Knowledge Graph Completion with FAMER

Daniel Obraczka[*]
obraczka@informatik.uni-leipzig.de

Alieh Saeedi[*]
saeedi@informatik.uni-leipzig.de

Erhard Rahm
rahm@informatik.uni-leipzig.de

University of Leipzig

ABSTRACT

We outline the use of the tool FAMER to address the schema and entity matching tasks for the DI2KG 2019 challenge. FAMER supports both the static and incremental matching and clustering of entities from multiple sources. To alleviate entity matching, we first identify matching properties in the provided datasets based on the similarity of property names and instance values. This approach utilizes the given training data to derive property matches from entity matches. For entity matching, we consider multiple configurations to determine entity similarities with the optional use of word embeddings.

ACM Reference Format:
Daniel Obraczka, Alieh Saeedi, and Erhard Rahm. 2018. Knowledge Graph Completion with FAMER. In *Proceedings of Anchorage '19: 1st International Workshop on Challenges and Experiences from Data Integration to Knowledge Graphs (Anchorage '19).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Knowledge graphs (KG) physically integrate numerous entities with their properties (attributes) and relationships as well as associated metadata about entity types and relationship types in a graph-like structure [5]. A product KG may thus contain a huge number of products of many types where the product types can also be organized in an ontological structure, e.g., to differentiate camera-related products into different kinds of cameras (DSLR, mirrorless, ...), camera parts (e.g. camera bodies, lenses, ...) and different kinds of camera accessories. The KG entities and relationships are typically integrated from numerous sources, such as other knowledge graphs, databases, web pages, documents etc. Integrating such sources implies a matching and fusion of equivalent entities and relations.

The initial KG may be created from a single source (e.g., a pre-existing knowledge graph such as DBpedia or the product KG of a specific merchant) or a static integration of multiple sources. KG completion (or extension) refers to the incremental addition of new entities and relationships. The addition of new entities requires solving several challenging tasks:

(1) *preprocessing* of new datasets for data profiling (e.g., to determine the cardinality and value ranges of properties) and data cleaning
(2) determining the entity type *(classification) of new entities*
(3) *incremental schema matching* to match and group (cluster) properties of new entities with known properties in the KG
(4) *incremental entity resolution* to match and cluster new entities with already known entities in the KG
(5) fusion of newly matching entities
(6) addition of relationships for new entities.

At the Univ. of Leipzig, we are developing a scalable framework for the end-to-end generation and maintenance of KGs building on our previous work on learning-based product matching [3] and parallel entity resolution [2]. The core of this framework is a new parallel tool called FAMER (FAst Multi-source Entity Resolution) for both static and incremental matching and clustering of entities from multiple sources [7]. FAMER first determines or updates a so-called similarity graph between entities of a certain type and then applies a clustering approach to determine or update clusters of matching entities. These clusters group matching entities from different sources and thus support both the fusion of matching entities as well the tracking of original entities (which is also helpful for a possible cluster repair).

We address both the schema and entity matching tasks of the DI2KG 2019 challenge for KG integration of product entities about cameras. We are not providing a full-blown schema (property) matching solution but focus on a simple approach to support entity matching on the most frequent properties. We also use FAMER for building a similarity graph on properties and to determine and incrementally update clusters of matching properties.

In the next section, we provide an overview about FAMER. We then describe how we address preprocessing and schema matching (Sec. 3) and entity matching (Sec. 4) for the DI2KG 2019 challenge. Obtained results will be described in the final version of this paper.

2 FAMER OVERVIEW

Figure 1 illustrates the main components of the FAMER framework for incremental matching. As outlined in [8][9][7], the framework consists of two major configurable phases (gray boxes) named *Linking* and *Clustering*. In the Linking phase, a similarity graph is generated so that similar entities are linked pairwise with each other. This phase starts with blocking on selected properties so that only entities of the same block need to be compared with each other. In the initial version of FAMER, pairwise matching is manually configured by specifying a combination of several property similarities that has to exceed a minimal similarity threshold. We have now also added support for learning-based linking configurations, e.g. using random forest classification, which utilizes training data of matching and non-matching entity pairs. Similar to [4], we also added support for word embeddings, e.g. using FastText,

[*]Both authors contributed equally to this research.

Anchorage '19, August 05, 2019, Anchorage, Alaska
© 2018 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/1122445.1122456

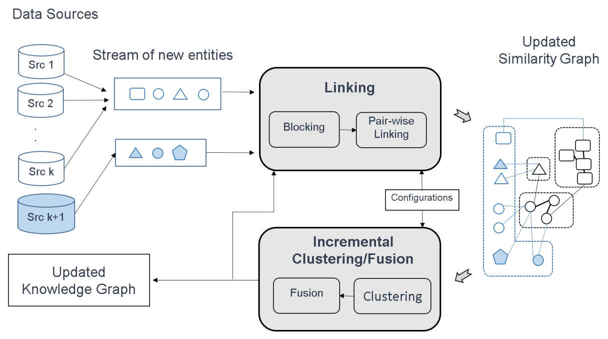

Figure 1: Incremental entity resolution with FAMER

to replace the value of string (textual) properties by their embeddings for a possibly improved similarity computation. Depending on the method to determine potential matches, the edges in the similarity graph include a similarity score to indicate the match likelihood. The second part of FAMER uses the similarity graph to determine entity clusters where a cluster groups all matching entities from the different input sources. Clustering can be based on different algorithms including the so-called CLIP approach favoring so-called strong inter-source links that connect maximally similar entity pairs from both sides [9].

FAMER is able to update the output result for new entities and new sources [10] as needed for KG completion. In this case, the input is a stream of new entities from known sources or from a new source plus the already determined entity clusters (stored in the KG) (Figure 1). Here, the *Linking* part focuses on the new entities and does not re-link among previous entities. The output of the linking is an updated similarity graph composed of existing clusters and the group of new entities and the newly created links (the light-blue colored group in Figure 1). The *Incremental Clustering/Fusion* part integrates the group of new entities into clusters. The updated clusters are fused in the *Fusion* component so that all entities are represented by a single entity called cluster representative.

FAMER is implemented using Apache Flink so that the calculation of similarity graphs and the clustering approaches can be executed in parallel on clusters of variable size. For the implementation of the parallel clustering schemes we also use the Gelly library of Flink supporting a so-called vertex-centric programming of graph algorithms to iteratively execute a user-defined program in parallel over all vertices of a graph. The vertex functions are executed by a configurable number of worker nodes among which the graph data is partitioned, e.g., according to a hash partitioning on vertex ids.

3 PREPROCESSING AND SCHEMA MATCHING

To illustrate the data quality problems in the given dataset of the DI2KG challenge, we show in Table 1 two matching *Nikon* camera products from different sources. We observe significant differences in the set of properties and property values. For example the first entity owns the property *features* while the second camera does neither contain this property nor the corresponding value (*Slimline*). This may happen even among entities of the same source. Moreover, the same property values are not represented similarly in different entities. For example in the first camera the property *camera resolution* with the value *16 Megapixels* is represented as *"approx resolution": "16MP"* for the second camera. Altogether, the challenge includes 24 sources with vastly heterogenuous schemas. For example, the source "ebay" has over 2000 properties some of which are likely duplicate properties such as "maximum shutter speed" and "max shutter speed".

Before we perform an incremental schema matching and entity matching we first perform *preprocessing* on the input dataset to derive some statistics and to perform data cleaning steps. In particular, we focus both entity and schema matching on the most frequent properties since infrequent properties are unlikely to be present for all matching pairs of entities so that their use is of limited value. For example the property "energy consumption per year" only occurs in one entity in the entire dataset and will therefore most likely not

Table 1: Example Raw Data

property	value
"35mm equivalent"	"25-300mm"
"<page title>"	"Nikon Coolpix S6800 Digital Camera (Black) | UK Digital Cameras"
"brand"	"Nikon"
"camera resolution"	"16 Megapixels"
"colour"	"Black",
"features"	"Slimline"
"hd video"	"Full HD (1080P)"
"lcd size"	"3.0”"
"lens tele mm"	"300"
"lens wide mm"	"25"
"mpn"	"VNA520E1"
"optical zoom"	"23"
"optical zoom range"	"18x and higher"
"<page title>"	"Nikon Coolpix S6800 Price in India with Offers, Reviews & Full Specifications | PriceDekho.com"
"color"	"Black",
"amazon"	"Infibeam Ebay Homeshop18 Snapdeal Flipkart"
"digital zoom"	"4x"
"bangalore"	"Hyderabad Chennai Mumbai Delhi Pune"
"approx resolution"	"16 MP"
"external memory"	"Yes"
"face detection"	"NA"
"gps"	"NA"
"hdmi"	"NA"
"maximum shutter speed"	"1/2000 sec"
"metering"	"NA"
"minimum shutter speed"	"1 sec"
"optical zoom"	"18x"
"screen size"	"3 Inches"
"usb"	"Yes",
"video display resolution"	"NA"
"wifi"	"Yes; Wi-Fi 802.11 b/g/n"

have a corresponding property in other sources and is therefore useless for entity resolution. For each source we therefore determine the k (<= 10) most frequent properties.

We also perform data cleaning to harmonize property values to make similarity computations more meaningful. For example, we can see that different units are used for "weight" in different sources. Comparing values in ounces with values in grams would lead to a poor similarity value and we therefore transform both into the same unit. Further data cleaning procedures are performed, such as lowercasing strings and using canonical abbreviations.

Incremental schema matching Schema matching or schema alignment consists of determining which properties of different sources correspond with each other. There are a plethora of different approaches like e.g., instances-based or linguistic matchers that try to tackle this problem (see [6] and [1] for an overview). FAMER currently expects to be provided with already matched properties for entity resolution. For the DI2KG challenge, we however need to first align the properties before we can apply our entity resolution approach.

Our approach makes use of the provided training data for entity resolution task that includes a subset of the true matching entity pairs. While the provided training data contains example entities for all sources, entity matches are only available for a subset of source combinations. For example we are given entity matches for "canon-europe.com" and "price-hunt.com", but not for the combination of "canon-europe.com" and "ebay.com". However we have matches for "price-hunt.com" and "ebay.com". We can therefore first align the properties of "ebay.com" and "price-hunt.com" and then integrate "canon-europe.com" into this intermediary result using the given entity match between "canon-europe.com" and "price-hunt.com".

We therefore follow an incremental property clustering approach that starts with the pair of sources with the most matches in the training data and consider the further sources for property matching in the order of their number of matches. For each source s we thus use the training data to count the number of entities that have s as provenance. We will refer to this as the *provCount* of s. Assuming that the source with the highest *provCount* has already been integrated into the KG, we start property matching with the source that has the second highest *provCount* and continue with the further sources in descending order of their *provCount* until all sources are processed.

Each incremental step consists of the following procedure:

(1) Categorize properties by value range
(2) Calculate property similarities by computing the weighted arithmetic mean of property name similarity and aggregated property value similarity
(3) Update similarity graph
(4) Cluster properties.

To avoid comparing apples with oranges all properties are first categorized by looking at the property value range. Possible categories are for example "string", "number" or "boolean". Looking at Table 1 for example the properties "optical zoom" and "color" clearly belong to different categories since the former mainly has number values and the latter consists of strings.

In the next step we calculate a combined similarity between properties of a new source and already considered properties of previous sources of the same category. The similarity between two properties is based on the similarity of property names and the aggregated similarity of all property values. The property values are derived from all relevant matches for the considered sources from the training data.

The calculated similarites are used to build and update a similarity graph consisting of the properties as vertices and the similarities as edges. This graph is given to FAMER's *clustering* module to determine new property clusters. This is iteratively done until no more sources are left to integrate. The resulting property clusters can now be used in the entity resolution step by fusing all members of a cluster to a new property.

Table 2: Example Data after Preprocessing and Property Alignment

property	value
page title	nikon coolpix s6800 digital black
manufacturer	nikon
resolution	16 mp
color	black
optical zoom	18x
screen size	3.0 inch
page title	nikon coolpix s6800
resolution	16 mp
color	black
optical zoom	18x
digital zoom	4x
screen size	3.0 inch

4 ENTITY RESOLUTION

FAMER assumes the knowledge of matching properties for both blocking and pair-wise linking. We therefore use the scheme matching result and data cleaning for the most frequent properties to harmonize the entities before entity resolution. Table 2 indicates the improved data of Table 1 after preprocessing and property alignment. As illustrated we consider only a subset of the properties and both the property names and some property values have been harmonized.

FAMER provides many options to perform entity resolution for the prepared dataset and we aim at a comparative evaluation of several configurations. In particular, we can apply a batch-like (static) matching and clustering for all (24) sources at once or we can apply an incremental approach that iteratively adds and matches one source after the other. In both cases, we first deduplicate individual sources (by using the linking and clustering of FAMER) before we perform entity resolution between sources and we apply the CLIP approach for clustering (either in its static or incremental version).

Blocking is done on the *manufacturer* property if needed for a sufficiently low runtime. The camera products lacking the value of *manufacturer* form a special block and are matched with all other entities.

Table 3: Results for schema matching on top 10 properties

(a) Quality

Measure	Value
F-measure	0.686
Precision	0.615
Recall	0.775

(b) Statistics about gold standard. Only contains non-singleton clusters.

Measure	Value
# clusters	32
biggest cluster	24

For determining the entity similarity in the linking phase we consider several configurations with different use of property values and with the optional use of word embeddings:

- We perform linking with a single property *page title* that is existing for all entities of all sources and mostly summarizes important product information as illustrated in Table 2.
- We concatenate the values of the top *k* most frequent properties (or a subset of them) into a single document-like value and use the document similarities for linking.
- We determine and combine the pairwise property similarities for a subset of the most frequent properties that are present in most sources. To weight and combine the individual similarities we use either manually determined rules or a learning-based approaches that utilize the provided training data.

After evaluating the mentioned methods and configurations, we will report the results in terms of the entity resolution quality and efficiency.

5 PRELIMINARY RESULTS

A comprehensive evaluation of the different approaches and configuration was not possible since the golden truth has not been made available by the organizers of the DI2KG challenge beforehand. We therefore report some preliminary results for a small subset of the data.

5.1 Schema Matching

For the evaluation of our incremental schema matching we manually created a gold standard, which only includes the top 10 properties for each source. The biggest cluster contains the attribute `<page title>` which is present in all sources. The second largest cluster has a size of 19 and contains attributes which describe the resolution of the camera. The results of our schema alignment and some more information about the gold standard are shown in Table 3.

We can see that even for the most frequent properties this is a very difficult task, due to the heterogeneity of these datasets. Attributes might have the same attribute name, but contain different information and are not to be matched. E.g. The attribute `resolution` in *www.priceme.co.nz* contains a technical description about what resolution, while in all other datasets this attribute contains the number of megapixel of the camera. Another problem lies in distinguishing attributes with similar value ranges. For example properties that have a value range that simply consists of numbers are very similar to each other.

The obtained results are not yet of sufficient quality indicating that inferring property matches from given entity matches is not as effective for the given dataset as we had hoped for. To obtain better and more complete results for all properties we have started to develop a more general solution for property matching. This approach will use machine learning on numerous features derived from property values. The initial results look promising and the approach will be described in an upcoming paper.

Table 4: Selected single-source linking results, th = 0.6

source	#entities	#links	FP	FN	#GTMatches
walmart.com	195	154	0	0	3
priceme.co.nz	740	929	0	0	15
gosale.com	1002	3842	0	0	1
flipkart.com	157	216	0	0	1
eglobalcentral.co.uk	571	2678	0	0	1
ebay.com	14274	420623	3	1	17
camerafarm.com.au	120	138	0	0	1

Table 5: Results for single-source entity clustering on ebay

clustering algo	FP	FN	#pairs	#clusters	CS_{max}
CC	81	1	9268633	3270	2930
CCP	2	4	274236	4185	309
Cntr	1	6	111926	4738	134
MCntr	81	1	9112586	3382	2898

5.2 Entity Matching

We report some initial results based on manual match rules with several matching properties. We thus do not utilize the training data and use the small set of 244 entity matches (#GTMatches) in the provided training data as a subset of the golden truth for evaluation. In particular, we count the number of false positives (FP) and false negatives (FN) regarding the subset of the golden truth. If a computed link connects entities from different perfect clusters, it is counted as 1 false positive. Reversely, if two entities from the same perfect cluster are not matched, it is counted as 1 false negative. For clustering, we specify the number of determined output clusters, the maximum cluster sizes CS_{max} and the number of matching entity pairs represented by the determined clusters.

As explained, we first deduplicate each source and then match the entities of the deduplicated sources. We apply standard blocking on the initial 3 letters of *manufacturer* and determine the similarity of entity pairs based on the similarities of *page titles* and the combination of the values of the properties *resolution*, *manufacturer*, *screen size* and *camera type*. Table 4 reports the obtained results for similarity threshold *th = 0.6* exclusively for the sources containing perfect matches based on the available golden truth indicating that most matches from the golden truth are found for ebay.com.

After linking, several algorithms on the linking output can be applied for single-source clustering. In Table 5 we report some results for the biggest source *ebay.com* and the clustering schemes Connected components (CC), Correlation Clustering (CCP), Center (Cntr) and Merge center (MCntr). Connected components and Merge center create many false positives while achieving the lowest number of false negatives. Center creates smaller clusters compared to Correlation clustering.

Table 6: Linking and clustering output

input	#entities	#links	FP	FN	#pairs	#cluster	CS_{max}
CC-0.5	11303	665	1	0	1429	10763	13
CC-0.6	11303	727	0	0	1547	10719	13
CC-0.7	17407	1938	0	20	2980	16472	16
CC-0.8	17466	2137	0	20	3513	16452	16
CC-0.9	17922	1854	0	21	2957	17005	16
CCP-0.6	12966	1027	0	4	2402	12181	18
Cntr-0.6	13578	1212	0	6	2973	12705	19
MCntr-0.6	11766	780	2	1	1843	11136	16

After clustering each source, the entities inside a cluster are fused into one entity and then all entities from different sources are linked with each other. For linking entities across sources we used the same linking configuration as for single sources and then we applied CLIP [9] for entity resolution across sources. The results of linking and clustering are shown in the left and right part of Table 6 respectively.

The results of linking indicates that when single sources are deduplicated by Connected components and lower thresholds or by Merge center algorithm, the total number of entities is lower, because many entities are fused as one single entity. Moreover in the same cases the number of false negatives is very low. On the other hand for Connected components and higher thresholds, there is a big number of false negatives. As a result, the approach for single-source deduplication has a substantial impact on the overall match quality which deserves further investigation.

ACKNOWLEDGMENTS

This work is partially funded by the German Federal Ministry of Education and Research under grant `BMBF 01IS18026B`. Some computations have been done with resources of Leipzig University Computing Centre.

REFERENCES

[1] Jérôme Euzenat and Pavel Shvaiko. 2013. *Ontology matching* (2nd ed.). Springer-Verlag, Heidelberg (DE).

[2] Lars Kolb, Andreas Thor, and Erhard Rahm. 2012. Dedoop: Efficient deduplication with Hadoop. *PVLDB* 5, 12 (2012).

[3] Hanna Köpcke, Andreas Thor, Stefan Thomas, and Erhard Rahm. 2012. Tailoring Entity Resolution for Matching Product Offers. In *Proc. EDBT*.

[4] Sidharth Mudgal, Han Li, Theodoros Rekatsinas, AnHai Doan, Youngchoon Park, Ganesh Krishnan, Rohit Deep, Esteban Arcaute, and Vijay Raghavendra. 2018. Deep learning for entity matching: A design space exploration. In *Proc. SIGMOD*. ACM, 19–34.

[5] Erhard Rahm. 2016. The case for holistic data integration. In *Proc. ADBIS*. Springer, 11–27.

[6] Erhard Rahm and Philip A. Bernstein. 2001. A Survey of Approaches to Automatic Schema Matching. *The VLDB Journal* 10, 4 (Dec. 2001), 334–350. https://doi.org/10.1007/s007780100057

[7] Alieh Saeedi, Markus Nentwig, Eric Peukert, and Erhard Rahm. 2018. Scalable matching and clustering of entities with FAMER. *Complex Systems Informatics and Modeling Quarterly* 16 (2018), 61–83.

[8] Alieh Saeedi, Eric Peukert, and Erhard Rahm. 2017. Comparative evaluation of distributed clustering schemes for multi-source entity resolution. In *European Conference on Advances in Databases and Information Systems*. Springer, 278–293.

[9] Alieh Saeedi, Eric Peukert, and Erhard Rahm. 2018. Using link features for entity clustering in knowledge graphs. In *European Semantic Web Conference*. Springer, 576–592.

[10] Alieh Saeedi, Eric Peukert, and Erhard Rahm. 2019. Incremental multi-source entity resolution with FAMER. submitted for publication.