DEESSE: entity-Driven Exploratory and sErendipitous Search SystEm

Olivier Van Laere[1] Ilaria Bordino[2] Yelena Mejova[3] Mounia Lalmas[4]

[{1,2,4}]Yahoo Labs, Barcelona & London [3]Qatar Computing Research Institute, Doha, Qatar

{[1]vanlaere, [2]bordino [4]mounia}@yahoo-inc.com [3]mejova@qf.org.qa

ABSTRACT

We present DEESSE [1], a tool that enables an exploratory and serendipitous exploration – at entity level, of the content of two different social media: Wikipedia, a user-curated online encyclopedia, and Yahoo Answers, a more unconstrained question/answering forum. DEESSE represents the content of each source as an entity network, which is further enriched with metadata about sentiment, writing quality, and topical category. Given a query entity, entity results are retrieved from the network by employing an algorithm based on a random walk with restart to the query. Following the emerging paradigm of *composite retrieval*, we organize the results into topically coherent bundles instead of showing them in a simple ranked list.

Categories and Subject Descriptors

H.4.1 [**Information Systems Applications**]: World Wide Web—*Web searching and information discovery*

Keywords

Entity Search, Entity Networks, Composite Retrieval

1. BACKGROUND

To provide web users with an engaging search experience, modern search engines need to go beyond the pure relevance of search results and to consider more subjective qualitative factors, such as *serendipity* and *interestingness*. Serendipitous search systems [15] attempt to entertain users with information items that are surprising and interesting, albeit sometimes not fully related to their search intent.

In a recent work [6], we have proposed how to define, operationalize, and evaluate serendipitous search over user-generated content. We have introduced an entity search framework that supports a serendipitous exploration of two of the most popular social media: Wikipedia and Yahoo Answers. The former is a highly curated, collaboratively edited online encyclopedia, whereas the latter is the largest community question/answering system, representing a faithful mirror of people's interests and opinions.

In this paper we demonstrate the tool introduced in our previous work [6]. Our system, dubbed DEESSE (entity-Driven Exploratory and serendipitous Search SystEm), is based on the paradigm of *entity search* [2, 12, 13], which is nowadays a prominent alternative to document search to answer complex information needs by building semantically rich answers in the form of entities and their relations. DEESSE is available online [1].

DEESSE represents the content of each data source as a network of entities, which are connected based on the similarity of the documents in which they appear. Given a query entity, entity recommendations are retrieved from the network using an algorithm based on a random walk with restart to the query. Entity networks are enriched with further metadata, still derived from the documents, and regarding the intensity of the emotion, the quality of the writing, and the topical category of the text surrounding each entity.

In our previous work [6], we have used the above metadata to constrain the result sets, and measure the extent to which each dimension contributes to the perceived serendipity, proposing two different approaches for assessing the serendipity of search results. We showed that both Wikipedia and Yahoo Answers offer results that are relevant, and dissimilar to those found through a web search. However, Yahoo Answers showed to be better at favoring the most interesting entities. The usage of topical metadata was found to be helpful for discovering more interesting results.

The tool described in this paper implements our original framework, and presents some extensions that we introduce to provide users with an improved exploratory search experience: *multi-linguality* and *bundled retrieval*.

For the first aspect, we have enriched DEESSE, which could originally only support queries in the English language, with Spanish. A separate entity network is built and queried for each language.

The second extension concerns the retrieval module. We adopt the paradigm of *composite* retrieval [11], which recently emerged as a powerful way to assist the users with complex information seeking activities. The idea is to organize results into item *bundles* designed to satisfy a number of properties, based on the users's preferences or needs. After evaluating different methods based on the various metadata available in DEESSE, we implemented a topical-bundling algorithm that organizes search results into topically coherent bundles, based on the categories of the query entity.

CIKM'14, November 3–7, 2014, Shanghai, China.
Copyright 2014 ACM http://dx.doi.org/10.1145/2661829.2661853.

2. SYSTEM DESIGN

The tool presented in this paper demonstrates our previous research [6], and improves it by adding support for multi-lingual and composite retrieval. DEESSE builds a separate module for each data-source/language combination. The architecture of such module consists in a back-end and a front-end part. The back-end works offline, extracting an enriched entity network from the considered dataset, and precomputing bundled result sets for every entity.

The front-end receives the query submitted by the user and sends a request to the back-end to retrieve the corresponding results. The technologies used for the front-end consist of a combination of a MySQL database, PHP, CSS, HTML and Javascript. D3.js[1] is being used to retrieve JSON formatted data and manipulate it for display, while the Bootstrap[2] framework is used to style the web interface.

We next describe the main components of the back-end.

Data and Languages. DEESSE exploits two main data sources, Yahoo Answers and Wikipedia, which are both available in different languages. While our original framework only considered English-language documents, the current, improved version of DEESSE also supports Spanish. From Yahoo Answers, we collected English and Spanish questions from 2010-2012, and the answers to these questions. For Wikipedia, we extracted the English and the Spanish dumps from December 2011. We use WikiExtractor[3] to strip the meta-content from each Wikipedia page.

Entity extractor. For entity extraction, we follow the common approach that for an extracted entity to exist, it must appear as a Wikipedia page [7, 10, 12, 13]. Our entity-extraction methodology, described in full details in the original paper[6], was chosen due to its suitability for large-scale data processing. It uses a machine-learning approach proposed by Zhou et al. [17] for resolving surface forms extracted from the text to the correct Wikipedia entity, and Paranjpe's *aboutness* ranking model [13] to rank the obtained entities according to their relevance for the text.

Network Extractor. Given the set of entities extracted from each language-specific dataset, we construct a network using a content-based similarity measure to create arcs between entities. Adopting the vector-space model [14], we represent each entity by a TF/IDF vector, extracted by the (order-insensitive) concatenation of all the documents where the entity appears. We then measure the similarity between any two entities in terms of the cosine similarity of the corresponding TF/IDF vectors. Because the TF/IDF weights cannot be negative, the similarity values will range from 0 to 1. We create an undirected network by computing all the pairwise similarities between the entities. However, the final network is not a complete graph. To avoid considering poorly significant relations, we restrict to the pairs of entities that co-occur in at least one document, and we prune all the arcs with similarity lower than a minimum threshold $\sigma = 0.5$[4]. The all-pairs similarity computation required to build the network was performed efficiently by using a distributed algorithm [3] that works on Hadoop.[5]

Entity feature extraction. The entity network was originally enriched with metadata regarding topic, quality and sentiment. Given that DEESSE currently displays only topical and sentiment features, we provide a brief description of these. The interested reader is deferred to the original paper [6] for a full description of the metadata extraction.

To build topical features, we exploit a proprietary taxonomy to assign topical categories to the documents in each dataset (full details in [6]). We derive category features for each entity in a graph, by aggregating over all the documents where the entity appears, and retaining the top 3 categories that are most frequently associated with such documents.

To assign a sentiment score to every entity, we classify the originating documents with SentiStrength,[6] obtaining a positive and a negative score that we combine into a *polarity* score [9], measuring the inclination towards positive or negative sentiment. By default, the tool computes document-level sentiment scores. However, a document is likely to mention many different entities, and the sentiment expressed around them may vary considerably. To handle this, we compute entity-level scores by considering small windows (20 words) of text around each mention of an entity, and then averaging across all mentions.

Entity Ranker. The entity ranker module extracts from a network, the top n entities that are most related to a query entity. Our core method [6] is inspired by random-walk based algorithms [8, 16], which have been successfully applied in many recommendation problems [4, 5].

The algorithm performs a *lazy* random walk with restart to the input entity. At each step, it either remains in the same node with high probability β, or follows one of the out-links with probability $1 - \beta$. In this case, the links are followed with probability proportional to the weights of the arcs. We rank all nodes based on the stationary distribution of this random walk, and select the top n with highest rank. Details concerning parameter settings, stopping criteria and corrections applied to reduce the bias towards popular entities, are provided in the original paper [6]. We use a `giraph`[7] implementation to perform the random walks efficiently.

Bundling algorithm. Our previous work [6] showed that the topic metadata contributed most to improve interestingness and relevance of the search results. Based on this, and on the fact that a topical organization of the results is a natural choice to facilitate the exploration of a large-scale knowledge base, we enrich our retrieval module with an algorithm that organizes search results into topically coherent bundles. We mentioned before that each entity has 3 categories. Our bundling algorithm produces a bundle for each of the categories associated with the query entity.

The bundle corresponding to one category is populated by taking from the ranking returned by the basic entity ranker, the top n entities among those belonging to the considered category. This approach may produce overlapping bundles. We choose not to force the algorithm to create disjoint bundles, not only because going further down in the rank would naturally hurt the relevance of the results, but also because this choice is more adherent with the natural scenario that happens for many queries, where the categories overlap. The current version of DEESSE extracts 3 topical bundles for each entity, with a maximum of 5 entities each.

[1] http://d3js.org/

[2] http://getbootstrap.com/

[3] `medialab.di.unipi.it/wiki/Wikipedia_Extractor`

[4] The value of the threshold was chosen heuristically

[5] `hadoop.apache.org`

[6] `http://wwww.sentistrength.wlv.ac.uk`

[7] `http://giraph.apache.org`

Indexing. Despite the fact that we use a distribute parallel implementation to perform random-walk computations on large-scale data efficiently, our retrieval algorithm requires a temporal computational cost of minutes to obtain results for a query entity. This cost makes running the ranking module at query time prohibitive. To make our solution viable, we perform the computation offline. To avoid storing the full stationary distribution of every node, we also run the bundling algorithm offline, and only store the resulting topical bundles obtained for each entity. Given the current settings, DEESSE stores a maximum of 15 (bundled) results per query, which requires on average $3KB$ per entity.

When a query comes from the front-end, the resulting (pre-computed) bundles are retrieved from the index. Complimentary metadata is provided by the entity feature extractor (e.g., sentiment) or fetched from external sources (abstract or image urls of Wikipedia page and top-rated Yahoo Answers question/answer pair).

3. SYSTEM INTERACTION

Regarding how the user can interact with DEESSE, the search results for the query entity are presented in the central panel of the web user interface. The results are grouped into 3 bundles, based on the categories of the query entity. For example, in the case of the query "NASA", the categories are *Astronomy & Space*, *Politics* and *Religion & Spirituality*.

For each of the entities in a bundle, an illustration of sentiment polarity is provided (if available), along with a link to the Wikipedia page of that entity. A click on an entity result will initiate a search with this entity as query.

Hovering an entity in the result list will trigger the retrieval of any available metadata from Wikipedia (thumbnail picture and Wikipedia abstract) and Yahoo Answers (top rated question and answer mentioning the entity).

Multiple searches can be carried out, and buttons will appear under the search bar to keep track of them. Clicking one of the previous searches will again show the results for that specific search, while clicking the close button in the top-right corner will remove the results for that query entity.

4. SYSTEM MAINTENANCE AND UPDATE

When it comes to updates of the data available in DEESSE, two constraints should be taken into account. First, the computation of the results for each query entity, as described in Section 2, is quite expensive due to the use of random walks. For this reason, results are precomputed and stored in an index. Next, when the number of query entities grows, the computation becomes even more expensive. Both these constraints make that a daily update of the data is not worth the minor improvement in query behaviour for the end user. Of course, having a slower process implies that we will not be able to serve extremely recent time-sensitive queries. However, we believe that this limitation is acceptable, given that DEESSE is a tool built to support the exploration of Yahoo! Answers data, and extremely time-sensitive queries are not a critical use case in this context.

5. CONCLUSIONS

We have presented DEESSE, a tool that enables an exploratory and serendipitous exploration of the content of different social media. It combines the exploration of data from both Wikipedia and Yahoo Answers, two platforms that allow contributions from their users, but are different in the nature of their moderation. We are currently investigating a number of natural extensions to this work.

First, we are working to include support for additional languages. Directly related to this is the question on how to link the results from different language domains. A first, naive, approach would be to aggregate the results from different languages, while a more advanced approach would be to enable cross-lingual entity linking. Second, we would like to explore better exploitation of the current metadata by devising new bundling algorithms, while we are considering to include new sources of metadata (e.g. temporal features) to enable adaptive filtering of results based on freshness (e.g., exclude the less recent Yahoo Answers discussions). Finally, to evaluate our tool at scale, DEESSE should be released to a fully open user base, which is not the case at this moment.

6. ACKNOWLEDGEMENT

This work was partially funded by the Linguistically Motivated Semantic Aggregation Engines (LiMoSINe[8]) EU project.

References

[1] URL `http://deesse.limosine-project.eu`.
[2] K. Balog, E. Meij, and M. de Rijke. Entity search: building bridges between two worlds. In *SEMSEARCH*, 2010.
[3] R. Baraglia, G. De Francisci Morales, and C. Lucchese. Document similarity self-join with mapreduce. In *ICDM*, 2010.
[4] F. Bonchi, R. Perego, F. Silvestri, H. Vahabi, and R. Venturini. Efficient query recommendations in the long tail via center-piece subgraphs. In *SIGIR*, 2012.
[5] I. Bordino, G. De Francisci Morales, I. Weber, and F. Bonchi. From machu picchu to rafting the urubamba river: Anticipating information needs via the entity-query graph. In *WSDM*, 2013.
[6] I. Bordino, Y. Mejova, and M. Lalmas. Penguins in sweaters, or serendipitous entity search on user-generated content. In *CIKM*. ACM, 2013.
[7] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *EMNLP, 2011*, 2011.
[8] G. Jeh and J. Widom. Scaling personalized web search. In *WWW*, 2003.
[9] O. Kucuktunc, B. Cambazoglu, I. Weber, and H. Ferhatosmanoglu. A large-scale sentiment analysis for yahoo! answers. In *WSDM*, 2012.
[10] S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. Collective annotation of wikipedia entities in web text. In *SIGKDD*, pages 457–466, 2009.
[11] I. Mendez-Diaz, P. Zabala, F. Bonchi, C. Castillo, E. Feuerstein, and S. Amer-Yahia. Composite retrieval of diverse and complementary bundles. *IEEE TKDE*, 99(PrePrints):1, 2014.
[12] D. Milne and I. H. Witten. Learning to link with Wikipedia. In *CIKM*, 2008.
[13] D. Paranjpe. Learning document aboutness from implicit user feedback and document structure. In *CIKM*, 2009.
[14] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18(11), 1975.
[15] E. Toms. Serendipitous information retrieval. In *DELOS Workshop*, pages 11–15, 2000.
[16] H. Tong and C. Faloutsos. Center-piece subgraphs: problem definition and fast solutions. In *KDD*, 2006.
[17] Y. Zhou, L. Nie, O. Rouhani-Kalleh, F. Vasile, and S. Gaffney. Resolving surface forms to Wikipedia topics. In *COLING*, 2010.

[8] `www.limosine-project.eu`