Exploiting ERP Systems in Enterprise Search

Diego Tosato

Eurosystem S.p.a.,Via Newton 21, Villorba (Treviso), Italy
diego.tosato@eurosystem.it

Abstract. Enterprise resource planning (ERP) systems are the core of many companies: they contain entities which are the focus of enterprise searches [5]. In this paper, a model which exploits those entities to improve the search experience of enterprise users is proposed. Specifically, a graph knowledge base called *entity graph* is defined. It is used both to offer a novel data exploration experience that reflects the business processes and to improve the search accuracy contributing to the score of a search result into a weighted linear model. The applicability of the model is demonstrated by implementing an enterprise search prototype called SeNSE (Skyline eNterprise Search Engine).

Keywords: Enterprise search, Entity centric retrieval, Entity graph exploration

1 Introduction

According to [5], enterprise search on small data is much more important than web search on big data for many companies, but this issue still receives little attention from the information retrieval community. However, last advances in enterprise search focus on the extraction of concepts or *entities* from enterprise data, which might be a promising way to enhance the search performances. Among the different sources of information of an enterprise (such as relational databases, file system documents, web pages, etc.), a key role is played by ERP systems [10], which are typically composed of several modules, such as sales, finance and production, or business intelligence. Since ERP systems capture information among modules and provide an integrated view of information through enterprise business processes, we decide to model its main entities and the related *entity links*. The latter are arranged in a graph knowledge base that we called *entity graph* (EG), which can be used to boost the search results and to explore data in a way that reflects the business processes and the work-flow of enterprise users. Despite some state-of-the-art enterprise search systems, based on entities [1,5], ours are a small number of complex concepts (such as orders, invoices, estimates, etc.). This choice has two main advantages: from the enterprise user point of view, entities and their links can be displayed as a meaningful graph that can be exploited for the everyday work; from the machine learning point of view, since we have a small number of entities and entity links, it is easier to assign them weighs that can be used to improve search results.

Our contributions are summarized as follows: (1) to our best knowledge, we are the first one to use ERP entities and their relations to build a knowledge base to improve the search experience; (2) we propose a novel data navigation model based on the EG; (3) we build an enterprise search prototype that demonstrate the applicability of our model.

2 System Design

ERP is the core of a company [10] because it contains most of the fundamental entities searched by enterprise users. Despite that, most of the enterprise search solutions are not able to achieve satisfying search performances because they still aim at working at word level. However, there are remarkable recent works that show how to extract concepts or entities from data automatically [1,2,4,5,8], but they still cannot deal with complex ERP entities made up of many relational tables. To improve the search experience, we decided to model the fundamental entities and their relations explicitly by exploiting our knowledge of ERP systems and enterprise user needs, which is necessary to build an effective search system [13]. Therefore, we asked our users which were the most relevant type of ERP entities and what kind of relationships connected them. We obtained a list of 33 entity types that are connected by 70 relationship types that represent the core of the work for most of our users. These entities are made of structured and unstructured data that are represented as documents. In order to preserve the structure of the entities, documents are organized as a set of fields (see [7] for more details). Furthermore, according to our ERP domain experts, we defined a set of components, detailed in Sec. 2.1, that must influence the rank of an entity. These are related to the following fundamental aspects of an entity: content, context (in terms of its relationships), and last modified date. To combine the contributions of the components, we follow the idea proposed by [12], which led us to design a modular enterprise search engine. The modules are organized into a pipeline and the contribution of each of them is computed sequentially.

2.1 Entity Ranking

When a search is performed, the final rank of the results is a weighted linear combination of contributions computed by a pipeline of components. More formally, let $\{\alpha_i\}_{i=1,\dots,N}$ a set of scores and $\{w_i\}_{i=1,\dots,N}$ a set of weights, the final rank r of an entity ε is given by

$$r(\varepsilon) = \sum_{i=1}^{N} w_i \bar{\alpha}_i \quad \text{s.t.}\, 0 \leq \bar{\alpha}_i \leq 1, \tag{1}$$

where $\bar{\alpha}_i$ represents the normalized version of α_i through the min-max normalization method [9]. We instantiated the model (Eq. (1)) considering the following contributions:

$\alpha_{\mathbf{cnt}}$ Given the document representation d of entities ε, this is the TF-IDF score that reflects how relevant an entity is by its content (see [6]). More specifically, we computed the cosine similarity between the user query q and an indexed document d represented as vectors. Therefore, the α_{cnt} can be expressed as

$$\alpha_{\text{cnt}}(d) = \cos(q, d) = \frac{V(q)V(d)}{|V(q)||V(d)|}, \quad d \in \mathcal{D}$$

where $V()$ is the vector form of a document and $\mathcal{D}$ is the set of indexed documents.

$\alpha_{\mathbf{dte}}$ It is a linear score that boosts recent entities [7]. Given the date of a document expressed in days t and a normalization constant $n = \max(t \in \mathcal{T})$, where $\mathcal{T}$ represents the set of dates of the indexed documents, the score is defined as

$$\alpha_{\text{dte}}(d) = \gamma \frac{n - t}{n},$$

where γ is a boost factor that we set to 2 and d is the document associated with t.

$\alpha_{\mathbf{egs}}$ Considering the subgraph $\mathcal{S}$ of EG provided by the top results of α_{cnt} ranking, this is a logarithmic score that boosts connected documents. α_{egs} is defined as

$$\alpha_{\text{egs}}(\varepsilon) = \log\left(1 + \frac{1}{\varphi(\varepsilon, \varepsilon')}\right), \quad \varepsilon, \varepsilon' \in \mathcal{S}$$

where $\varphi()$ is a weighted distance computed by summing the weight of the edges on the shortest path between a pair of entities $(\varepsilon, \varepsilon')$ such as $\varepsilon \neq \varepsilon'$.

$\alpha_{\mathbf{prk}}$ The score provided by Page Rank which is proven to lead to better search performances [12].

Therefore the rank model used by our system is

$$r(\varepsilon) = w_1 \bar{\alpha}_{\text{cnt}} + w_2 \bar{\alpha}_{\text{dte}} + w_3 \bar{\alpha}_{\text{egs}} + w_4 \bar{\alpha}_{\text{prk}}, \tag{2}$$

where $w_1, \ldots, w_4$ are assigned by analyzing search results as explained in Sec. 3.3. Exploiting the click-through data [6], it could be interesting to try to compute weights automatically by using a machine learning technique such as SVM, boosting, or neural networks [9].

2.2 Entity Graph (EG)

To meet the user need of exploring ERP entities, we build the EG, enhancing enterprise search with an exploration experience complementary to faceted navigation and full text search. EG is a graph which consists of nodes that represent entities extracted from a set of queries on the data sources, one for each entity

type. Edges represent the underlying business relations among the entity types. They are extracted by queries that link pairs of entity references. Formally, an EG is a directed graph $G = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, where $\mathcal{V}$ is a set of nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ a set of edges, and $\mathcal{W}$ a set of edge weights. We place an entity identifier into each node, while edges contain labels that explain the meaning of the relations. A configuration file determines the queries to extract the relations, their direction, and the weights of each type of relation. By analyzing the links of EG, we found that there are huge node hubs because there are some types of entities (i.e., master data type) that are linked to almost all the others. This is a problem for the computation of α_{prk}, because ranking methods such as PageRank or HITS [3] are built to rank web pages. So, they give higher rank to hub nodes which are not necessarily relevant for each enterprise information need. Even if the problem is still open, our system gives to α_{prk} a lower weight in order to mitigate the huge hub nodes effect.

3 Prototype

SeNSE (Skyline eNterprise Search Engine) is the name of the prototype that demonstrates the applicability of the model described in Sec. 2. The prototype is based on the ERP system Freeway Skyline[1].

3.1 Architecture

Fig. 1 provides an overview of the architecture of our system. SeNSE is designed to search on data coming from any source of information such as file servers, ERP applications, and databases. During the indexing phase made by the indexing server (see Fig. 1), entities are extracted in the form of documents and analyzed, and the security information are computed. Then entity links are extracted, the EG is built and analyzed, finally the preview of the entities is computed. From the time and space complexity point of view, this last operation is the most expensive of the entire indexing process, but it is very useful for the users because it provides some entity details without leaving the SERP (Search Engine Results Page) page. The entities extracted and processed by the Indexing Server are stored in three different repositories: the *Inverted Index* implemented through [7] that contains all the textual information of the entities; the *entity graph database* contains an instance of EG; the *preview database* stores an image preview for each entity. Both the databases are implemented through [11]. In order to guarantee the integrity and the synchronization of the repositories an enterprise service bus (ESB) is adopted.

The searching server provides two search web services, namely the *full text search* and the *faceted search* implemented through Bobo-Browse[2], which are based on the *search pipeline* that contains the following search components:

[1] www.freewayskyline.com
[2] senseidb.github.io/bobo

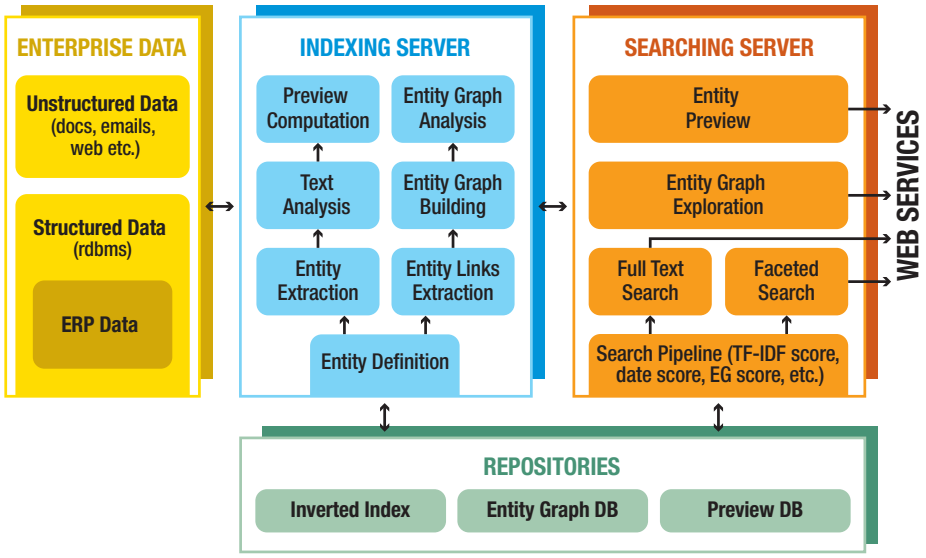

Fig. 1. Overview of the architecture of SeNSE.

Content Search computes the score α_{cnt} by exploiting the full text search capabilities of Lucene [7]; *Date Boost* computes the score α_{dte}; *Link Score* computes the score α_{egs}; *Page Rank* computes the score α_{prk}; *Final Score* computes the equation (2) given the result of the previous components; *Abstract Highlighting* highlights terms of the result documents that match the user query; *Entity Security* defines a cached security filter that is provided by the Content Search component. The searching server provides two other services, namely the *entity graph exploration* and the *entity preview* services independent of the search pipeline. We store into document fields the security information ϕ such as user name, company name, and database table grants. For each ϕ, we define the *allow* a and *deny* d policies. To establish if a result can be listed into the SERP the following boolean expression is evaluated

$$(\phi_a^1 \wedge \neg\phi_d^1) \vee \cdots \vee (\phi_a^i \wedge \neg\phi_d^i) \vee \cdots \vee (\phi_a^L \wedge \neg\phi_d^L),$$

where $i \in 1, \ldots, L$ is the index of a security information. The presence of ϕ_d^i is not strictly necessary, but it allows to implement security roles such as "allow all but ...".

One of the major problems we found in designing the architecture of SeNSE is that it needs different representations of an entity (namely sparse vector, node of a graph, and database entry) to provide its services. This is not only a scalability issue but also a modeling one. In fact, the extension of the search pipeline with further components could introduce novel representations for the entities. In particular, for many machine learning techniques a dense vector representation is necessary [12]. To the best of our knowledge there is not a unified representation to search, analyze and explore entities.

Another tricky problem concerns the update of the indexed entities, because enterprise search engines updates should be processed in near real-time. The system has to deal with all the type of updates, in particular it has to manage the cancellation of entities which is the most difficult case. To tackle the update problem, SeNSE implements three update policies: *batch full* that updates all the entities of a certain type, *batch delta* that updates entities modified up to a specific date, and *real time*. The first two policies can be scheduled depending on the number of entities involved into the update and their indexing speed. The current implementations of the policies is specific for each data source, but there is still room to improve because the performances of the update notification infrastructures provided by data sources are not always satisfactory, since the infrastructures provide too many false positive update notifications or too generic notifications.

3.2 User Experience

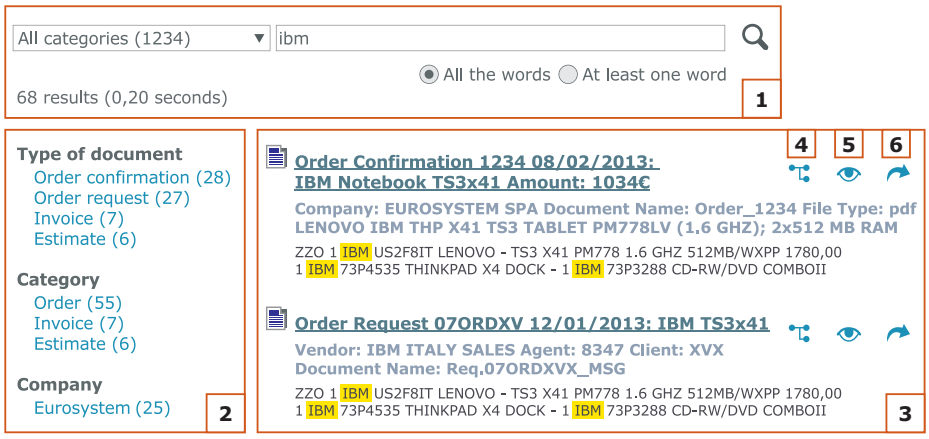

Fig. 2. The SERP page.

The most relevant pieces of the user interface of our system is shown in Fig. 2, Fig. 3, and at www.freewayskyline.com/demosense. In particular, Fig. 2 shows a small part of the SERP which is divided into three main areas. The first area contains the search box as depicted in Fig. 2.1. According to our users, we provide the possibility to choose the type of entity before entering the search query. Once the search is performed, the faceted navigation can be started from the left part of the UI as shown in Fig. 2.2. Simultaneously, the results are listed in the right part of the interface (Fig. 2.3). For each result three functions are available: starting from the left, the first function is the EG exploration (Fig. 2.4) which is detailed below. In the middle (Fig. 2.5), it is placed the preview function that displays the image of the entity associated with the result into a

flexbox according to its type and format. Finally, on the right (Fig. 2.6) there is the *user actions* function that list a set of user defined business actions available for the result such as compile an order or print a bill.

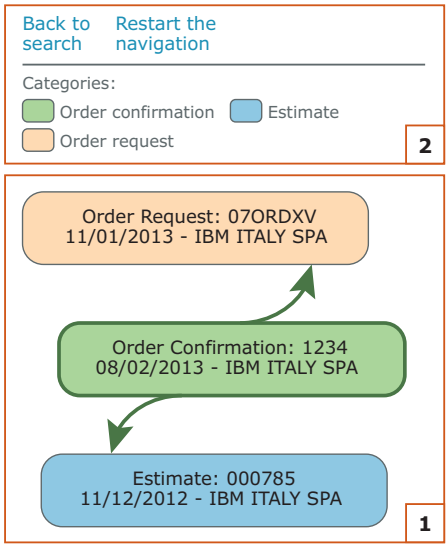

Fig. 3. The EG exploration page.

To implement the exploration of the EG, we use the Vis.js[3] library that is able to display automatically and interact with the graph at the same time. When a result of a search is displayed into the SERP, the exploration can be started from the entity associated with the selected result and its neighborhood as shown in Fig. 3.1, then it is possible to continue the exploration experience by selecting a neighbor. Since the ER is interactive, from each node it is possible to execute its business actions. On the top part of the EG exploration page (Fig. 3.2) the map legend and the main navigation functions are displayed. Users found the EG exploration effective and intuitive on both tablet and pc and ask to personalize the appearance of each entity type.

3.3 Experiments

We experimented SeNSE with success on an X64 Intel Xeon E5450 3.00 Ghz processor with 10 Gbytes of RAM server. Since we are not aware of any public database that fit our ERP entity model, we built three different enterprise datasets with real data. They contain approximately 1 million entities and 10 million entity links which are typical magnitude of data for small and medium-sized enterprises. To evaluate the performance of our system we chose the largest

[3] visjs.org

dataset and we computed the *Precision at k* (P_k) [6] on a testing set of 100 user information needs. We collected the needs both by interviewing users and by logging their search queries, then relevance judgments are obtained by merging the user ranking on the top 5 entities. The performance baseline of SeNSE is given by the α_{cnt} rank. It yields that the top 5 entities in user queries are recognized with an average precision of 54%. To improve the performances up to 15%, we added all the others score components (α_{dte}, α_{egs}, and α_{prk}). We assigned a weight $\{w_i\}_{i=1,\dots,4}$ performing a grid search [9] that maximize P_k. For this purpose all the scores are normalized (see Eq. (1)) and weights are selected by searching into a range $0 \leq w_i \leq 1$ using a step of 0.1. Final weights are not uniformly distributed, in fact α_{cnt} is the most important contribution with respect to the others.

4 Conclusions and Future Works

We presented an enterprise search model that exploits ERP entities to enhance the enterprise search experience and its implementation: SeNSE. We discussed the main design aspects of the model and the related open issues. Then we present the architecture of the prototype and its user experience. In future work, we aim to clarify the benefit given by each contribution to entity ranking and we will implement an automatic method to compute the weights for those contributions.

References

1. Brauer, F., Huber, M., Hackenbroich, G., Leser, U., Naumann, F., Barczynski, W.M.: Graph-based concept identification and disambiguation for enterprise search. In: WWW (2010)
2. Graus, D., Tsagkias, M., Weerkamp, W., Meij, E., de Rijke, M.: Dynamic collective entity representations for entity ranking. In: WSDM (2016)
3. Leskovec, J., Rajaraman, A., Ullman, J.D.: Mining of massive datasets. CUP (2014)
4. Li, J., Yang, J.J., Liu, C., Zhao, Y., Liu, B., Shi, Y.: Exploiting semantic linkages among multiple sources for semantic information retrieval. EIS (2014)
5. Liu, X., Chen, F., Fang, H., Wang, M.: Exploiting entity relationship for query expansion in enterprise search. IR 17(3), 265–294 (2014)
6. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
7. McCandless, M., Hatcher, E., Gospodnetic, O.: Lucene in Action: Covers Apache Lucene 3.0. Manning Publications Co. (2010)
8. Meij, E., Balog, K., Odijk, D.: Entity linking and retrieval for semantic search. In: WSDM (2014)
9. Murphy, K.P.: Machine learning: a probabilistic perspective. MIT press (2012)
10. Nazemi, E., Tarokh, M.J., Djavanshir, G.R.: Erp: a literature survey. IJAMT (2012)
11. Owens, M., Allen, G.: SQLite. Springer (2010)
12. Turney, P.D., Pantel, P., et al.: From frequency to meaning: Vector space models of semantics. JAIR (2010)
13. White, M.: Critical success factors for enterprise search. BIR (2015)