STICS: Searching with Strings, Things, and Cats

Johannes Hoffart
Max Planck Institute for Informatics
jhoffart@mpi-inf.mpg.de

Dragan Milchevski
Max Planck Institute for Informatics
dmilchev@mpi-inf.mpg.de

Gerhard Weikum
Max Planck Institute for Informatics
weikum@mpi-inf.mpg.de

ABSTRACT

This paper describes an advanced search engine that supports users in querying documents by means of keywords, entities, and categories. Users simply type words, which are automatically mapped onto appropriate suggestions for entities and categories. Based on named-entity disambiguation, the search engine returns documents containing the query's entities and prominent entities from the query's categories.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]

Keywords

Entity Search; Category Search; Document Retrieval

1. MOTIVATION AND CONTRIBUTION

"Things, not Strings" has been Google's motto when introducing the Knowledge Graph and the entity-awareness of its search engine. When you type the keyword "Klitschko" as a query, Google returns Web and news pages and also explicit entities like Wladimir Klitschko and his brother Vitali (with structured attributes from the Knowledge Graph). Moreover, while typing, the query auto-completion method suggests the two brothers in entity form with the additional hints that one is an active boxer and the other a politican.

However, the Google approach still has limitations. First, recognizing entities in a keyword query and returning entity results seems to be limited to prominent entities. Unlike the Klitschko example, a query for the Ukrainian pop singer "Iryna Bilyk" does not show any entity suggestions (neither for auto-completion nor in the search results). Second, Google seems to understand only individual entities, but cannot handle sets of entities that are described by a type name or category phrase. For example, queries like "Ukrainian celebrities" or "East European politicians" return only the usual ten blue links: Web pages that match these phrases. The search engine does not understand the user's intention to obtain lists of people in these categories.

SIGIR'14, July 6–11, 2014, Gold Coast, Queensland, Australia.
ACM 978-1-4503-2257-7/14/07.
http://dx.doi.org/10.1145/2600428.2611177.

This demo presents a novel system, STICS, that extends entity awareness in Web and news search by tapping into long-tail entities and by understanding and expanding phrases that refer to semantic types. STICS supports users in searching for strings, things, and cats (short for categories) in a seamless and convenient manner. For example, when posing the query "Merkel Ukrainian opposition", the user is automatically guided, through auto-completion, to the entity `Angela Merkel` and the category `Ukrainian politicans`, and the latter is automatically expanded into `Vitali Klitschko`, `Arseniy Yatsenyuk`, etc. The search results include pages talking about "the German chancellor met the Ukrainian opposition leader and former heavy-weight champion", even if these texts never mention the strings "Angela Merkel" and "Vitali Klitschko".

2. RELATED WORK

Prior work on semantic search and entity retrieval (e.g., [1, 2]) has mostly focused on keyword input and entities as query results. Some work on searching with entities and relations (e.g., [3]) requires structured input queries and returns RDF triples or Web tables. [4, 5] focuses on efficient auto-completion for combined phrase- and entity-queries, with the goal of retrieving entities. All this is quite different from our model which, additionally to phrases, allows entities as input and returns documents as output. No prior work has considered a combination of text phrases, entities, and semantic categories in the query model.

3. ALGORITHMIC BUILDING BLOCKS

The basis for querying text documents in terms of entities and categories is the automatic detection and marking of entities in the corpus and their association with type categories. We use the Named Entity Disambiguation system AIDA [6] and the type system of `yago-knowledge.org`, which combines Wikipedia categories with WordNet classes. We have run this machinery on a corpus of ca. 500,000 news articles collected from 100 important feeds since mid 2013. Based on this annotated and indexed corpus, the STICS search engine comprises a number of building blocks for query auto-completion, document ranking, and category expansion at query time.

Auto-Completion. The user does not know the underlying knowledge in advance; so in order to navigate a huge number of entities and categories, some form of input auto-completion is crucial. Our method consists of two components: the first one for retrieving the candidates for a given prefix typed by the user, and the second one for ranking the

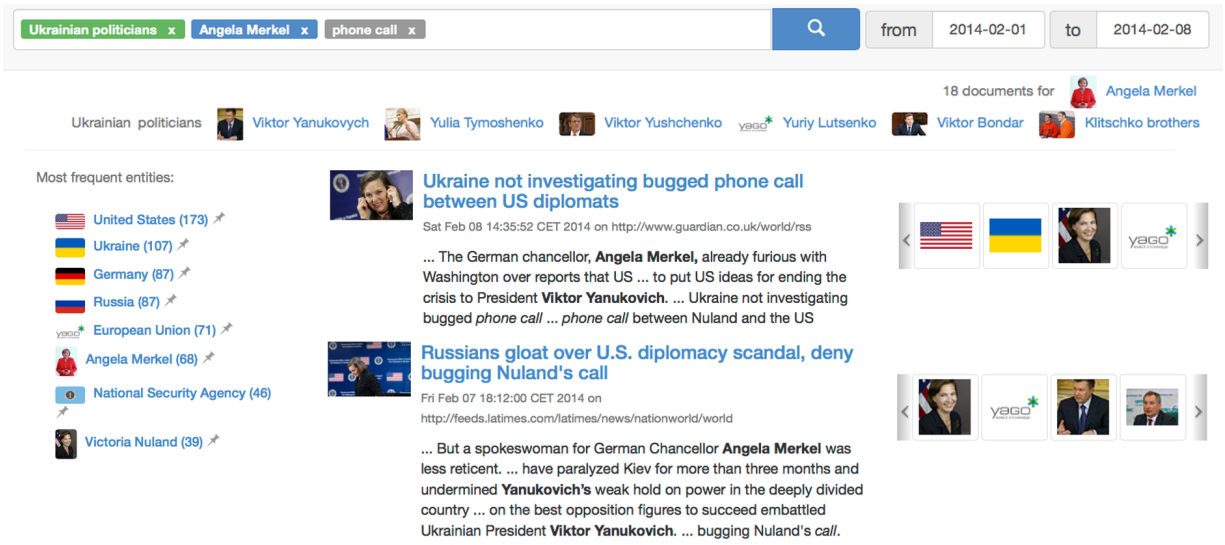

Figure 1: Screenshot of the STICS search engine over a news corpus

candidates in a judicious manner. For *candidate retrieval*, the full canonical names of entities and categories are indexed for prefix lookups. For entities, we additionally store all token-level suffixes; so users are able to start typing at any token (e. g. "Merkel" will also show `Angela Merkel`). For categories, where the order of tokens is somewhat arbitrary ("Presidents of the United States" vs. "United States Presidents"), we additionally index all 3-token permutations. For the *candidate ranking*, the entities are ranked by global popularity, estimated by the number of incoming links to their Wikipedia articles. The categories are ranked by a mixture model capturing both the category's prominence estimated by the prominence of all its entities as well as its specificity. A category's specificity is estimated by the likelihood of its entities being present also in other categories. This way, the suggestions prefer categories whose entities are truly specific and do not appear in many other coarse-grained categories.

Document Ranking. Our demo setting pertains to a continuous stream of news documents that we annotate with entities in real-time. In this setting, ranking is best done chronologically, with the latest news shown at the first spot. However, we can change the ranking model to incorporate statistical language models where token and entity models are combined using a mixture model including time [8].

Category Expansion. Once the query is fully specified, categories need to be expanded into a set of individual entities to be looked up in the index. If there is no further context given in the query, the most prominent entities for the categories are used. However, once the user intent is made clear by specifying additional entities, the expansion should prefer entities that are both prominent and related to the user-given entities. Once `Angela Merkel` is specified next to `Ukraninain politicians`, the Klitschko brothers, as long-time German residents, are ranked higher. We realize this context-awareness by a mixture model of global entity prominence (like for auto-completion ranking) and global entity-entity coherence computed by the entity relatedness measure of [7]. The coherence is computed between all entities of a given category and all entities specified directly in the query, averaging over all of them.

4. USE CASES

Searching with String, Things, and Cats. The possibility to pose queries beyond strings enables a new search experience, allowing users to specify their actual intent in a crisper manner. However, not all search intents can be expressed using entities and categories, so the combination with regular search terms is kept. An example where combining all options that STICS offers is crucial is searching for opinions of `Angela Merkel` on the scandalous "phone call" of the US diplomat `Victoria Nuland` in the context of the Ukrainian riots. Figure 1 shows a screenshot of STICS for this example query.

Navigating Documents by Entities. The most frequent entities in the retrieved documents are displayed next to each result, allowing quick refinement of the initial query. In this setting the category search shows its potential, as users can begin their search by a more general category of entities, then refine by clicking the displayed entities. Thanks to the context-aware expansion, the refinement effect is even more pronounced. For instance, adding `Angela Merkel` in the example query raises the expanded `Klitschko brothers` by 5 ranks, thus re-adjusting the query's focus.

5. REFERENCES

[1] K. Balog, M. Bron, M. de Rijke: Query modeling for entity search based on terms, categories, and examples. TOIS 29(4), 2011
[2] Z. Nie, J.-R. Wen, W.-Y. Ma: Statistical Entity Extraction From the Web. Proc. IEEE 100(9), 2012
[3] S. Elbassuoni et al.: Language-model-based ranking for queries on RDF-graphs. CIKM 2009
[4] H. Bast, B. Buchhold: An index for efficient semantic full-text search. CIKM 2013
[5] H. Bast et al.: ESTER: efficient search on text, entities, and relations. SIGIR 2007
[6] J. Hoffart et al.: Robust Disambiguation of Named Entities in Text. EMNLP 2011
[7] J. Hoffart et al.: KORE: Keyphrase Overlap Relatedness for Entity Disambiguation. CIKM 2012
[8] X. Li, W. B. Croft: Time-based language models. CIKM 2003