Blognoon: Exploring a Topic in the Blogosphere

Maria Grineva*
ETH Zürich, Switzerland
grinevam@inf.ethz.ch

Maxim Grinev*
ETH Zürich, Switzerland
grinevm@inf.ethz.ch

Dmitry Lizorkin*
Google
lizorkin@google.com

Alexander Boldakov*
Semantic Dimension
boldakov@
semanticdimension.com

Denis Turdakov
ISP RAS
turdakov@ispras.ru

Andrey Sysoev,
Alexander Kiyko
ISP RAS
{syssoev,kiyko}@ispras.ru

ABSTRACT

We demonstrate Blognoon, a semantic blog search engine with the focus on topic exploration and navigation. Blognoon provides concept search instead of traditional keywords search and improves ranking by identifying main topics of posts. It enhances navigation over the Blogosphere with faceted interfaces and recommendations.

Categories and Subject Descriptors

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

General Terms

Design, Algorithms

1. INTRODUCTION

Today, the functionality offered by blog search engines is similar to web search. Most blog search engines provide conventional keywords-based search with minor extensions. For example, Google provides keywords-based search for blogs that differs from the web search in the frequency of index updates and ability to rank results by recency. Technorati, probably, the best known blog search engine, in addition to keyword search, allows users to search by tags which bloggers attach to their posts. Also, it computes "authority" score for each blog based on the number of blogs linking to a given blog, and it allows readers to browse popular posts both by their recency and by the amount of "attention" they received on mainstream media sites.

Keywords-based search, even augmented with the features described above, is good when the user knows exactly what to search for. However, it gives poor help when the user's goal is to learn about, explore, or understand a broad topic. Being a rich collection of people's opinions, discussions and reviews, the Blogosphere often appears to be a better place for *topic exploration* than the whole Web [5]. Paradoxically, topic exploration facility is a weak part of the existing blog search engines.

We have developed Blognoon, a semantic blog search engine with the focus on topic exploration and navigation. Blognoon tackles the problem by providing concept search instead of keywords search and improves ranking by identifying main topics of posts. It enhances navigation over the Blogosphere with faceted interfaces and recommendations.

The technology behind Blognoon leverages Wikipedia as the world biggest resource of human knowledge. There is a large body of work on using Wikipedia to enhance text processing [3]. Blognoon constitutes a unique combination of Wikipedia-based techniques to identify main topics of posts while indexing, build faceted navigation interfaces and recommend relevant posts. In Section 4 we describe these techniques in detail.

2. SURFING THE BLOGOSPHERE WITH BLOGNOON

We will demonstrate the following features of Blognoon.[1]

Search by concept. Blognoon provides search by concepts instead of traditional keywords search. Suppose, the user wants to explore the *Clean Tech* topic. The result of the query is a ranked list of posts that contain *Clean tech* concept and also relevant concepts such as *Renewable energy* and *Biofuel*. *Clean tech* is the central and well-covered topic in top-ranked posts, while lower ranked posts may cover relevant topics. When typing the query in a query form, the user gets suggestions which are the concepts described in Wikipedia. Query suggestions are sorted by their popularity in Wikipedia instead of alphabetical order, which would turn out in a useless list of many little known concepts and synonyms.

Concepts tips. When searching using conventional web search engines, it is often hard to understand by reading the title and the snippet of the result item, why this item is relevant to the search query and what is it about. To help with this, we generate query-relevant concepts tips for each post that appears in the result. In Figure 1, for example, the top-ranked post is accompanied with concepts tips such as *Renewable energy* and *Global warming*. Every concept tip is provided with a pop-up window where the user can read its Wikipedia description.

Faceted navigation interface. The result of a query can contain thousands of posts. To help the user to look over the result, Blognoon provides facets on the right pane.

*This work was carried out while the authors worked at the Institute for System Programming of RAS.

WWW 2011, March 28–April 1, 2011, Hyderabad, India.
ACM 978-1-4503-0637-9/11/03.

[1]Blognoon is the property of ISP RAS and is available on the Web at *http://blognoon.com*

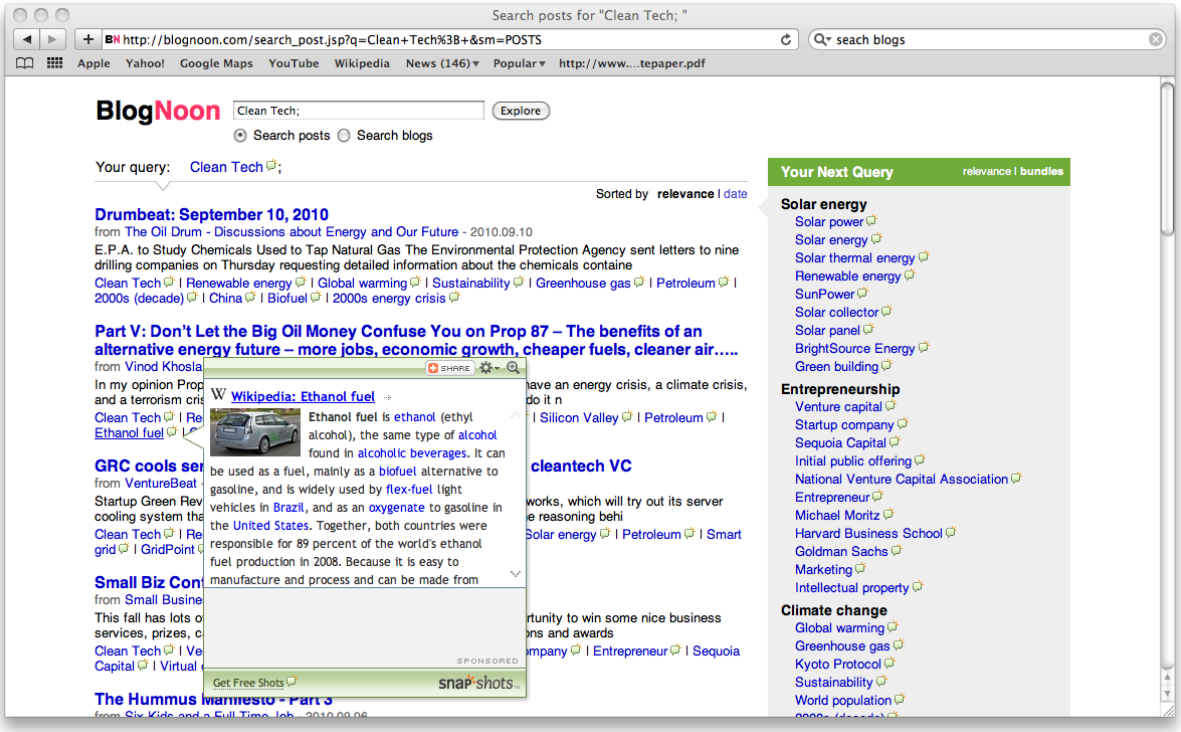

Figure 1: Blognoon concept search, with concept tips under each result item and facets on the right

You can think of the facets as a map that shows the user where he is in the field and highlights the terrain. Facets can be used to narrow the current search or to start a new search. So, the user can decide which particular section to explore more closely, at arbitrary levels of specificity. For example, for "Clean tech" query, Blognoon provides named groups of facets such as *Solar Energy*, *Enterpreneurship* and *Climate Change*. In facets, the user can find a lot of useful details about the explored topic, such as people's names, companies or brands.

What this blog is about. When the user becomes interested in a particular blog and wants to read more about it, Blognoon provides an overview for each blog in a form of structured tag cloud.

Navigation via recommendations. When the user opens up a post or a blog overview, the systems provides recommendations of other relevant blogs or posts on the right pane. Each recommended blog or post is accompanied with keywords which explain its content and why it is relevant.

3. OUR MODEL

Concepts. The basic element of our model is a *concept*. A concept in our approach is associated with a Wikipedia article, since each Wikipedia article is typically dedicated to describe a single concept of the real world. Concepts are denoted by lower-case letters, e.g. c_1, c_2.

Semantic Relatedness. The primary operation for comparing concepts is their pair-wise *semantic relatedness* metric. Semantic relatedness is specified via a *neighbourhood* for a Wikipedia article, which is considered as a set of all articles hyperlinked with that one (in either direction of a link). Relatedness for a pair of concepts is then computed as an extended Dice coefficient over neighbourhoods of their respective articles, with weight boost given to certain Wikipedia link types [9]. This link-based metric is computationally inexpensive and effectively captures human intuition on semantic relatedness of concepts [9]. The relatedness metric is denoted as a function: $\mathrm{sim}(c_1, c_2) \Rightarrow s,\ 0 \le s \le 1$.

For a primary data abstraction, we use a *map* – a set of pairs, denoted as $M = \{c_1 \triangleright v_1, \ldots, c_n \triangleright v_n\}$, where the first component of each pair denotes a key and the second component – an associated value. Accessing a value for a given key is denoted as $M[c_i]$, and the set of all keys is obtained as $M.keys \Rightarrow \{c_1, \ldots, c_n\}$. Conventional set operations like union have their usual meaning for a map. A particular case of a map that has concepts for keys and non-negative weights for values is refered to below as *weighted concepts*.

Semantic relatedness measure allows obtaining all concepts related to a given concept together with their similarity scores, denoted as a function as:

$$\mathrm{getSimilar}(c_i) \Rightarrow \{c_j \triangleright \mathrm{sim}(c_i, c_j) \mid \mathrm{sim}(c_i, c_j) > 0\}$$

Wikipedia as a knowledge base. We use several other operations over concepts, all of which rely on solely the link structure of Wikipedia:

We call a *semantic graph* induced by a set of concepts $C = \{c_1, \ldots, c_n\}$ a graph $G(C, E)$ which has these concepts for vertices and has weighted edges between semantically related concepts: $E = \{(c_i, c_j, \mathrm{sim}(c_i, c_j)) \mid \mathrm{sim}(c_i, c_j) > 0\}$, where the third component in each triple is weight annotating the edge. We assume below that a semantic graph for C is constructed by a function $\mathrm{SemanticGraph}(C) \Rightarrow G$.

For a semantic graph G we use the Girvan-Newman algorithm [2] for detecting *communities* there, which in our case are clusters of concepts that are more semantically similar to each other within a single cluster than across different clusters. Formally, we assume having a function GirvanNewman$(G) \Rightarrow \{V_1, \ldots, V_m\}$, such that $V_j \subset C$, $\cup_{j=1}^{m} V_j = C$, $V_i \cap V_j = \emptyset$ for $i \neq j$.

Finally, we make use of the Wikipedia category structure for concepts generalization. For a set of concepts C, we infer a concept c' being a semantic generalization for all members of C. We use the Spreading Activation algorithm [8] for this task, refered to below as SpreadingActivation$(C) \Rightarrow c'$.

4. OUR TECHNIQUES

Our system consists of four components: (i) Wikipedia knowledge base, (ii) blog database, (iii) an offline module that crawls blogs and indexes them into the database, (iv) a web application that serves user requests.

Wikipedia knowledge base (WKB below) contains all data from Wikipedia required for our techniques: concept names, link structure and statistical information. For achieving the best computational efficiency, data is stored in main memory, with a dedicated machine with 8Gb RAM used for hosting the WKB. The API of the WKB in particular includes the functions introduced in the previous section, and the other modules access the WKB via remote method invocation.

The *offline module* periodically crawls a set of top blogs. For each new blog post collected, all concepts mentioned in the post are retrieved as described in [9]. From those, key concepts are selected that effectively illustrate the main subject of the post using our technique described in [4]. The processed post and its concept information are indexed and stored in the *blog database* on a separate machine. The online *web application* queries the database for answering user requests.

The following subsections describe our techniques involved in the processing pipeline in more detail.

4.1 Indexing and Search

We use Apache Lucene[2] for indexing blogs and posts and we have customized its indexing framework to support concept search. Seamless integration of concept search with conventional full-text search is provided, by working out special matching rules.

For each processed blog post, all its words are indexed in conventional manner. Additionally, all concepts are located in the post as described in [9]. With N being the total number of concepts in the post and N_i the number of occurrences for a concept c_i, a concept c_i is associated with an indexing weight equal to:

$$\begin{aligned} \text{weight}_i &= \frac{N_i}{N} \cdot \text{textRank}(c_i) \ , \\ \text{textRank}(c_i) &= (1-d) + d \sum_{c_j} \frac{\text{sim}(c_i, c_j)}{\sum_{c_k} \text{sim}(c_i, c_k)} \text{textRank}(c_j) \end{aligned}$$

Here, textRank(c_i) is the TextRank score [6] for c_i, computed over the semantic graph induced by all concepts in the blog post. Using TextRank achieves the effect of redistributing weights towards concepts that play semantically central role in a blog post. We employ an undirected specialization of TextRank, which agrees with finding by Mihalcea and Tarau who discovered that an undirect relation between graph nodes produces better results for natural language tasks [6]. Normalization by N eliminates potential bias towards longer blog posts, thus taking into account a relative frequency of a concept within a post.

Search functionality provided by Blognoon allows a query to contain both concepts $\{c_i\}$ and conventional words $\{w_j\}$. To assist the user in specifying desired meaning for ambiguous terms, as well as to speed up typing, a query suggest facility is provided. To incorporate semantic relations between concepts, the query is expanded with concepts similar to the specified ones:

$$\begin{aligned} \text{Query} &= \{c_1, \ldots, c_n, w_1, \ldots, w_m\} \\ \text{ExpandedQuery} &= \bigcup_{i=1}^{n} \text{getSimilar}(c_i) \bigcup \{w_1, \ldots, w_m\} \end{aligned}$$

Such an expansion allows finding not only blog posts with the concepts directly queried, but also with semantically related ones, with match being weaker for less similar concepts. The ExpandedQuery is sent to Lucene, and concept matches are given score boost over word matches, thus prioritizing semantic match over merely textual match. Note that if a query contains no concepts, search functionality seamlessly falls back to conventional full-text one. Thus Blognoon provides concept search as a transparent extension over full-text search.

4.2 Facets

In conjunction with a search result, the user is presented with *facets* – concepts that are key ones for the whole content of the search result. Facets provide two features to the user: a semantical excerpt for the result as a whole and a suggestion for subsequent searches for narrowing the result space.

Facets are computationally expensive to be constructed from scratch on the fly, so we pre-build them incrementally while indexing blog posts. A map from a potential search concept to its weighted facets is memoized as FacetCache; Algorithm 1 illustrates how FacetCache is incrementally constructed as a new blogPost arrives for indexing. A call to KeyConcepts denotes obtaining weighted key concepts of the blogPost. FacetCache gets expanded with relations between the key concepts; a local variable Related associates each key concept with its similarity to c_i.

Algorithm 1 IncrementallyBuildFacets

Input: blogPost, FacetCache
Output: FacetCache

```
1: KW = KeyConcepts(blogPost)
2: for c_i ∈ KW.keys do
3:     Related = {c_j ▷ sim(c_i, c_j) | c_j ∈ KW.keys}
4:     if c_i ∈ FacetCache.keys then
5:         FacetCache[c_i] = FacetCache[c_i] ⋃ Related
6:     else
7:         FacetCache = FacetCache ⋃ {c_i ▷ Related}
```

With frequencies of key concepts generally following the power law, the size of FacetCache grows quickly only initially and stabilizes as more blog posts get indexed. We observed that for blogs in English the size of FacetCache does not

[2]http://lucene.apache.org/

exceed $250K$ keys, allowing us to store FacetCache in main memory for faster retrievals.

Concept Tips. To illustrate the particular relevance of a blog post to a search query, each result item is accompanied by concept tips – a subset of its key concepts that are semantically related to concepts in the query. Concept tips are essentially an intersection between facets and keywords:

$$\bigcup_{c_i \in \text{Query}} \text{FacetCache}[c_i].keys \bigcap \text{KeyConcepts(post)}.keys$$

Facet Views. Two kinds of presentational views are supported for facets: a list view and a clustered view. In *list view*, facets are displayed in a flat list, ranked by their relevance to a search query. In *clustered view* facets are grouped into named topical clusters according to their pair-wise semantic similarity. Clustered view is especially illustrative for a large set of facets. This allows the user to quickly observe general categories related to a search query and then go for finer-grained facets of a certain category.

Clustered view is shown in Figure 1 on the right pane. Internally, the view is represented as a map from a general concept that names a topical cluster to weighted facets that constitute the cluster. Algorithm 2 illustrates the computation of the clustered view, accepting as input weighted facets for a search query: $\text{ClusteredView}(\cup_{c_i \in \text{Query}} \text{FacetCache}[c_i])$. Topical clusters are computed from a semantic graph induced by facets using the Girvan-Newman algorithm [2]. For each cluster that is semantically dense enough to exceed a certain threshold λ, a general concept is inferred for naming the cluster using the Spreading Activation algorithm [8]. The remaining clusters are merged under an administrative concept "Misc". The threshold λ is chosen experimentally to balance between making clusters that exceed it contain reasonably related facets and keeping the "Misc" cluster moderate in size.

Algorithm 2 ClusteredView

Input: WeightedConcepts
Output: View

```
1: View = ∅
2: graph = SemanticGraph(WeightedConcepts.keys)
3: ClustersSet = GirvanNewman(graph)
4: for cluster ∈ ClustersSet do
5:    density = 1/|cluster|^2 Σ_{c_i,c_j ∈ cluster} sim(c_i, c_j)
6:    if (density > λ) then
7:       View[SpreadingActivation(cluster)] = cluster
8:    else
9:       View["Misc"] = View["Misc"] ⋃ cluster
```

What a Blog is About. To give the user an illustrative impression on a subject of a blog at a glance, we display its primary topics inferred from its content and key concepts for each topic. This data is computed with the same Algorithm 2, only invoked for weighted key concepts of the blog. Unlike conventional tag clouds, the clustered view is more structured and involves background knowledge on semantics.

4.3 Recommendation

Blognoon offers recommendations with respect to both individual blog posts and whole blogs, internally computed via concept search. When the user opens a blog post, a search query is constructed from key concepts of the post and search results are presented as relevant recommendations. In the same way, recommendations for blogs are performed using key concepts of a currently opened blog. Recommendations offered by Blognoon constitute a valuable tool for topic exploration, for they are based on non-trivial semantic relations between concepts and allow the user discover relevant material which is difficult to locate otherwise.

5. RELATED WORK

Kosmix [7] aims at providing topic exploration functionality in scope of the whole Web, by combining deep web crawl and federated search. Alternatively, Blognoon addresses only blog data, which makes the crawling approach sufficient in our case and allows us to employ sophisticated preprocessing of crawled data for providing *concept* search which includes finding information that is semantically similar to a query, not only textually similar. For semantically ambiguous words in data sources, Kosmix achieves their sense disambiguation implicitly, by either relying on particular data sources or by adding a particular meaning to a query as another word. Blognoon performs explicit word sense disambiguation for all words in data [9] as a preprocessing step.

Generation of a topic page for a query is investigated in [1], primarily for the biographical domain. Their approach to detecting multiple aspect of a topic based on term clustering is somewhat reminiscent to clustered view of key concepts in Blognoon. We however do not aim at generating a single page that would cover a whole topic, but instead offer semantic-driven navigational features to facilitate topic exploration.

6. REFERENCES

[1] N. Balasubramanian and S. Cucerzan. Topic pages: An alternative to the ten blue links. In *Proc. IEEE Int. Conf. on Semantic Computing, 2010*.

[2] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Phys. Rev. E*, 70(6):066111, Dec 2004.

[3] E. Gabrilovich and S. Markovitch. Wikipedia-based semantic interpretation for natural language processing. *J. Artif. Int. Res.*, 34:443–498, March 2009.

[4] M. Grineva, M. Grinev, and D. Lizorkin. Extracting key terms from noisy and multitheme documents. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 661–670, New York, NY, USA, 2009. ACM.

[5] M. A. Hearst, M. Hurst, and S. T. Dumais. What should blog search look like? In *SSM '08: Proceeding of the 2008 ACM workshop on Search in social media*, pages 95–98, New York, NY, USA, 2008. ACM.

[6] R. Mihalcea and P. Tarau. TextRank: Bringing order into texts. In *Proceedings of EMNLP-04 and the 2004 Conference on Empirical Methods in Natural Language Processing*, July 2004.

[7] A. Rajaraman. Kosmix: high-performance topic exploration using the deep web. *Proc. VLDB Endow.*, 2(2):1524–1529, 2009.

[8] Z. Syed, T. Finin, and A. Joshi. Wikipedia as an ontology for describing documents. In *Proceedings of the Second International Conference on Weblogs and Social Media*. AAAI Press, March 2008.

[9] D. Turdakov and P. Velikhov. Semantic relatedness metric for Wikipedia concepts based on link analysis and its application to word sense disambiguation. In *SYRCoDIS*, 2008.