READFAST: High-relevance Search-engine for Big text

Michael Gubanov
MIT CSAIL
The Stata Center
Cambridge, MA, 02139
michaelgubanov@csail.mit.edu

Anna Pyayt
University of South Florida
IBIS Laboratory
Tampa, FL, 33620
pyayt@usf.edu

ABSTRACT

Relevance of search-results is a key factor for any search engine. In order to return and rank the Web-pages that are most relevant to the query, contemporary search engines use complex ranking functions that depend on hundreds of *features*. For example, presence or absence of the query keywords on the page, their proximity, frequencies, HTML markup are just a few to name. Additional features might include fonts, tags, hyperlinks, metadata, and parts of the Web-page description. All this information is used by the search-engine to rank HTML Web pages returned to the user, but is unfortunately absent in free text that has no HTML markup, tags, hyperlinks, and any other metadata, except *implicit* natural language structure.

Here we demonstrate one of the first Big text search engines that leverages hidden structure of the natural language sentences in order to process user queries and return more relevant search-results than a standard *keyword*-search. It provides a *structured index* extracted from the text using Natural Language Processing (NLP) that can be used to browse and query free text.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: General; H.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.2 [**Database Management**]: Heterogeneous Databases

Keywords

Search; Data Integration; Natural Language Processing (NLP); Information Retrieval; Structure Extraction

1. INTRODUCTION

Natural language has a rich structure that implicitly encodes information about objects and their relationships. All this information is challenging to extract with high precision and recall, therefore *keyword*-search and its modifications largely ignoring this information became standard for search over free text. Being easy to use and implement, standard *keyword*-search (exact substring match) largely ignores linguistic properties of natural language and therefore retrieves either too few relevant or too many irrelevant results compared to what an ideal search-engine could have returned from the same corpus. For example, if a user has a 500-page reference book about information extraction systems and is interested to find out what they consist of, s/he could try searching for *system consists*, *system consist* that would return no relevant search results or for *system* or *consist* separately that would return more than 50 irrelevant sentences. Other combinations of these words again would either return nothing relevant or just too many irrelevant results to be of any value. Finally, the only option left is to read the book or to skim through it and manually find relevant pages. For Big text, this task becomes completely infeasible. Therefore, despite being simple and intuitive, in practice, *keyword*-search turns out to be imprecise and inflexible for search in text. There is no guarantee that the keywords in the user query happen to match a variety of different linguistic forms of the sentences containing needed information. As a result, one can only hope that some queries would match and retrieve at least some of the relevant sentences.

The major challenge on the way to solving this problem and providing more precise and flexible text search is *automatic* extraction of the structure from natural language sentences to facilitate more robust search algorithms. There are many attempts [2, 19] to perform entity tagging, synonym resolution, and other relevant information retrieval too numerous to list here, but all of them usually focus on specific data representation, corpus, domains, language, patterns, or other reasonable restrictions to make *fully automatic* extraction feasible and precise enough to be useful. Here we demonstrate a system that is intentionally designed to be general and work on any free text.

Another important challenge is leveraging the extracted structure to provide an interface or advanced query processing algorithm to make the overall search experience more robust than *keyword*-search. For a new interface design, the main goal is to provide the most useful snapshot of information using limited space; for the search-algorithms - overcoming different naming conventions for the same entities in the corpus (e.g. *car*, *vehicle*), and intelligently using the semantic structure during search or query processing to return the most relevant search results available in the corpus.

Here we demonstrate one of the first Big text systems that simplifies access to information in a large free text

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.
ACM 978-1-4503-2263-8/13/10.
http://dx.doi.org/10.1145/2505515.2508215.

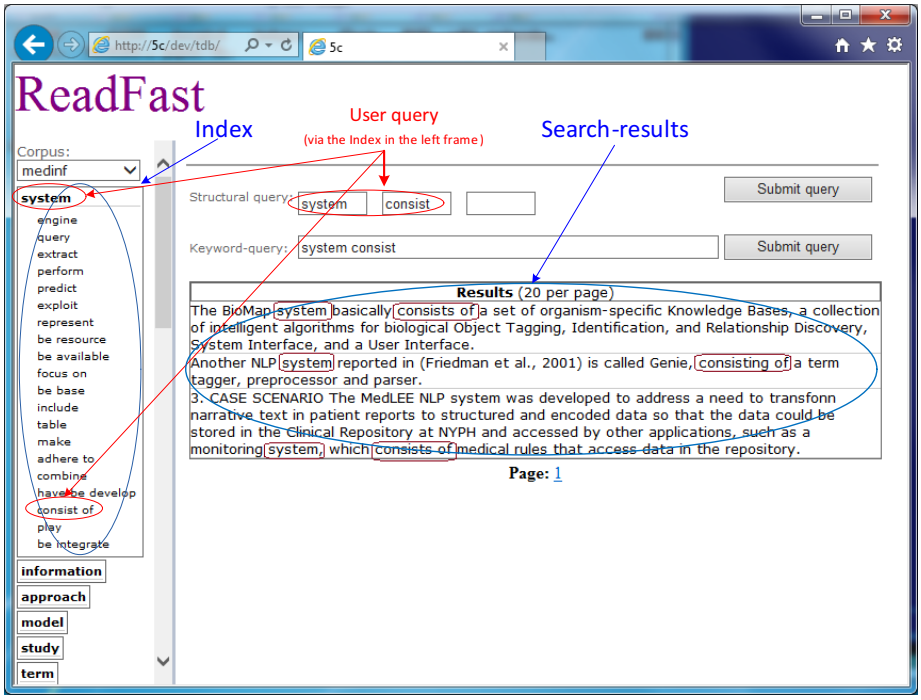

Figure 1: A user navigates a Big text corpus composed from Biomedical literature over *system* and *consist* to quickly find all typical components of an information extraction system.

corpora via a structured *index* used to facilitate structured browsing and search. Figure 1 shows the index automatically extracted by ReadFast (RF) from a 500-page book on Biomedical Informatics about state-of-the art Biomedical information retrieval systems. Without even opening the book the user can see in the index what are its main terms and therefore conclude the book's major themes. If a text corpus is new, the user can first explore the index, select the entities, its related actions and query the corpus with the structural queries generated by the system. We evaluate this approach and justify that on average it provides significantly more relevant search results than *keyword*-search on a certain class of queries.

We plan to partially demonstrate the information extraction pipeline followed by the user queries and browsing of Big text corpus. The interface in Figure 1 contains the edit boxes for structural search, *keyword*-search, and the browser in the left frame. Some of the algorithms and evaluations that served as a foundation for this system are described in more detail in [6, 9, 7, 8].

2. ARCHITECTURE

The ReadFast components are in Figure 2. The system provides structural access to any natural language text corpus. First, Big text corpus is split into chunks and each sentence is parsed using a distributed parser [14] that produces a set of the main *entities* from the text and their *actions* as well as converts the text corpus into a semi-structured format, indexed for structural search. The *entities* and *actions* are then composed into a search-index that is shown to the user in the interface (e.g. left frame in Figure 1) and used for browsing and search. The text corpus is stored in a semi-structured distributed storage, indexed and suitable for fast retrieval of sentences by terms. A query (*entity*:*action* pair) either generated by the browser or coming from the user is executed against the storage engine and a set of relevant sentences is returned to the user. Each component of the system is discussed below in more detail.

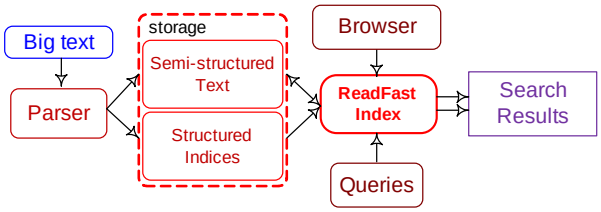

Figure 2: Text is parsed with the parser to extract an index that is used for navigation and querying.

Index: ReadFast index extracted from the corpus consists of *entities* and their *actions*. It is stored in the back-end storage indexed for faster retrieval (see Figure 1). It is displayed in the interface in form of tables with attributes (e.g. left frame in Figure 1). For example, for the health records corpus the main entities could be `patient, pain, heart` with the `attributes` *relieve, administer, beat*). See [7, 9, 8] for more details on extraction algorithms.

Browsing: A sample ReadFast user interface is shown in Figure 1. It supports structural browsing using the RF index. A user can first select the corpus and then click on the *entity* (e.g. *system* in Figure 1), and its *action* (e.g. *consists*). The engine generates a structural query based on the selected terms in the index, executes it against the text corpus, and returns search-results in the right frame. We

use UFOs [6] to store different entity representations behind a standardized search interface shown to the user.

Search: A sample ReadFast user interface is shown in Figure 1. Similar to browsing described above, it supports structural search. A user can first select the corpus and then type in a structural query in the corresponding edit box in the right frame. The engine executes the structural query based against the text corpus, and returns search-results in the right frame.

Storage: ReadFast uses a sharded semi-structured storage to store the Big text corpora and a parallel relational storage engine for indexes and optimized query processing.

3. EVALUATION

Evaluation of ReadFast search-results relevance was done similar to how it is usually done for contemporary Web-search engines. The biggest difference is that the Web-search result set is composed of ordered links to the Web-pages relevant to the query, whereas here the result set is a set of natural language sentences from the corpus ordered by relevance to the user query. We used NDCG - Normalized Discounted Cumulative Gain as relevance metric as it combines precision and recall and also is one of the standard metrics for evaluating search results in Information Retrieval [1]. We designed a several experiments to measure relevance gain of ReadFast compared to *keyword*-search on several Big text corpora. On average, ReadFast NDCG was 20-30%, which is a significant improvement. Interested reader is referred to [7] for a more detailed description of the evaluation methodology.

4. RELATED WORK

Much of the current research in Data Management, Information Retrieval, and Search is devoted to the Web, social networks, personal resources, and unfortunately does not apply directly to text. The most recent relevant research by Dong et al [5] lays foundations for generic selective information integration critical in search. Another significant work in [4] sheds light on controversial decision making process in large-scale data fusion. Halevy in [12] describes research efforts in a structural realm of large-scale information fusion. Gupta in [11] gives a partial overview of recent structured data research related to Web search. Many recent venues keep highlighting text as an area of growing interest to Data Management communities [19, 6, 3, 20, 18, 17, 15, 10, 16, 13].

ReadFast is more general in that it intentionally avoids any specific format, corpus, language, or other algorithmic restrictions to provide a general structural search foundations applicable to any text corpus.

5. FORWARD-LOOKING STATEMENTS

Here we demonstrate one of the first systems for advanced access to Big text. This system is based on generic algorithms to extract and leverage additional information from the linguistic structure of the sentences to enhance search over free text. We evaluated relevance gain compared to *keyword*-search using NDCG and on average observed 20-30% improvement in relevance of search results. We demonstrated and justified two main use cases of the system - browsing of a new corpus using the ReadFast index, useful when the user does not know yet what to search for and structural search with the user query. We expect Big text quickly become an area of growing interest, because of the wealth of information buried in text behind the inaccessibility barrier.

6. REFERENCES

[1] E. Agichtein, E. Brill, and S. Dumais. Improving web search ranking by incorporating user behavior information. In *SIGIR*, 2006.

[2] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 2008.

[3] L. Chilton, G. Little, D. Edge, D. Weld, and J. Landay. Cascade: Crowdsourcing taxonomy creation. In *CHI*, 2013.

[4] X. L. Dong, B. Saha, and D. Srivastava. Explaining data fusion decisions. In *WWW*, 2013.

[5] X. L. Dong, B. Saha, and D. Srivastava. Less is more: Selecting sources wisely for integration. In *VLDB*, 2013.

[6] M. Gubanov, L. Popa, H. Ho, H. Pirahesh, P. Chang, and L. Chen. Ibm ufo repository. In *VLDB*, 2009.

[7] M. Gubanov, A. Pyayt, and L. Shapiro. Readfast: Browsing large documents through unified famous objects (ufo). In *IRI*, 2011.

[8] M. Gubanov and L. Shapiro. Using unified famous objects (ufo) to automate alzheimer's disease diagnostics. In *BIBM*, 2012.

[9] M. Gubanov, L. Shapiro, and A. Pyayt. Learning unified famous objects (ufo) to bootstrap information integration. In *IRI*, 2011.

[10] M. Gubanov and M. Stonebraker. Bootstraping synonym resolution at web scale. In *DIMACS*, 2013.

[11] N. Gupta, A. Halevy, B. Harb, H. Lam, H. Lee, J. Madhavan, F. Wu, and C. Yu. Recent progress towards an ecosystem of structured data on the web. In *ICDE*, 2013.

[12] A. Halevy. Data publishing and sharing using fusion tables. In *CIDR*, 2013.

[13] R. Helaoui, D. Riboni, M. Niepert, C. Bettini, and H. Stuckenschmidt. Towards activity recognition using probabilistic description logics. In *AAAI*, 2012.

[14] D. Klein and C. Manning. Fast exact inference with a factored model for natural language parsing. 2007.

[15] M. Niepert. Lifted probabilistic inference: An mcmc perspective. In *STAR AI*, 2012.

[16] M. Niepert. Markov chains on orbits of permutation groups. In *UAI*, 2012.

[17] M. Niepert. Rockit: Exploiting parallelism and symmetry for map inference in statistical relational models. In *AAAI*, 2013.

[18] M. Niepert. Symmetry-aware marginal density estimation. In *AAAI*, 2013.

[19] A. Singhal. Introducing the knowledge graph: Things, not strings. In *Google Blog*, 2012.

[20] C. Zhang, R. Hoffmann, and D. Weld. Ontological smoothing for relation extraction with minimal supervision. In *AAAI*, 2012.