{1...m\}$ and receive a set of concepts $C_{l_i K_j}$. Finally, we combine all results from all knowledge bases K_j and for the same label l_i into a single set C_{l_i}.

5.2.3 Inter Source Scoring

In this process, we compare the set of concepts C_{l_i} for each label l_i with the set of concepts C_{l_j} for each label l_j where $j \in \{1...n\}$, $j \neq i$ to select the most fitting concepts for each label. Therefore, all concepts are scored based on their coherence with concepts found for other labels. Assuming that multiple sources yield results from different domains, we want to identify the most prominent domain or combination of labels. As we do not have any valid relations between concepts from different semantic networks, we can only consider the concept's name and the description provided by the network as well as possible linked concepts (from the same source).

Scoring is done by comparing keywords from concept labels, glossary and related concept labels. For two concepts c_a and c_b the coherence score is computed using five partial scores s. We assume that each concept c_x consists of a set of main labels ml_x, a glossary g_x and a set of related concept labels r_x. As glossaries mostly contain natural language, a keyword extraction algorithm KE is used to extract keywords from the text.[1]

$$
\begin{aligned}
c_x &= (ml_x, g_x, r_x) \\
s_1(c_a, c_b) &= \frac{|KE(g_a) \cap KE(g_b)|}{min\{|KE(g_a)|, |KE(g_b)|\}} \\
s_2(c_a, c_b) &= |ml_a \cap KE(g_b)| \\
s_3(c_a, c_b) &= |ml_a \cap r_b| \\
s_4(c_a, c_b) &= |r_a \cap r_b| \\
s_5(c_a, c_b) &= |ml_a \cap ml_b|
\end{aligned}
\tag{1}
$$

[1]Currently, we use https://stanfordnlp.github.io/CoreNLP/

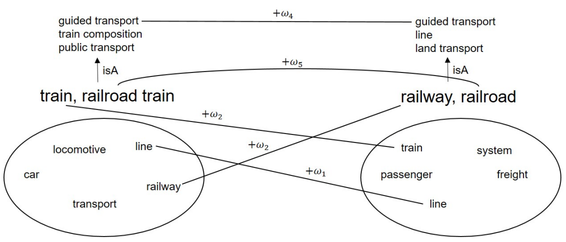


Figure 5: Example comparison of concepts 'train' and 'railroad' obtained from BabelNet and reduced for simplicity. Each match on one of the given scores adds the assigned weighting ω to the total score.

Equation (1) shows the definition of a semantic concept c_x for the algorithm and the aforementioned five partial scores.

- s_1 weights the matches from glossary keywords by comparing all found keywords for both concepts and scoring matches against the total number of possible matches which is equal to the number of elements in the smaller keyword set.
- s_2 yields scores if concept labels match to keywords in the second concept's glossary.
- s_3 scores the number of concept c_a labels matching labels in c_b's related concepts (e.g., parent-child relations). From related concepts, only the labels are checked, not the glossary or even more related concepts.
- s_4 increases if matches of both concepts' related entities exist (e.g., sibling relation indicated by same parent element).
- s_5 compares the labels of both concepts and increases if there is a direct match.

With C_{l_i} being the set of concepts for each label l_i, we define $C_l = \bigcup_{i=1}^{n} C_{l_i}$ as the set of concepts for all labels. Let $c_k \in C_{l_i}$ be a concept for which we want to calculate the coherence score. The total score S for the concept c_k is defined as:

$$S(c_k) = \sum_{c_o \in C_l \setminus C_{l_i}} \sum_{p \in 1,..,5} \omega_p \times s_p(c_k, c_o) \qquad (2)$$

Each score is weighted by ω to allow fine tuning of the results for each comparison. The resulting comparison can be seen in Figure 5. Here, each match on one score yields the mentioned ω value, and thus increases the score. By performing this calculation for each concept $c_k \in C_{l_i}$ and each $C_{l_i} \in C_l$ we receive a coherence score S for each concept of each label.

5.2.4 Sorting and return

After weighting and comparing, the initial user preference as given in the request is applied. This user preference contains which sources shall get a higher priority when merging the results. Therefore, we multiply the retrieved scores for all concepts with a factor given for their original data source. This enables the algorithm to, e.g., prioritize domain-specific sources before common networks. The candidate list of each node is then sorted in descending order and the top n elements of each list are selected and returned to the user. This way, the framework yields a number of ranked candidate concepts, including their description and possible other meta information as well as linked concepts.

5.3 Implementation

For evaluating our approach, we implemented the concept as a Java application available via REST interface. The application expects a JSON Object containing the labels of the data set that should be annotated and a weighting for each knowledge base enabling the user to easily prioritize those knowledge bases from which he expects better results (e.g., those containing domain knowledge). Connectors for WordNet, BabelNet and the internal smart city knowledge graph available via ESKAPE have been implemented as proof of concepts. The Google Knowledge Graph was also connected in earlier stages but did only perform well on named entities, which are less common in data set labels. It was therefore skipped for the evaluation.

Before labels of a data set are sent to the different semantic networks, they are pre-processed to improve the result's quality and comparability. For example, to tackle the mentioned problem classes, the labels are converted to lowercase, translated to English and all special characters, such as '-' or '_', are removed. In addition, the camel case labels are split into multiple words. Further pre-processing steps, such as resolving labels for abbreviations, are not implemented yet.

After pre-processing the labels, the framework queries the semantic networks for concept suggestions for all labels. To narrow down the number of results from the semantic networks, only concepts are added to the list of suggestions. Named entities are currently not considered in the framework as they rarely appear as labels in data sets. The connectors are set to return up to $n = 5$ elements to the engine. As described in Section 5.2, the returned elements are filtered for duplicates and rated according to the defined steps. As the amount of comparisons between the different suggestions grows quickly with a larger data set, multi-threading is used for the rating process. Thereafter, for each label, the suggestions are sorted by their rating and the top five entries are returned.

The weighting inside the algorithm is currently undergoing constant evaluation with different factors while keeping the initial order $\omega_1 \leq \omega_2 \leq \omega_3 \leq \omega_4 \leq \omega_5$. Our initial setting, which is also used during evaluation initializes ω_1 with 1, ω_2 with 2 and so forth. We disabled all source filtering and weighted all sources equally.

6 EVALUATION

We evaluated our approach by annotating different publicly available data sets using results from WordNet (WN), BabelNet (BN) and a small knowledge graph (KG) covering the domain of smart city and transportation. Table 3 shows the results of an annotation of the crime data set from the public domain of Vancouver[2]. For all contained labels, we measured the number of returned concepts (# column) and, if a suitable concept has been found, how high it was ranked in the returned list (*c* column). We observed that general concepts like 'month', 'year', 'hour', 'minute' and 'neighborhood' are easily detected as they are quite unique and can hardly be interpreted for something else. The label 'hundred_block' returned, although normalized, no results. This is mostly due to the fact that we currently cannot safely handle multi word labels. In addition, our framework also lacks public semantic networks of this very specific term from the domain of rural addressing or law enforcement (which is also not covered by the smart city KG). However, due to the design of ESKAPE's knowledge graph, ESKAPE is capable of learning this concept (cf. (Pomp et al., 2017b)). The 'x' and 'y' labels, which represent coordinates in this model could not be identified, and although they were added to the KG by another source and therefore recognized, they were ranked low (rank three and four) in the process as we did not boost the KG ratings as a domain specific source. This lead to a higher ranking for more general concepts like 'X [24th letter of Roman alphabet]'. The total processing time for this request was on average 2.18 seconds, including queries.

The results from all evaluation sets can be seen in Table 4. For this evaluation, we chose another data set from the public domain of Vancouver[3], two from the San Francisco transportation domain[4][5] and one from the European Data Portal[6]. The total number of labels for each data set is written in brackets, with the position columns indicating how many most fitting labels were found on that position in the returned lists. For most of the labels, we could find suitable matches, although not always in prime positions. This shows that our basic idea of presenting the top *n* results instead of the best ranked result will lead to better recommendations in a production system. As in the case of the crime data set, missing concepts were mostly caused by sources that did not offer any concepts at all for specific labels which we could trace back to some of our defined problem classes (cf. Section 4) like 'Applied Color Rule', 'Object ID', which is far too generic, or 'CULTURAL_SPACE_NAME' which could not be identified altogether. However, the effect of adding a possibly proprietary semantic source is clearly visible as multiple concepts, e.g., from the clearance data set, could be resolved by using a source containing smart city domain knowledge. Multiple of those sources could be attached to improve the results.

We also compared our approach to Babelfy, an online algorithm dealing with the context aware semantic annotation of sentences, keywords or labels. Those keywords can be inserted at the same time, allowing Babelfy to consider relations of the underlying BabelNet network when building the reply. We tested a general set of labels which would completely rule out the smart city knowledge graph as the labels belong to the domain *IT infrastructure*. Thus, both algorithms operate solely on the WordNet/BabelNet networks. For labels, we chose a common combination on a list of network participants: 'computer', 'lan', 'network', 'subnet', 'address' 'port'. This combination has multiple domain specific meanings (e.g., 'address' can either be IP or MAC address or postal address) to test the detection of the right domain. The results can be seen in Table 5. Both algorithms managed to clearly identify unambiguous labels like 'computer' or the acronym 'lan' 'network' was interpreted differently in both cases and no superior concept could be identified between both suggestions. However, considering the ambiguous labels 'subnet', 'address' and 'port', our framework performed better than Babelfy, which in all cases returned the most common concepts when querying for the label without any context and in case of 'subnet' even chose the sub par second concept for a mathematic topology. Although we cannot state that our approach will be performing well on all label combinations, it has been shown that even the baseline algorithm can improve the recommendation of semantic concepts for labels.

[2]http://data.vancouver.ca/datacatalogue/crime-data.htm

[3]http://data.vancouver.ca/datacatalogue/culturalSpaces.htm

[4]https://data.sfgov.org/Transportation/Clearance-Heights-for-Large-Vehicle-Circulation/sccd-iwvp

[5]https://data.sfgov.org/Transportation/Meter-Operating-Schedules/6cqg-dxku

[6]https://www.europeandataportal.eu/data/en/dataset/east-sussex-county-council-recycling-sites

Table 3: Results for the crime data set for sources Wordnet (WN), WordNet and BabelNet (WN/BN), Knowledge Graph (KG) and all sources combined. The # column indicates the considered number of results from a source, *c* indicates the position of the (subjectively) most matching concept in the suggestion list.

	Sources	WN		WN/BN		KG		All
	Numbers	#	c	#	c	#	c	c
Label	hundred_block							
	month	2	1	4	1			1
	year	4	1	5	1			1
	neighborhood	2	1	5	1			1
	hour	4	3	5	1	1	1	1
	x	3		5		1	1	3
	y	2		5		1	1	4
	minute	5	2	5	1			1
	type	5		5		1	1	3
	day	5	3	5	1			1

Table 4: Summary of results from WordNet (WN), WordNet and BabelNet (WN/BN) and our smart city knowledge graph (KG) on publicly available data sets (numbers in brackets indicate total number of labels). *Position* indicates the position of the (subjectively) most fitting concept in the suggestion list.

	Sources	WN			WN/BN			KG			All		
	Position	1	2	3	1	2	3	1	2	3	1	2	3
Data set	crimeData(10)	3	1	2	6	0	0	3	0	0	6	0	2
	culturalSpaces(11)	4	2	0	5	1	1	0	0	0	6	1	0
	recycling(7)	4	0	1	4	0	1	0	0	0	4	0	1
	clearance(6)	1	1	1	2	1	0	3	0	0	2	2	0
	schedules(12)	1	1	0	1	0	1	5	0	0	5	0	1

Table 5: Comparison of Babelfy results to top concepts found by our framework. Text in brackets indicates a description to distinguish different concepts of the same name. Semantic source of both results was BabelNet only. Bold markings indicate (subjectively) better matching results.

label	**Babelfy result**	**Framework result**
computer	**computer**	**computer**
lan	**local area network**	**local area network**
network	[interconnection of things or people]	Internet
subnet	subnet [mathematic topology]	**subnetwork [IP]**
address	address [postal]	**IP address**
port	port [sea port]	**interface [computing]**

7 CONCLUSION AND FUTURE WORK

In this paper, we presented the current state of a framework for combining multiple semantic networks to suggest semantic concepts for multiple labels. Results form different semantic sources are combined solely by comparing (main) labels, glossary and labels of related concepts as no valid relations exist between concepts of separate networks. We have shown that our framework is able to use knowledge from public sources as well as proprietary sources undisclosed to the public (e.g., company networks). Using a modular approach, our framework can be extended using connectors to attach new semantic sources. The scoring of single concepts is retrieved by comparing its attributes to those of possibly related concepts in pursuit of identifying the correct domain.

We achieved acceptable results when applying our approach to data sets from public sources. We could assign matching concepts to labels, especially when domain-specific labels were annotated. This shows that the strength of our framework is strongly based on the sources connected, as some entries could not be found in any data source and therefore remained unassigned at all. It has to be tested, whether our approach is capable of handling results from more than three sources and if the domain identification, e.g. as shown in the computer network example, still performs as strong as it did in our current evaluation. It could be seen that the weighting of different sources is important, as multiple sources providing different definitions of a label contest against each other and more general concepts like 'X' as a letter sometimes get higher scores in direct comparison. However, as no domain has been declared during a first matching run, this has to be done manually or with the help of multiple runs.

For future work, we are planning to improve the framework by establishing a more iterative ranking procedure. Instead of ranking all single concepts against all other available concepts, we will establish a ranking system which iteratively combines concepts to larger tuples. This way, the framework would not return a list of possible concepts for each label, but a combination of probable concepts which harmonize best. In addition, we are planning to increase the flexibility of the connectors to allow for more relations to be considered which could help to improve intra-source selection of fitting labels. Implementing a subgraph matching similar to the one used by Babelfy could be a further possibility in the future. This would allow the framework to select only the most context fitting concepts before the comparison algorithm runs, thus keeping the runtime low. Altogether, we try to improve our algorithm to perform better on the identified problem classes. Here, we especially want to focus on the problem class of *Random Labels* by examining and comparing presented data with already captured data. Finally, we are planning to extend the framework to identify and return concept subgraphs for labels that contain more than a single semantic information.

REFERENCES

Banerjee, S. and Pedersen, T. (2003). Extended gloss overlaps as a measure of semantic relatedness. In *Ijcai*, volume 3, pages 805–810.

Du H, W., Rau, J. W., Huang, J. W., and Chen, Y. S. (2012). Improving the quality of tags using state transition on progressive image search and recommendation system. In *2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 3233–3238.

Goel, A., Knoblock, C. A., and Lerman, K. (2012). Exploiting structure within data for accurate labeling using conditional random fields. In *Proceedings of the 14th International Conference on Artificial Intelligence (ICAI)*.

Hong, H. K., Park, K. W., and Lee, D. H. (2015). A novel semantic tagging technique exploiting wikipedia-based associated words. In *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual*, volume 3, pages 648–649.

Jonquet, C., Musen, M. A., and Shah, N. H. (2010). Building a biomedical ontology recommender web service. *Journal of Biomedical Semantics*, 1(1):S1.

Kim, H. L., Passant, A., Breslin, J. G., Scerri, S., and Decker, S. (2008). Review and alignment of tag ontologies for semantically-linked data in collaborative tagging spaces. In *Semantic Computing, 2008 IEEE International Conference on*, pages 315–322.

M. H. Khan, S. Jan, I. Khan, and I. A. Shah (2015). Evaluation of linguistic similarity measurement techniques for ontology alignment. In *Emerging Technologies (ICET), 2015 International Conference on*, pages 1–6.

M. Kalender, J. Dang, and S. Uskudarli (2010). Unipedia: A unified ontological knowledge platform for semantic content tagging and search. In *Semantic Computing (ICSC), 2010 IEEE Fourth International Conference on*, pages 293–298.

Maedche, A., Motik, B., Silva, N., and Volz, R. (2002). Mafra—a mapping framework for distributed ontologies. In *Knowledge engineering and knowledge management: ontologies and the semantic web*, pages 235–250. Springer.

Mascardi, V., Locoro, A., and Rosso, P. (2010). Automatic ontology matching via upper ontologies: A systematic evaluation. *IEEE Transactions on Knowledge and Data Engineering*, 22(5):609–623.

Navigli, R. and Ponzetto, S. P. (2012). Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Pomp, A., Paulus, A., Jeschke, S., and Meisen, T. (2017a). ESKAPE: Information Platform for Enabling Semantic Data Processing. In *Proceedings of the 19th International Conference on Enterprise Information*. SCITEPRESS - Science and Technology Publications.

Pomp, A., Paulus, A., Jeschke, S., and Meisen, T. (2017b). ESKAPE: Platform for Enabling Semantics in the Continuously Evolving Internet of Things. In *2017 IEEE 11th International Conference on Semantic Computing (ICSC)*, pages 262–263.

Ramnandan, S. K., Mittal, A., Knoblock, C. A., and Szekely, P. (2015). Assigning semantic labels to data sources. In *Proceedings of the 12th ESWC*.

Singhal, A. and Srivastava, J. (2014). Leveraging the web for automating tag expansion for low-content items. In *Information Reuse and Integration (IRI), 2014 IEEE 15th International Conference on*, pages 545–552.

Smirnov, A., Kashevnik, A., Shilov, N., Balandin, S., Oliver, I., and Boldyrev, S. (2011). Principles of ontology matching, translation and interpretation in smart spaces. In *2011 IEEE Consumer Communications and Networking Conference (CCNC)*, pages 158–162.

Syed, Z., Finin, T., Mulwad, V., and Joshi, A. (2010). Exploiting a web of semantic data for interpreting tables. In *Proceedings of the Second Web Science Conference*, volume 5.

Taheriyan, M., Knoblock, C., Szekely, P., Ambite, J. L., and Chen, Y. (2015). Leveraging linked data to infer semantic relations within structured sources. In *Proceedings of the 6th International Workshop on Consuming Linked Data (COLD 2015)*.

Taheriyan, M., Knoblock, C. A., Szekely, P., and Ambite, J. L. (2016). Learning the semantics of structured data sources. *Web Semantics: Science, Services and Agents on the World Wide Web*, pages –.

Wang, J., Wang, H., Wang, Z., and Zhu, K. (2012). Understanding tables on the web. *Conceptual Modeling*, pages 141–155.

Wolfram Alpha (2017). Computational Knowledge Engine. https://www.wolframalpha.com/.