Proceedings of the Federated Conference on Computer Science and Information Systems pp. 153–160
DOI: 10.15439/2016F562
ACSIS, Vol. 8. ISSN 2300-5963

iQbees: Interactive Query-by-example Entity Search in Semantic Knowledge Graphs

Marcin Sydow
Grzegorz Sobczak
Institute of Computer Science,
Polish Academy of Sciences, Warsaw, Poland
msyd@ipipan.edu.pl
grzegorz.sobczak0@gmail.com

Ralf Schenkel
University of Passau, Germany
rschenkel@acm.org

Krzysztof Mioduszewski
Polish-Japanese Academy of
Information Technology, Warsaw, Poland
krzysztof.mioduszewski@pjwstk.edu.pl

***Abstract*—We present IQBEES, a novel prototype system for similar entity search by example on semantic knowledge graphs that is based on the concept of maximal aspects. The system makes it possible for the user to provide positive and negative relevance feedback to iteratively refine the information need. The maximal aspects model supports diversity-aware results.**

***Index Terms*—entity search; relevance feedback; interactive retrieval; aspect model; diversity**

I. Introduction

IN THIS paper we present *iQbees*, a prototype system for *Interactive Query-By-Example Entity Search* on semantic knowledge graphs. The system solves the following list completion problem: given one or more example entities (e.g. actors, movies, etc.) as representatives of a group of entities, find other entities in that group. Our system solves this task through a sequence of *interactive query refinement steps* based on positive or negative user feedback. The results in each step are selected according to our own model based on *maximal aspects* that supports diversity awareness of the results.

The target of our interactive prototype *iQbees* system are non-expert users searching for a well-defined group of entities, for example European scientists who won a Nobel prize in physics. We assume that the users are unaware of the structure of the knowledge base and are not familiar with powerful structured query languages like SPARQL, thus cannot formulate a precise query that would satisfy their need. For such users, it is much simpler to provide one or a few examples of entities within the target group; in our example, such entities could be Maria Skłodowska-Curie or Max Planck. Our system will then, in a sequence of steps, present candidate result entities, for some of which the user will be able to give positive or negative feedback, refining the initial query and moving closer to her search goal. Our system thus combines techniques from entity list completion and relevance feedback and interactive search. The *iQbees* is built on the model previously applied in our (off-line) QBEES system [1], equipped with the interactive approach for providing positive and negative feedback by the user.

The first author is also with Polish-Japanese Academy of Information Technology, Warsaw, Poland

An obvious issue in such a scenario is the *ambiguity* of the actual user information need that is imperfectly represented by the given set of example entities. This problem is particularly difficult when just a single example entity is given which may represent a large number of possible entity sets; more example entities make this somewhat easier, but still far from trivial to solve. In such a scenario, it is almost impossible to immediately retrieve the correct set of entities in a single step, as standard list completion systems would do. The QBEES system, which is used as the backbone of our interactive *iQbees* system, does not provide user interaction and thus, as other list completion systems too, does not always retrieve the results that match the user information need.

We model possible user information needs by the concept of *aspects* of an entity, i.e., subsets of the facts and types in the semantic knowledge graph that characterize the entity. This model is designed so that it naturally supports *diversity* of the retrieved entities in order to cover as many as possible potential aspects (and thus possible information needs). A user will therefore likely find at least one relevant entity among the results for which she can give positive feedback, refining her query for the next iteration. The result diversity is guaranteed by the properties of *maximal aspects* that will be shortly described later. The *iQbees* system relaxes the strict aspect-based retrieval model used in QBEES, focusing more on popular entities that a user may know instead of obscure entites that perfectly match a possible information need.

iQbees and its underlying rankers are based on the structural and statistical properties of the underlying semantic knowledge graph. Such approach is orthogonal to any other approaches using external data or textual features of the entities. Hence, our approach can be easily enriched or combined with other text-based approaches that have been havily researched before. Our model assumes the correctness and completeness of the data, treating the issue of missing and wrong facts as out of scope of this paper.

II. The QBEES Aspect-based Model

Our system is based on the *aspect model* that will be very shortly described in this section. The full description of the model can be found in [1].

A. Knowledge graph

A Knowledge Graph KG is a directed multi-graph that consists of three basic components, a *Fact Graph* FG, an *Ontology Tree* O, and a set of type assignment arcs TA connecting the two. Arcs in KG are labelled. We will use the notation `relation(arg1,arg2)` for any directed arc with label `relation` in KG that points from node `arg1` to `arg2`.

The *Fact Graph* $FG = (E, F)$ is a directed multigraph where nodes in E represent *entities* (e.g. `Fryderyk_Chopin`, `Warsaw`) and edges in F represent *facts* about the entities. For example, an arc `bornIn(Fryderyk_Chopin, Żelazowa_Wola)` represents the fact that a Polish composer Fryderyk Chopin was born in Żelazowa Wola or `hasWonPrize(Max_Planck,Nobel_Prize_in_Physics)` means that the German scientist Max Planck is a Nobel Prize awardee in Physics.

The *Ontology Tree* $O = (C, S)$ is a graph where each node (class) $c \in C$ represents some *type* of entities (e.g. person). The class nodes are connected by directed arcs labelled as `subClassOf`. For instance, `subClassOf (composer,musician)` indicates that every composer is also a musician. Because the only relation present in this ontology tree in our current setting is the `subClassOf` relation, it can be also viewed as just a *taxonomy* of types.

The *Type Assignment* TA is a set of arcs labelled `hasType` which connect entities from the Fact Graph and classes from the Ontology Tree. For example the arc `hasType(Chopin, composer)` means that "Chopin is a composer".

B. Basic aspects

A basic aspect represents an "atomic property" that describes a specific entity q. In our most general setting, we distinguish three types of basic aspects: *fact aspects*, *relational aspects* and *type aspects*.[1]

1) Fact aspects: are created from the arcs incident with the entity in the fact graph FG that represent a fact concerning this entity. For example: the arc `bornIn(Chopin,Poland)` induces the fact aspect `bornIn(.,Poland)`.

2) Relational aspects: are predicates obtained from arcs that represent facts in the fact graph FG concerning the entity but the remaining argument of a factual aspect is replaced by a free variable ?. For example, `actedIn(.,?)` indicates that an entity acted in at least one movie.

3) Type aspects: are obtained similarly as relational aspects but inside the type assignment TA set of edges. It is obtained by replacing the particular entity q in an arc that represents a type with a free variable. Intuitively it represents the type of the entity under consideration. For example, a type arc `hasType(Chopin,composer)` naturally induces predicates of the form `hasType(.,composer)` that represents the "basic property" of this entity of "being a composer".

The three kinds of basic aspects can be viewed as forming a 3-level hierarchy, from the most general type aspects to more specific relational aspects (since particular kinds of relations between entities can be bounded only to particular types of entities) to the most specific factual aspects (as being realisations of relational aspects by substituting the free variable with a particular value referring to other entity).

[1]In some particular practical tasks the general framework can be simplified by reducing the considered types of basic aspects to only fact aspects, for example.

C. Compound aspects

A set of basic aspects is called a *compound aspect*. For example, a property "being a composer born in Poland", which consists of two basic aspects - "being a composer" and "being born in Poland", is represented by a set of two basic aspects, i.e. a compound aspect: {`bornIn(.,Poland)`, `hasType(.,composer)`}.

It is easy to see that each entity can be characterised by its set of basic aspects and vice versa – each set of basic aspects represents some set of entities that share it.

For an entity $e \in E$ we will henceforth use A_e to denote a *set of all basic aspects of* e.

We will say that "entity e *satisfies* a compound aspect A" whenever $A \subseteq A_e$.

D. Entity set of an aspect

For each basic aspect $a_q \in A_q$, of some entity q, we naturally define its entity set $E(a_q)$ as the set of all entities $e \in E$ that share this aspect with q, i.e. contain a_q in their set of basic aspects A_e:

$$E(a_q) = \{e \in E : a_q \in A_e\}$$

We extend the above definition of entity set $E(a_q)$ (for a basic aspect a_q) to the concept of entity set $E(A)$ of a compound aspect A as the set of all entities that share *all* basic aspects in A.

E. Maximal aspects

Let q be a query example and A_q be its set of basic aspects. For any $e \in E, e \neq q$ consider the set of all basic aspects of e common with q, that is $A'_e = A_e \cap A_q$.

A compound aspect A_q for entity q is called a *maximal aspect* if and only if it satisfies the following two conditions:

1) it is satisfied by q and *at least one entity other than* q
2) A_q is maximal wrt inclusion (i.e., extending this set of basic aspects with any more basic aspect of q would violate the first condition).

In other words, the maximal aspects of q are maximal compound aspects satisfied by q and any other entity.

We assume that the concept of maximal aspect is defined with respect to the current content of the underlying semantic knowlege graph.

There may exist multiple different maximal aspects for a given entity.

The definition of maximal aspect for a single entity q extends to a set Q of query entities as follows. For a query entity set Q we say that a compound aspect A_Q is *a maximal aspect for* Q if and only if it satisfies the following two conditions:

1) it is satisfied by all entities in Q and at least one entity outside Q
2) A_Q is maximal wrt inclusion

We introduce the denotation of $E(A_Q)$ as the natural extension of the concept of $E(A_q)$ corresponding to a single entity q to the set of entities Q.

To illustrate the maximal aspect concept we will use the following example. Assume an entity set $Q = \{\texttt{Schwarzenegger}, \texttt{Stallone}\}$ is given. Then, consider two compound aspects for Q:

$A_1 = \{$`hasType(.,ActionMovieActor)`, `livesIn(.,USA)`$\}$
$A_2 = \{$`hasType(.,ActionMovieActor)`, `livesIn(.,USA)`, `hasType(.,MovieDirector)`$\}$

If, in the underlying knowledge base, there is at least one entity that is an action movie actor and director living in USA, other than Schwarzenegger and Stallone the first condition of the maximality definition holds. Moreover, if adding any other basic aspect of Q (e.g. additionally being a body-builder) would make the compound aspect satisfied only by the two mentioned entities, the set A_2 is a maximal aspect for Q while A_1 is obviously not since it subsumes A_2.

F. Diversity-awareness of Maximal Aspects

Maximal aspects have the following two crucial properties:

- (RELEVANCE) any entity that satisfies a maximal aspect is *maximally similar* (in terms of sharing maximally many basic aspects with the target entities)
- (DIVERSITY) each maximal aspect A_Q represents a *different* set of entities, i.e. they do not intersect except Q.

The last property can be expressed in the form of the following theorem:

Theorem: *Let Q be a (query) set of entities and $A_Q \neq B_Q$ be two different (non-empty) maximal aspects of Q. Then, $E(A_Q)$ and $E(B_Q)$ do not share any entities, except those in Q (i.e. $(E(A_Q) \cap E(B_Q) \setminus Q) = \emptyset$).*

Proof: *Assume, $e \in E(A_Q) \cap E(B_Q)$ for some entity $e \notin Q$. This implies that e shares all the basic aspects from both A_Q and B_Q. Let us introduce denotation $C_Q = A_Q \cup B_Q$. Thus, e shares all basic aspects from C_Q which implies that $e \in E(C_Q)$. But, since A_Q and B_Q are different and non-empty, C_Q strictly contains both A_Q and B_Q which would contradict the maximality property of them.*

Due to the above theorem, the set of candidate similar entities to be returned is *partitioned* by the entity sets of maximal aspects. Thus, the maximal aspects model supports *diversity* of the results, since they are non-redundant (i.e. different maximal aspects imply non-intersecting sets of entities).

G. Searching with basic QBEES approach

Given a set of query examples, the approach calculates all maximal aspects and generates results from them based on a two-step ranking scheme that first selects the most promising maximal aspect and then the most promising entity that satisfies this maximal aspect. This process is repeated until k results are retrieved. Entities are ranked based on various factors including graph properties (e.g. random-walk based) and importance (popularity). Maximal aspects are ranked based on their size and the popularity of their entities, hence retrieving (and thereby removing) an entity from a maximal aspect changes its score. We adapted the QBEES ranking system that is out of scope of this paper and is described in [1].

III. Interactive Search with *iQbees*

With *iQbees*, the static search procedure used by QBEES is extended with interactive feedback cycles.

Initially, the user provides an example entity.[2]. The system then returns an initial list of result entities using the ranking approach. If this answer is not yet perfect, the user can use the returned entities as *refinement suggestions* and select some of them as *positive* hints, some of them as *negative* hints. The system will then exploit this feedback and produce new, hopefully better, results. This interaction cycle proceeds until the user is satisfied with the results.

A. Positive feedback and Negative feedback

To shortly explain the mechanism of positive and negative feedback, let's introduce some ancillary concepts. Let *domain* be defined as the intersection of compound aspects of all initial entities and all positive feedback entities. Let *candidate aspects* be defined as the set of compound aspects that are contained in the *domain*. In the basic setting they are exactly *maximal aspects* contained in the *domain*. In the *relaxed* variant, that will be described below, *candidate aspects* are any compound aspects contained in the *domain*.

Filtered candidates are exactly those *candidate aspects* that are *not satisfied* by any entity from the negative hint set.

Then, we run the algorithm using *filtered candidates* only, i.e. only the entities that satisfy filtered candidates can be returned.

B. Relaxed Variant of Aspect Selection

The maximal aspect concept proved to perform well in the basic QBEES algorithm [1] where the user provided a set of entities in a "one-shot" mode.

However, in the interactive *iQbees* approach, if we applied the basic model described above, one can observe the following phenomenon. The more positive feedback entities are provided by the user, the more entities can be potentially returned which is counter-intuitive to the concept of *query refinement*. This phenomenon is due to the fact, that the more positive examples are provided, the smaller is their aspect intersection (*domain*). As a result, more entities can potentially satisfy such maximal aspect being the intersection of more entities, since it is easier to contain a smaller aspect set than a bigger one. This is counter-intuitive, since providing more positive examples should *refine* the query intent rather than make it more vague.

[2]Actually, our model can be easily extended to allow for any number of initial query entities. This can be also easily simulated by providing one entity and marking the others as positive feedback. The extension to multi-entity input is planned as a close future work

Hence, to "correct" such undesirable behaviour, in *iQbees* we "relax" the original QBEES mechanism by considering (as *candidates*) *all* compound aspects contained in the *domain* instead of considering only *maximal aspects* contained in the *domain* as *candidates*. Such "relaxed", non-maximal compound aspects are ranked using the base QBEES approach to promote the maximal or near-maximal aspects and only top-k are considered. After this "relaxation" the above phenomenon is removed, i.e. the more positive hints are provided, the fewer entities potentially satisfy (contain) any relaxed compound aspect.

IV. An example search session

A working demo application was built as a preliminary proof of concept, and as an experimental platform and to illustrate the approach studied in this paper. It is currently using YAGO [2] as the underlying semantic knowledge base[3]. By using the demo application a user, interested in a particular entity present in the semantic knowledge graph, can query the graph to find similar entities. *User Interface UI* of the demo application allows to mark each search result (entity) as relevant or not. This information is used in subsequent queries and allows the user to iteratively refine his initial query until he receives satisfactory results.

A. Preliminary Demo UI

Demo application UI consists of three main parts - *Query Input Field*, *Search Parameters Section* and *Search Results Section*. *Query Input Field* is a typical text input, where the user can type in the entity name which will be the subject of the query. *Search Parameters Section* contains UI controls which allow to change the number of expected search results and specific settings of the query algorithm.

Search Results Section, visible after performing the initial and subsequent queries, contains similar entities found in the previously executed query and *Feedback Subsection*. Each result entity has two buttons labeled "+" and "-". These buttons are used to mark a particular result entity as positive or negative feedback i.e. relevant or not to the initial user information need. Pressing one of these buttons results in moving the entity to the *Feedback Subsection* and marking it accordingly as "positive" or "negative" feedback for future searches. Each feedback entity has an additional "show debug" button which displays the set of all *basic aspects* satisfied by it. Likewise each search result entity has a "show debug" button which displays *maximal aspect* of query subject satisfied by this particular entity.

The "Calculate" button triggers query execution. The "Fresh Start" button restarts the searching session, clears the initial entity and previous search feedback. Screenshots of available "work in progress UI" are visible in Figures 1 and 2.

[3]Other semantic knowledge graphs are considered to be used as the underlying databes in our future work

B. Example usage scenario

Application user is interested in Arnold Schwarzenegger and entities similar to him in terms of political career. Notice the particular *ambiguity* of this entity, since it represents also many other aspects, e.g. famous body-builders, action movie actors, directors, etc. Assume the user applies the default search parameters visible in Figure 1 and inputs "Arnold Schwarzenegger"[4] in *Query Input Field* and clicks the "Calculate" button. The system verifies if "Arnold Schwarzenegger" entity is present in the underlying knowledge graph (demo application requires exact match of the entity name) and starts computing similar entities. After certain amount of time the results are visible in *Search Results Section*. In the results, among others, the user can see:

- Gray Davis: a former governor of California.
- Dany DeVito: an American actor who acted in a move "Junior" with Schwarzenegger.

User marks Gray Davies as positive feedback and clicks "Calculate" once again. The new result set contains only politicians. The user can further search for similar entities by providing new feedback.

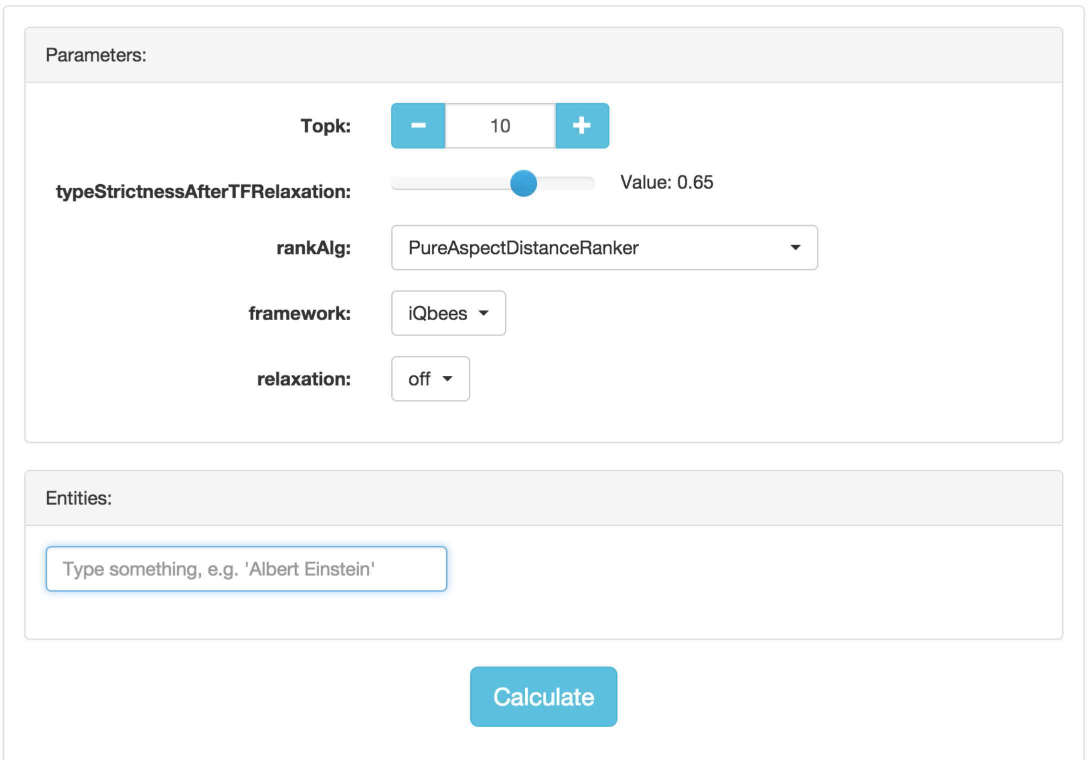
If user eventually decides that entities similar to Arnold Schwarzenegger in terms of political career are not interesting to her, she can start a new searching session by clicking the "Fresh Start" button. This will clear previous search results and return *Search Parameters Section* settings to default values. User once more inputs "Arnold Schwarzenegger" in *Query Input Field* and *Search Results Section* gets populated by previously visible results. This time user marks Gray Davies as negative entity by clicking the "-" button next to it. Once the search is finished a new result set contains mainly actors and movie producers and does not contain any politicians. In the next step user chooses Sandahl Bergman as a positive entity. Sandahl Bergman is an actress who starred in "Conan the Barbarian" and "Red Sonya" together with Schwarzenegger. The final result set contains actors and actresses among whom three actors played in "Conan the Barbarian" (Gerry Lopez, James Earl Jones) and one actress starred in "Red Sonya" (Brigitte Nielsen).

V. Experiments

The prototype system has been implemented and is being tested. Besides experimenting with the on-line prototype, we made a preliminary experimental evaluation aiming at an objective comparison of the described variants of our system on publicly available benchmark data.

We used the YAGO semantic knowledge graph [2] as the underlying database. We used data from the INEX 2007 entity track to build a 163-query one-entity gold-standard dataset by mapping Wikipedia pages to YAGO and using list completion topics, following the approach sketched in [1]. Each topic

[4]In our current working demo the input is not preprocessed so that the user has to type the exact string as it is represented in the database. Obviously, in future we plan to add semi-automatic query correction using methods proposed in [3] or [4] to make the user interface more user-friendly


Parameters:

Topk: 10

typeStrictnessAfterTFRelaxation: Value: 0.65

rankAlg: PureAspectDistanceRanker

framework: iQbees

relaxation: off

Entities:

Type something, e.g. 'Albert Einstein'

Fig. 1. Initial application view with default search parameters.

is provided with a set of relevant entities (called *ground truth*). As the basic back-end, we reused the existing QBEES engine that was extended by positive and negative feedback functionality and ranking all the candidate aspects (not only maximal aspects).

We simulated three simple 2-step strategies of the user. In each scenario the "simulated" user, in step 0, inputs one query entity to the system and then, in step 1, marks one (randomly) selected result entity that is relevant according to the ground truth as "positive" (p), or one (randomly) selected result entity that is irrelevant according to the ground truth as "negative" (n), or both (p+n). We simulate this by marking 1 *random* relevant or irrelevant entity, respectively.

Each combination of one of the three mentioned strategies (p, n, p+n) and one of the two extensions described above (FEEDBACK for positive and negative feedback, RELAX for additional consideration of non-maximal aspects) was run on each of the 163 one-entity queries. For all settings, we computed average MAP and MNDCG in step 0 and step 1 to measure and compare the performance of the system in each step and each setting. For each query and each setting the procedure of computation of measures is as follows: we run two steps; we remove the selected entieties from the ground truth and from both result list; then we calculate measures (MAP, MNDCG) for each step.

The presented experimental evaluation results indicate that in the two examined settings: marking positive (p) and positive and negative (p+n) entities observably improves the quality of the results in the next step (Tables I and III). Interestingly, we also found out that marking only 1 negative entity did not improve the results in the next step (Table II), in this particular experimental setting even if in our on-line experiments the negative feedback functionality seems to improve the results. This issue will be the matter of our further study.

Furthermore, one can observe that the relaxed variant of our approach (i.e. RELAX) performs consistently better than FEEDBACK in this setting.

VI. Previous and Related Work

The system is based on the *aspect model* described in detail in [1]. An early version of the *iQbees* system (without negative feedback functionality nor relaxation nor experimental evaluation) was presented in a short paper [5] that this paper is a substantial extension of.

Entity search has been considered extensively in the past, with a focus on finding related entities and list completion, and with extensive evaluation campaigns at TREC [6] and INEX [7]. We consider the specific scenario where entities

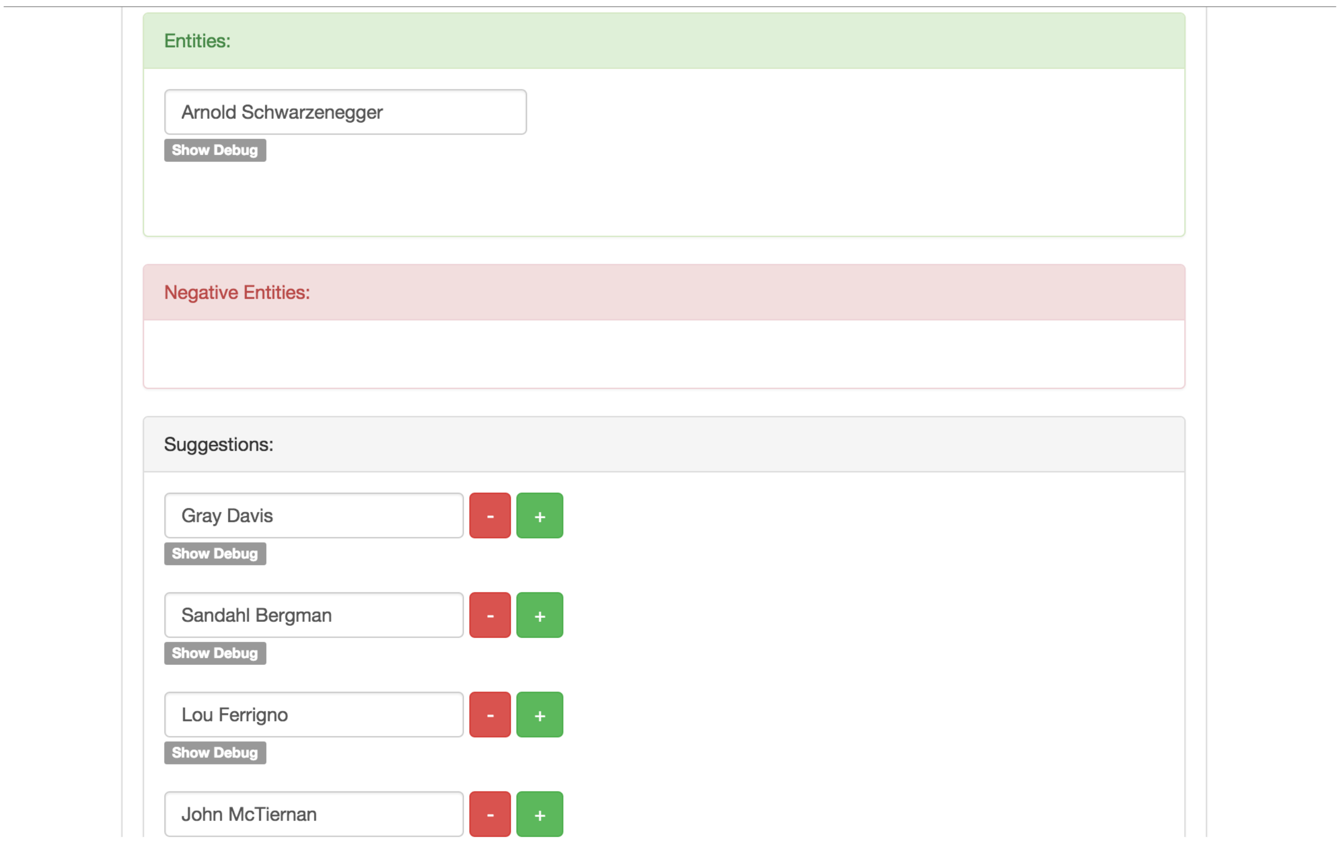


Fig. 2. Application view with search results visible.

TABLE I
THE RESULTS OF THE EXECUTED EXPERIMENTS FOR THE 1-POSITIVE HINT STRATEGY

Model	map		mndcg	
	step 0	step 1	step 0	step 1
FEEDBACK	0.0222	0.0404	0.0715	0.0941
RELAX	0.0386	0.0801	0.1015	0.1617

TABLE II
THE RESULTS OF THE EXECUTED EXPERIMENTS FOR THE 1-NEGATIVE HINT STRATEGY

Model	map		mndcg	
	step 0	step 1	step 0	step 1
FEEDBACK	0.0485	0.0441	0.1284	0.1173
RELAX	0.0650	0.0615	0.1563	0.1471

TABLE III
THE RESULTS OF THE EXECUTED EXPERIMENTS FOR THE MIXED (1-POSITIVE AND 1-NEGATIVE) STRATEGY

Model	map		mndcg	
	step 0	step 1	step 0	step 1
FEEDBACK	0.0212	0.0400	0.0679	0.0920
RELAX	0.0396	0.0567	0.0997	0.1289

from a knowledge graph are searched. Existing systems usually build on entity similarity measures, exploiting the graph structure (e.g., SimRank [8]), the context of entities in the graph (e.g., Albertoni and De Marino [9]), or additional context outside the graph (e.g., Bron et al. [10], which combines a term-based language model with a simple structural model).

The problem of example-based entity search has been actively studied recently. Yu et al. [11] solve a slightly different problem where entities similar to a single query entity are computed, exploiting a small number of example results. Focusing on heterogeneous similarity aspects, they propose to use features based on so-called meta paths between entities and several path-based similarity measures, and apply learning-to-rank methods for which they require labelled test data. Wang and Cohen [12] present a set completion system retrieving candidate documents via keyword queries based on the entity examples. Using an extraction system additional entities are then extracted from semi-structured elements, like HTML-formatted lists.

Mottin et al. [13], [14] introduce the concept of exemplar queries. Similar to our setting, an example result is used instead of a query. However, the setting in their XQ system is strictly different since it considers examples in the form of a connected subgraph of entities, not single entities, and determines result subgraphs based on their similarity to the query graph. The problem is therefore in some sense easier, as more information can be exploited for identifying query results.

The GQBE system by Jayaram et al. [15] is similar to XQ, but does not use connected subgraphs, but just entities that form a query result as input; the meaningful connections between those entities are explored by the system. Again, the main difference to our system is that we consider only single entities as results, not combinations, and hence have less information for identifying relevant results.

Relevance feedback has seen surprisingly little use in entity search on knowledge bases. Only very recently, Su et al. [16] proposed exploiting relevance feedback for improving results of searching a knowledge graph, but not for entity search. For entity ranking using text or semi-structured information, relevance feedback has been more popular [17].

Diversity-aware entity summarization was originally proposed in [18] and further studied in [19], but we are not aware of any work on entity search that takes diversity into account.

VII. Conclusions and Further Work

We presented *iQbees* – a prototype interactive approach to the problem of entity list completion based on semantic knowledge graphs. This is an extension of the previously presented QBEES system [1] by adding positive and negative interactive feedback functionality. The prototype of the approach was preliminarily implemented as a proof of concept and demonstration available online. We also presented experimental results that indicate that the proposed approach outperforms on a publicly available benchmark the basic (non-interactive) QBEES system without the feedback functionality. The future work includes better understanding and modeling of the feedback functionality, in particular negative feedback, and ranking strategies since the current experiments indicate that there is room for improvement in the current model. It is also envisaged to further work on developing the on-line demo in order to improve its functionality and optimise it and to perform more experimental evaluation on other semantic knowledge graphs, e.g. DBPedia.

VIII. Acknowledgments

This work is partially supported by the Polish National Science Centre grant 2012/07/B/ST6/01239. Grzegorz Sobczak was partially supported by the European Union under the European Social Fund Project PO KL "Information technologies: Research and their interdisciplinary applications", Agreement UDA-POKL.04.01.01-00-051/10-00 when working on this project during his research stay in the Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland. The authors would like to thank Steffen Metzger, who provided the substantial part of the code of the back-end of the earlier version of the system.

References

[1] S. Metzger, R. Schenkel, and M. Sydow, "Aspect-based similar entity search in semantic knowledge graphs with diversity-awareness and relaxation," in *WI-IAT*, pp. 60–69. [Online]. Available: http://dx.doi.org/10.1109/WI-IAT.2014.17
[2] Http://www.mpi-inf.mpg.de/yago-naga/yago.
[3] J. Piskorski and M. Sydow, *String Distance Metrics for Reference Matching and Search Query Correction*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 353–365. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-72035-5_27
[4] J. Piskorski, K. Wieloch, and M. Sydow, "On knowledge-poor methods for person name matching and lemmatization for highly inflectional languages," *Information Retrieval*, vol. 12, no. 3, pp. 275–299, 2009.
[5] G. Sobczak, M. Chochół, R. Schenkel, and M. Sydow, "iQbees: Towards interactive semantic entity search based on maximal aspects," in *Foundations of Intelligent Systems*. Springer International Publishing, 2015, vol. 9384, pp. 259–264, 10.1007/978-3-319-25252-0-28. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-25252-0_28
[6] K. Balog, P. Serdyukov, and A. P. de Vries, "Overview of the TREC 2011 entity track," in *TREC*, 2011.
[7] G. Demartini, T. Iofciu, and A. P. de Vries, "Overview of the INEX 2009 entity ranking track," in *INEX*, 2009, pp. 254–264.
[8] G. Jeh and J. Widom, "SimRank: a measure of structural-context similarity," in *KDD*, 2002, pp. 538–543. [Online]. Available: http://doi.acm.org/10.1145/775047.775126
[9] R. Albertoni and M. D. Martino, "Asymmetric and context-dependent semantic similarity among ontology instances," *J. Data Semantics*, vol. 10, pp. 1–30, 2008. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-77688-8_1
[10] M. Bron, K. Balog, and M. de Rijke, "Example based entity search in the web of data," in *ECIR*, 2013, pp. 392–403. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-36973-5_33
[11] X. Yu, Y. Sun, B. Norick, T. Mao, and J. Han, "User guided entity similarity search using meta-path selection in heterogeneous information networks," in *CIKM*, 2012, pp. 2025–2029. [Online]. Available: http://doi.acm.org/10.1145/2396761.2398565
[12] R. C. Wang and W. W. Cohen, "Language-independent set expansion of named entities using the web," in *ICDM*, 2007, pp. 342–350. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/ICDM.2007.104
[13] D. Mottin, M. Lissandrini, Y. Velegrakis, and T. Palpanas, "Exemplar queries: Give me an example of what you need," *PVLDB*, vol. 7, no. 5, pp. 365–376, 2014. [Online]. Available: http://dx.doi.org/10.14778/2732269.2732273
[14] ——, "Searching with XQ: the exemplar query search engine," in *SIGMOD*, 2014, pp. 901–904. [Online]. Available: http://doi.acm.org/10.1145/2588555.2594529

[15] N. Jayaram, A. Khan, C. Li, X. Yan, and R. Elmasri, "Querying knowledge graphs by example entity tuples," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 10, pp. 2797–2811, 2015. [Online]. Available: http://doi.ieeecomputersociety.org/10.1109/TKDE.2015.2426696

[16] Y. Su, S. Yang, H. Sun, M. Srivatsa, S. Kase, M. Vanni, and X. Yan, "Exploiting relevance feedback in knowledge graph search," in *KDD*, 2015, pp. 1135–1144. [Online]. Available: http://doi.acm.org/10.1145/2783258.2783320

[17] T. Iofciu, G. Demartini, N. Craswell, and A. P. de Vries, "Refer: Effective relevance feedback for entity ranking," in *ECIR*, 2011, pp. 264–276. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-20161-5_26

[18] M. Sydow, M. Pikula, and R. Schenkel, "Diversum: Towards diversified summarisation of entities in knowledge graphs," *2014 IEEE 30th International Conference on Data Engineering Workshops*, vol. 0, pp. 221–226, 2010.

[19] M. Sydow, M. Pikuła, and R. Schenkel, "The notion of diversity in graphical entity summarisation on semantic knowledge graphs," *Journal of Intelligent Information Systems*, vol. 41, pp. 109–149, 2013. [Online]. Available: http://dx.doi.org/10.1007/s10844-013-0239-6