RESEARCH PAPERS OF WROCŁAW UNIVERSITY OF ECONOMICS No. 104

Business Informatics 16 2010

**Mikołaj Morzy**
Poznań University of Technology, Poznań, Poland
Mikolaj.Morzy@put.poznan.pl

# INFLUENCE IN THE BLOGOSPHERE

**Abstract:** In this paper we turn our attention to the fascinating problem of mining and ranking of blogs. We present a brief introduction to the problem of blog ranking and we summarize the related work on the subject. We focus on discovering the influence relation between blogs and blog posts in order to identify the most influential blogs using the context of a user's query. To do so, we transform blogs and blog posts into dual feature term-theme vectors representation. We use term vectors to align candidate blogs with the user query and we rank themes appearing in relevant vectors as possible answers. We also report the results of preliminary experiments conducted on our algorithm, which indicate that our algorithm is capable of delivering interesting and valuable answers.

**Keywords:** blogs, data mining, influence, Web 2.0.

## 1. Introduction

Blogs are ubiquitous. Today, millions of Internet users are blogging about products they have bought, movies they have seen, books they have read, bands they have listened to, etc. Traditional blogging platforms are constantly enhanced, and new opportunities emerge. From the point of view of social network analysis, the entire blogosphere can be defined as a distributed network of user opinions and reviews of brands, companies, products, books, movies, music albums, etc., published on the Web. Apart from corporate blogs, the most influential blogs (and thus, quickly spreading opinions) are created by independent individuals. Grassroots journalists report on new ideas, products, brands, and people. Opinions circulating in the blogosphere can be either positive, increasing the reputation of a product, person, or company, or negative, effectively hurting the reputation of a given subject. Extraction of opinions from credible blogs is very important to numerous enterprises operating both on the Web and in the real world. The reputation of an enterprise can be quickly built using viral marketing techniques, but equally easily it can be destroyed by a bunch of angry bloggers. The amount of customer feedback published in the blogosphere is immense, and modern enterprises are beginning to notice this source of inspiration and innovation. The main goal of the research presented in this paper is to unearth opinions and reviews of bloggers on particular subjects, and to rank these opinions according to the reputation and credibility of blogs.

Analyzing blog networks has recently become a hot topic among academic, research, and commercial communities. Most vocal signs of this trend are the emergence of feed aggregators (such as Bloglines or Google Reader) and specialized blog search engines (such as Technorati or Google Blog Search). The main drawback of current solutions is the fact that most of these solutions present simple adaptations of algorithms and methods employed in the Web to the realm of the blogosphere. For instance, the search through Google Blog Search is performed using a slightly modified version of the PageRank algorithm. Similarly, Technorati blog search engine uses a PageRank-like algorithm to rank blogs based on their relative importance in the blogosphere, where the importance is defined by in-links from other blogs. Unfortunately, such simple adaptation of algorithms and techniques from the Web to the blogosphere is likely to omit important characteristics of the blogosphere. For example, many studies show that the blogosphere is significantly sparser than the Web and the distribution of in-links and out-links is far more skewed. Blogs are more separated and links between blogs are less frequent. In addition, links between blogs bear different weights. Blogs mentioned in the blogroll are explicitly acknowledged as authorities by the author of a given blog. A blog linked from an old and outdated post is merely an anchor for a quotation. Finally, links contained in comments under the post may result from spamming and cannot be considered as the acknowledgment of quality of the linked blog. Therefore, the blogosphere needs new algorithms and processing techniques that take into consideration distinct characteristics of this medium.

Analyzing the blogosphere is a difficult and challenging task. Many features account for this difficulty. Blogs, similarly to Web pages, are written using different languages, styles, and vocabularies. Blog publishing engines employ different formats, so crawlers for fetching blog posts have to be tailored to individual engines. Evolutionary nature of blogs requires constant monitoring and updating of blog contents as new posts are published asynchronously. Our approach to blog analysis and knowledge extraction from the blogosphere combines two basic ideas: extraction of named entities from blog posts and ranking of blogs according to their reputation. We purposefully refrain from using advanced text mining techniques due to the size of the blog database and efficiency concerns. Instead, we are trying to derive useful knowledge using only the simplest text processing techniques that add negligible processing overhead. Named entities are extracted using a simple heuristic and applying a set of filters for named entity recognition. Ranking of blogs reflects the relative importance of a blog within the blogosphere. The importance of a blog is computed based on the influence relation between blogs. A token of influence is either a hard link (i.e., an explicit hyperlink in the blog post or in the blogroll), or a soft link (i.e., a quotation, both attributed and unattributed, or semantic similarity between posts). The existence of the influence relation between blogs is equivalent to the notion of trust appearing between blogs.

The organization of the paper is the following. The related work on the subject is summarized in section 2. Section 3 contains basic definitions used throughout the paper. The formulation of the research problem and the presentation of the blog influence discovery algorithm is contained in section 4. In section 5 we report on the results of experiments conducted on the real world dataset crawled from the Web. Finally, the chapter concludes in section 6 with a brief summary and the outline of the future work agenda.

## 2. Related work

Blogs have attracted the attention of researchers and practitioners from the very advent of the blogosphere. The research conducted on analyzing and mining of blogs builds on experiences from multiple scientific areas and provides useful results for domains so diverse as social network analysis, economy, public relations, and marketing. Main directions in blog mining include finding spam blogs, identifying influential blogs, following the dissemination of information in the blogosphere, ranking of blogs, community discovery, and measuring macro-measures of the blogosphere. A thorough survey of blog mining problems can be found in [Facca, Lanzi 2005].

The problem of community discovery in the blogosphere is discussed by Chin and Chignell [2006]. Their solution consists in turning links between blogs into social hypertext and integrating network analysis with social view of the community to discover structures characteristic for small communities. Discovered communities of interconnected blogs are then projected onto centrality measures used in social network analysis and compared with explicit community measures received from surveying blog authors about their subjective perception of the degree of community. A more automatic method is presented in [Chaua, Xu 2007], where a complex four-tier model for community detection is presented. Authors use a combination of a blog spider, information extraction, network analysis, and visualization, to demarcate almost thirty racist hate groups hosted on Xanga, a popular blog hosting site. Publications on the Polish blogosphere are sparse, with the exception of [Bachnik et al. 2005], where the authors collect a large body of Polish blogs and characterize the Polish blogosphere using several measures of social network analysis to prove the existence of social structures in the data. Finally, Li and Liu [2005] present a method for community discovery based on co-occurrences of named entities in blog posts. Their solution is to use frequent combinations of named entities as markers of communities that form around particular subjects, and their results can be easily generalized to traditional Web pages or free text. Named entities discovered in blog data can be also used for event identification. Suhara, Toda and Sakurai [2008] present a method that uses related named entities occurring in a narrow time window to discover echoes of important events.

Another direction of blog research is the acquisition of blog data. A sophisticated system for monitoring, harvesting and mining of Japanese blogs is presented in [Nanno et al. 2004]. The system handles blog updates and processes not only blogs created using popular blog publishing engines, but hand-crafted blogs as well. The system relies heavily on processing date expressions and complex parsing of HTML contents of blog posts. A very interesting system, called BlogHarvest, is presented in [Joshi, Belsare 2006]. BlogHarvest is an entire blog mining framework that employs link analysis, classification, topic-based clustering, and part-of-speech tagging for opinion discovery. BlogHarvest collects blog data and extracts descriptions of interests of a blogger, provides recommendations on similar blogs, and allows to query and browse collected blogs. Another system, called BlogRanger, is presented in [Fujimura et al. 2006]. The system provides different search interfaces, such as topic search, blogger search, and reputation search. A survey of blog searching techniques can be found in [Mishne, De Rijke 2006].

Successful analysis of blog posts requires a robust method of spam identification and pruning. Java et al. [2007] present a method for noise reduction in blog data. Authors note that blogs are often poorly structured, written using informal language and vocabulary, riddled with grammatical and spelling errors. In addition, large parts of blog posts are irrelevant for the informative contents of the post, e.g., blog rolls, navigation links, link rolls, advertisement banners, sidebars, or superfluous contents generated by blog publishing engines. Another very interesting idea is to identify spam blogs based on their URL address analysis [Salvetti, Nicolov 2006]. A technique which segments long tokens into words forming phrases is evaluated in the paper. The resulting tokens are used as features for the blog classifier for spam filtering.

Automatic analysis of blog post content, in particular, discovering and aggregating user opinions, reviews and sentiments, is of crucial importance for marketing and public relations. Dave, Lawrence, Pennock [2003] introduce a method for opinion extraction and semantic classification of product reviews. The presented classifier distinguishes between positive and negative product reviews using feature extraction, feature scoring, and a set of heuristics which vary depending on the testing situation. A fusion approach to opinion mining is presented in [Yang et al. 2007]. The authors propose to partition the problem of opinion mining into two sequential tasks: the selection of on-topic blogs using traditional information retrieval methods followed by the rank boosting using opinion assessment methods. These methods include counting the number of occurrences of opinion terms, identifying rare opinion terms, I-and-you collocations, and using computational linguistics' distribution similarity approach to learn the subjective language from the training data. Interesting research on mining opinions from a narrowly defined topic, i.e., blogs on law and codification, is presented in [Conrad, Schilder 2007]. Finally, in the highly acclaimed research [Kempe, Kleinberg, Tardos 2003] present a framework for maximizing the influence on the social network by careful selection of individuals to be targeted. The authors

present two diffusion models according to which information spreads across the network, and they compute approximation guarantees for a simple greedy algorithm. A similar problem is researched by Gruhl et al. [2004], but the authors consider only the blogosphere as the environment for information diffusion. The authors distinguish between long-running topics and spike topics, the later resulting from outside world events, and present infection-based models for microscopic propagation of subjects from one individual to another.

The vast majority of models for blog analysis and blog mining build upon the famous vector space model of text documents [Salton, Wong, Yang 1975], also referred to as the term frequency-inverse document frequency model. Other models designed in the area of blog mining cope with the entire blogosphere. For instance, in [Java et al. 2006] a model for the spread of influence in the blogosphere is presented, which is further extended to the framework for modeling influence, opinions, and structure of social media [Java 2007]. Agarwal et al. [2008] make an interesting transition from blog modeling to blogger modeling. They recognize the features characterizing an influential blogger, present a preliminary model to quantify the influence of a blogger, and define various types of influence.

Several research efforts have been recently undertaken to unearth processes that build social networks. Backstrom et al. [2006] present models for group formation in social networks. The authors analyze structural features of groups within a social network and relate these features to group membership, group growth, and group evolution. Leskovec, Kleinberg and Faloutsos [2005] discuss models for large social network evolution over long periods of time and discover surprising regularities in the graph evolution. First, social networks become denser over time with the number of edges growing super-linearly in the function of the number of nodes. Secondly, the average distance between nodes in the social network shrinks over time as more nodes join the network. The authors also introduce two models of graph generation and evolution: the community guided attachment model and the forest fire model, with the later model generating graphs that exhibit all properties of the large real-world social networks, including the shrinking diameter of the network and heavy-tailed distributions of in-degree and out-degree of nodes. A similar research is presented by Kumar, Novak and Tomkins [2006] who discover a pattern of segmentation of popular online social networks into separate regions of singletons, isolated communities with star structure, and a central component with densely connected core. The authors also propose a simple graph generation model that produces networks displaying similar properties as the real-world networks. In [Kossinets, Watts 2006] the authors analyze a large social network comprising students, faculty and staff exchanging e-mails, and they discover an interesting phenomenon: although individual properties of network members and small communities within the network tend to be highly unstable, the average properties of the entire network seem to approach an equilibrium state. A thorough overview of challenges in mining social networks, as well as relevant bibliographic references, are presented in [Kleinberg 2007].

## 3. Basic definitions

In this section we introduce the basic definitions used throughout the paper and we state the formulation of the problem. First, we define a blog. There is no single definition of a blog due to the variety of blog publishing formats and engines. Informally, a blog is a Web-based publication consisting of short entries displayed in the reversed chronological order. For the purpose of our research we employ the following definition.

A *blog* is an ordered sequence of *blog posts*. Each blog post is a time-stamped text document that may contain embedded content (image, audio, video) and hyperlinks, either to other blog posts, or to external sources. Thus, a blog $B$ can be represented as

$$B = \langle (b_1, t_1), (b_2, t_2), \ldots, (b_n, t_n) \rangle,$$

where $b_i$ denotes a blog post, $t_i$ is the timestamp of the blog post $b_i$, and for all $i$ $t_{i-1} < t_i$.

A raw blog post is not suitable for automatic processing, so it is transformed into a feature vector. Given a set of features $F = \{f_1, f_2, \ldots, f_m\}$, each post $b_i$ is represented as a vector $\overline{b_i} = [w(f_1, b_i), w(f_2, b_i), \ldots, w(f_m, b_i)]$, where $w(f_j, b_i)$ is the weight associated with the feature $f_j$ appearing in the blog post $b_i$. Different algorithms utilize various types of features, and the most common feature model is the bag-of-words, where canonical forms of words appearing in blog posts are used as features. For weighting features across blog posts we use the traditional TF-IDF (term frequency-inverse document frequency) measure defined as

$$w(f_j, b_i) = n(f_j, b_i) * \log_2 \frac{|D_B|}{|\{b_k \in D_B : f_j \in b_k\}|},$$

where $n(f_j, b_i)$ is the number of occurrences of the feature $f_j$ in the blog post $b_i$, $D_B$ denotes the database of blogs, and the denominator of the fraction represents the number of blog posts that contain the feature $f_j$. For a given feature $f_j$ the term under the logarithm is constant and it represents the inverse frequency of the feature in the blog database (i.e., a feature appearing in many blog posts receives a lower score, while a feature appearing rarely receives a higher score).

Based on the TF-IDF measure we define the similarity between blog posts $b_r$ and $b_s$ as

$$sim(b_r, b_s) = \frac{\overline{b_r} \cdot \overline{b_s}}{\|\overline{b_r}\| * \|\overline{b_s}\|},$$

where $\overline{b_r} \cdot \overline{b_s}$ denotes the dot product of feature vectors compared for their similarity, $\|\overline{b_r}\| * \|\overline{b_s}\|$ is the product of vector lengths, and $\|\overline{b_r}\| = \sqrt{\sum_{f_i \in b_r} w(f_i, b_r)^2}$ is the Euclidean norm of the vector $\overline{b_r}$.

Processing of blog posts, identification of named entities, and matching a query with blog posts stored in the database, requires the ability to perform fuzzy matching of features. Features may be misspelled or different flexical forms of a feature may exist. For example, many languages utilize declination of verbs, and some languages use conjugation of nouns, adverbs, and adjectives. Computing the canonical form of a word is possible for dictionary words using popular language processing tools. Unfortunately, the most interesting features representing proper names, brand names, product names, etc., often cannot be trimmed to their canonical form easily. Therefore, we choose to use *n*-grams to compute the similarity between words.

An *n-gram* of a word is an *n*-element subsequence of consecutive characters contained entirely in the word. An *n*-gram of the length 1 is called a *unigram*, an *n*-gram of the length 2 is called a *bigram*, and an *n*-gram of the length 3 is called a *trigram*. For instance, the set of all trigrams of the word *blogosphere* is {*blo*,*log*,*ogo*, *gos*,*osp*,*sph*,*phe*,*her*,*ere*}.

## 4. Blog influence discovery

Having defined the blog post, its vector representation, and the measure of blog post similarity, we proceed to the formalization of the research problem. Given a blog post database defined as

$$D = \{(u_1, ts_1, B_1), (u_2, ts_2, B_2), \ldots, (u_n, ts_n, B_n)\},$$

where $u_i$ denotes the unique resource locator (URL) of the *i*-th blog, $ts_i$ is the timestamp at which the *i*-th blog has been retrieved from the blogosphere, and $B = \langle (b_{i_1}, t_{i_1}), (b_{i_2}, t_{i_2}), \ldots, (b_{i_m}, t_{i_m}) \rangle$ is the blog contents stored as the sequence of timestamped blog posts. All blog posts are transformed into dual feature vectors. A dual feature vector $\overline{b_i} = [term(b_i), theme(b_i)]$ consists of two independent feature vectors:

- *term feature vector term*($b_i$): contains the vector representation of terms (verbs, adjectives, common nouns) and defines the context or topic of the blog post $b_i$, e.g., astronomy or politics,
- *theme feature vector theme*($b_i$): contains the vector representation of themes (proper nouns) and defines named entity references, i.e., the subject of the blog post $b_i$, e.g., Alpha Centauri or Barack Obama.

The main reason for introducing the dual feature vector representation of blog posts is the ability to observe and analyze particular subjects discussed within more general topics in the blogosphere. As an example, consider a person interested in finding young and promising contemporary zydeco artists (zydeco is a form of folk music of black and multiracial French speaking Creoles of south and southwest Louisiana). Traditional search engines are not suitable for this task for a variety of reasons. First of all, ranking algorithms favor popular sites with many important in-

links, and these are usually major news sites and knowledge repositories. Sometimes, as is the case with the query *contemporary zydeco musicians*, the search engine directs to a fan site that works as a hub with many information and links to artists' websites. Unfortunately, this results in a static view of the Web. If the fan site in question is not updated on a regular basis, the information contained therein quickly becomes obsolete. A worse situation arises when the alignment of the query to the expected result is weak. For the query *promising zydeco musicians* the first hit from Google directs to a *New York Times* article on zydeco music published in 1987! On the other hand, recall that the blogosphere is highly dynamic and quickly responds to new developments and events in the real world, effectively presenting the dynamic view of the Web. If a music enthusiast encounters an interesting young band playing zydeco in one of the numerous bars along the Bourbon Street in New Orleans, and if she happens to be a blogger, the description, impressions and opinions on the concert are likely to appear in the blogosphere the day after. The question remains how to find that particular blog when searching for *promising zydeco musicians*.

Our algorithm uses the blogosphere contents to discover hot themes (people, products, movies, brands) discussed by authoritative representatives of narrowly defined communities. The algorithm performs browsing of the blogosphere, as opposed to searching the blogosphere. The main difference between browsing and searching consists in the direction, in which the process is performed. Searching may be visualized as a vertical process in which vast amounts of information are drilled with pinpoint keyword queries. In contrast, browsing may be visualized as a horizontal process in which the user demarcates the topic of interest using broad terms (such as politics, contemporary jazz music, or ancient Chinese poetry), but the goal of browsing is not known *a priori*. Browsing resembles exploratory search, where a particular region of the blogosphere defined by the context query is examined in pursuit of interesting themes, but the criteria for interestingness are defined vaguely. Often, the user does not know what she is searching for. Browsing is more difficult to implement than searching due to the vagueness of criteria for result quality evaluation. Therefore, as much knowledge as possible must be extracted from the blogosphere to decide, which of the themes present in blog posts might be interesting to the user.

The algorithm is based on the assumption that bloggers are early adopters, eager to share their opinions, experiences and expertise with the community. When a new theme appears (be it a new product, a new musical band, a new writer, a new actor, etc.), bloggers are highly likely to be the first to discover the theme, judge it, and provide authoritative opinions on the theme. The algorithm for mining the blogosphere in contextual browsing for new promising themes is presented in Figure 1. First, term feature vectors are aligned with the query in order to find the seed set $C$ of candidate blog posts on a particular topic. Then, blogs containing discovered blog posts are ranked according to the importance measure computed using one of many available algorithms, e.g., PageRank (the algorithm is not bound to a specific

algorithm for ranking the importance of nodes within the network, so other algorithms may be applied, e.g. HITS, SALSA, or Okapi BM25). For each blog post from the set of candidates $C$, its theme feature vector is retrieved and all named entities contained in the theme feature vector are added to the answer set $A$. Finally, discovered theme features (i.e., named entities mentioned in blog posts relevant to the query) are ranked using their frequency and the relative importance of blogs that reference these named entities. The resulting list consists of proper nouns (hopefully, names of people, places, titles of books, albums and songs, etc.) that represent the community buzz or hot themes mentioned frequently in the blogosphere.

**Require**: $D$, the database of blog posts
**Require**: $q$, the user query
1: $C = \{b_i \in D : sim(term(b_i, q)) \geq \alpha\}$
2: **for all** blogs $B : b_i \in C$ **do**
3: compute *weight*($B$)
4: **end for**
5: **for all** $b_i \in C$ **do**
6: $A = A \bigcup theme(b_i)$
7: **end for**
8: output $h_j \in A$ ordered by descending values of *weight*($h_j$)

**Figure 1.** Algorithm for contextual browsing of the blogosphere

The key aspect of the algorithm is the dual feature vectors technique. It allows to separate the first phase of the search, in which the context of the query is defined by the means of the term feature vector comparison, from the second phase, in which discovered theme features are ranked using their frequency and the importance of blog posts that mention these theme features (the details concerning the ranking of blogs are presented further). When defining the context of the search, the user provides pinpoint keyword query that is transformed into the term feature vector and compared with term feature vectors of all blog posts in the database using previously defined similarity measure. The sole parameter of the algorithm is the similarity threshold $\alpha \epsilon \langle 0,1 \rangle$ used to select blog posts relevant to the query. Higher values of the similarity threshold $\alpha$ restrict the seed set of relevant blog posts to only the most similar blog posts, whereas lower values of the similarity threshold allow more blog posts in the seed set.

One crucial difference between ranking of hypertext documents in the Web and ranking of blogs in the blogosphere is the density of the linkage. Many studies show that the linkage between blog posts is significantly sparser than the linkage between traditional documents in the Web. Because ranking algorithms rely heavily on explicit links between documents to compute the relative importance of documents, a straight application of these algorithms to the blogosphere results in highly biased rankings. Blogs are interconnected using explicit links differently than traditional documents.

Firstly, links contained in the blogroll (a list of other blogs endorsed by the blogger and displayed prominently, usually in the sidebar of the blog) contain a very strong statement of praise. Such links should be used with strong weights when computing the relative importance of the blog. Secondly, blog posts do not have to necessarily link to each other, but may be connected via an intermediary webpage. Sharing out-links implies that blog posts discuss the same topic or event. The chronological order of blog posts may be used to find the direction of influence and to deduce the existence of a relationship between two blog posts. The second possibility of indirect blog post linkage is a specific example of a more general relation of influence between blog posts which we describe next.

Informally, a blog post $b_p$ is said to influence a blog post $b_q$ if the two blog posts are similar accordingly to the previously defined similarity measure, and the blog post $b_p$ chronologically precedes the blog post $b_q$, with no requirement for a hard link between the blogs. Certain bloggers are considered experts in their domain. These people are early adopters of new products, enthusiasts of new artistic trends, or technology gurus eager to test new software. Experts also tend to gather and accumulate information on a given topic in order to share this knowledge with the community. In almost every domain such experts exist and, usually, these people are known to other bloggers. Since experts are, in general, a very credible source of information and opinion, their blog posts often influence and inspire other bloggers. Sometimes, the influence is subtle, prompting other bloggers to investigate a subject, rethink their positions, or simply turn people's attention to the subject. Sometimes, the influence is so strong that the original blog post is simply copy-pasted. Sadly, many blogs exist with the sole purpose of echoing others thoughts. Incorporating this influence into the blog ranking algorithm allows to reward expert bloggers for their influence on the community. Of course, chronological precedence alone does not have to indicate direct influence, for instance, two bloggers might have learned about an event independently. For this reason we use blog post similarity measure to identify the existence of an influence relation between blog posts. Certainly, a hard link pointing from the blog post $b_q$ to the blog post $b_p$ is the ultimate proof of the influence, but, as has been already said, bloggers have inclination towards careless treatment of authors' rights and may omit citing the source of information.

Let $Out(b_q)$ denote the set of out-links from the blog post $b_q$ and let $\rightarrow$ denote the relation of influence. Formally, the reflexive relation of influence between two blog posts $b_p$ and $b_q$ is defined as

$$b_p \rightarrow b_q \Leftrightarrow sim(b_p, b_q) \geq \beta \wedge t_p < t_q \wedge (Out(b_p) \cap Out(b_q) \neq \varnothing \vee b_p \in Out(b_q)),$$

i.e., the blog post $b_q$ is influenced by the blog post $b_p$ if it is similar to $b_p$ in the feature vector space, chronologically succeeds $b_p$, and either shares out-links with $b_p$, or explicitly links to $b_p$. Note that the influence relation is neither symmetric, antisymmetric, asymmetric, transitive, nor total. The relation of influence may be

quantified to measure the degree of the influence. In our experiments we have used the following formula:

$$influence(b_p, b_q) = \begin{cases} w_1 * sim(b_p, b_q) + w_2 * \dfrac{|Out(b_p) \cup Out(b_q)|}{|Out(b_p)|}, b_p \notin Out(b_q) \\ 1 \ otherwise \end{cases}$$

with the constraint on $w_1 + w_2 = 1$. Thus, $influence(b_p, b_q) \in \langle 0,1 \rangle$ and it reaches its maximum value when the blog post $b_q$ explicitly links to $b_p$, otherwise, it uses a weighted sum of feature vector similarity and out-link similarity.

The influence relation defines the relationships on the level of blog posts, but it can be easily aggregated to the level of blogs. The relation of influence between blogs $B_p$ and $B_q$ is defined as

$$B_p \rightarrow B_q \Leftrightarrow \exists b_p \in B_p, b_q \in B_q : b_p \rightarrow b_q$$

and it can be quantified as

$$influence(B_p, B_q) = \sum_{b_p \in B_p, b_q \in B_q} influence(b_p, b_q).$$

The influence measure defined above is used to discover relative importance (and thus the rank) of each blog in the blog database $D$. Let $G = (E, V)$ be the graph representing the mutual influences of blogs in the blog database $D$. The nodes of the graph are blogs stored in the database, $E = \{B_1, B_2, \ldots, B_n\}$, and an edge exists between any two nodes that participate in the influence relation, $V = \{(B_i, B_j) : B_i, B_j \in E \wedge influence(B_i, B_j) > 0\}$. Any node ranking algorithm can be used on the graph $G$ to compute the relative importance and ranking of the nodes. Theme features are returned as the result of a query to the user in the order defined by the relative importance of the blog, from which a theme feature originates. Instead of using linear weighting of blogs with respect to their rank returned from the node ranking algorithm, we choose to use a non-linear weighting function defined as follows:

$$weight(B_i) = 1 - \frac{rank(B_i)}{n * \log_{10}(n - rank(B_i))}$$

where $rank(B_i)$ is the rank of the $i$-th blog computed by a node ranking algorithm, and $n$ is the number of blogs in the blog database $D$. Figure 1 depicts the weighting function for $n = 100$. The purpose of the weighting function is to smooth the bias introduced by the node ranking algorithm.

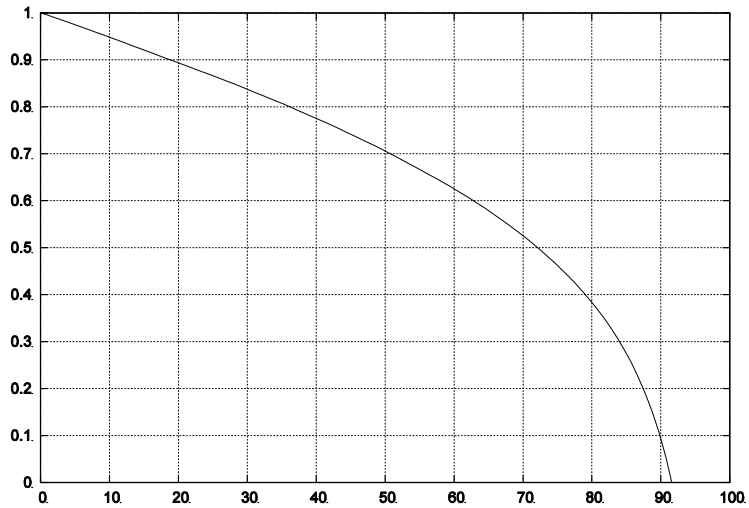


**Figure 2.** Blog ranking function $weight(B_i)$

Finally, we can define the order in which theme features are returned as the answer to the user query. Theme features are ordered by descending values of their weights defined as

$$weight(h_j) = \sum_{b_i : h_j \in theme(b_i)} weight(B_i),$$

i.e., the weight of the theme feature $h_j$ is the sum of weights of blogs that contain the given feature. Note that the summation is performed over blog posts, so if a given blog mentions a theme feature multiple times, each occurrence of the theme feature contributes to the final weight independently.

## 5. Experiments

In this section we present the results of initial experiments performed on blogs processed by the our blog processing framework. All experiments were performed on a computer equipped with Pentium 4 CPU and 512 MB of RAM. The first experiment displays the time overhead involved in processing raw blog entries and transforming them into relational representation. The results of the experiment are printed in Figure 3 and the data is contained in Table 1.

Processing raw blogs is a repetitive task scheduled to be run periodically as a job. Therefore, it is paramount to know the width of the processing window in order to schedule raw blog processing execution at intervals that do not result in job queuing. In other words, one must know the time required to process a given batch of raw

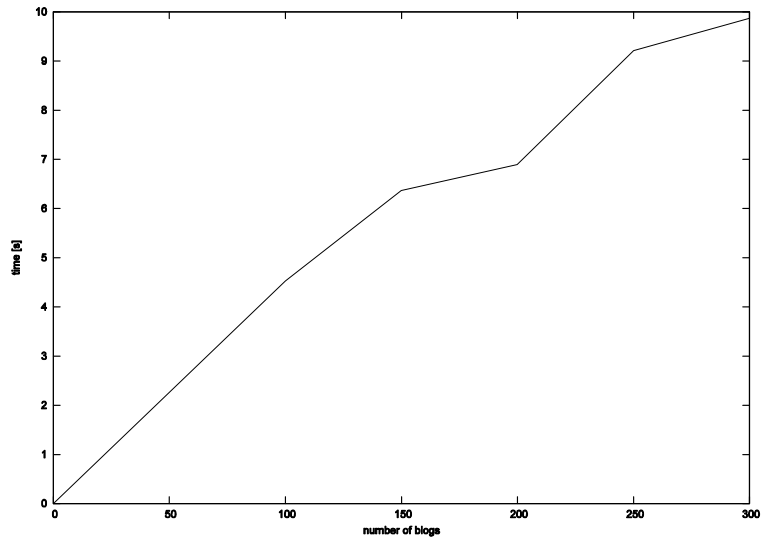


**Figure 3.** Raw blog processing time

blogs to schedule the frequency of processing. In the experiments, small batches of 100, 150, 200, 250, and 300 blogs were randomly chosen from the raw blog database. For each sample size the procedure for raw blog processing was run 10 times. From the figure follows that the processing time is roughly linear with respect to the sample size. The differences in blog sizes (i.e., in the number of blog posts and average lengths of blog posts) account for variations in the processing time, but for a sample of 300 raw blogs the worst case processing was below 13 seconds. We conclude that 1 minute interval as the frequency of raw blog batch processing can be safely chosen as it guarantees that no job queuing occurs. In addition, the processing time of raw blogs is shorter than the time needed by the web crawler to gather new blogs, so filtering blogs is not the bottleneck of the system.

The next experiment measures the number of blog posts in the raw blog sample. Knowing the average number of blog posts within a blog is crucial for setting the

**Table 1.** Raw blog processing time

| Sample size | Average processing time | Standard deviation |
|---|---|---|
| 100 | 4.5282 | 1.3424 |
| 150 | 6.3702 | 2.6403 |
| 200 | 6.8953 | 1.9898 |
| 250 | 9.2140 | 3.7883 |
| 300 | 9.8656 | 2.5647 |

update frequency. Posting habits of bloggers may differ significantly, some bloggers tend to post several short posts each day, some bloggers aggregate their posts to cover a longer period of time, e.g., a week. The update frequency should be set to an interval such that the overlap between consecutive checks is minimal, but, at the same time, the probability of loosing blog posts is low. For example, if the length of the blog feed is set to 10 posts, and a blogger posts one blog post per day, then updating this feed once a week is sufficient to read all new blog posts and reduce the overlap, because on average, upon each blog feed update, three posts out of ten posts will be processed superfluously. The experiment is performed on the same samples as the previous experiment, i.e., samples of the size 100, 150, 200, 250, and 300 are selected. Results of the experiment are presented in Figure 4 and the data is provided in Table 2.

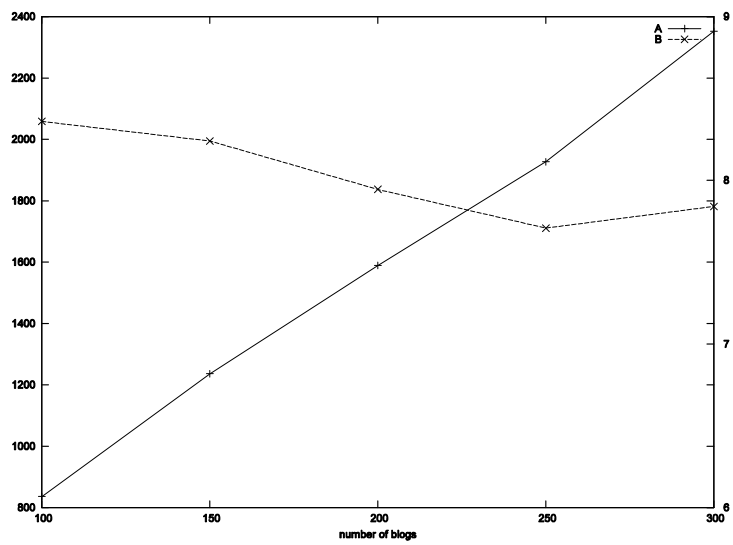


**Figure 4.** A – number of blog posts, B – average number of blog posts with respect to the number of blogs

**Table 2.** Blog posts, features, themes, and storage sizes

| Sample | Posts | Features | Themes | DB (feat.) | DB (th.) |
|---|---|---|---|---|---|
| 100 | 836 | 58 769 | 7 787 | 512 | 88 |
| 150 | 1236 | 85 477 | 11 435 | 704 | 128 |
| 200 | 1589 | 112 338 | 14 778 | 896 | 192 |
| 250 | 1927 | 140 240 | 17 694 | 1152 | 192 |
| 300 | 2352 | 167 769 | 21 612 | 1344 | 256 |

Figure 4 shows that the number of blog posts grows linearly with respect to the number of blogs. The average number of blog posts per blog is close to 8, with minor variability. This result encourages using a fixed update frequency interval for most blogs, with only exceptional blogs being refreshed more frequently or less frequently. We welcome this result with content, because it greatly simplifies the management of the Web crawler update module.

The next experiment measures the number of features and themes. Figure 5 presents the results, and the data are given in Table 2. The number of features and themes defines the sizes of vector spaces (i.e., the number of dimensions in feature vectors and theme vectors, respectively). Intuitively, a large number of features and themes discovered in blogs leads to larger feature vectors, and, consequently, to longer processing times. Figure 5 shows clearly, that the increase in the number of features and themes is linear with respect to the size of the sample. In addition, the number of themes is significantly smaller than the number of discovered features. This result is somehow counter-intuitive, because one would expect that the number of features should quickly reach the saturation point. We believe that the result of the experiment can be explained by two factors. Firstly, analyzed samples are relatively small and 300 blogs definitely do not exhaust the richness of the language. Secondly, analyzed blogs were primarily drawn from the Polish blogosphere. The characteristics of the Polish language (most notably, the existence of declination and complex conjugation) make stemming difficult and many features might be superfluous.

In the last experiment we have analyzed the storage requirements for feature vectors and theme vectors. Figure 6 shows the increase of the number of 8KB disk

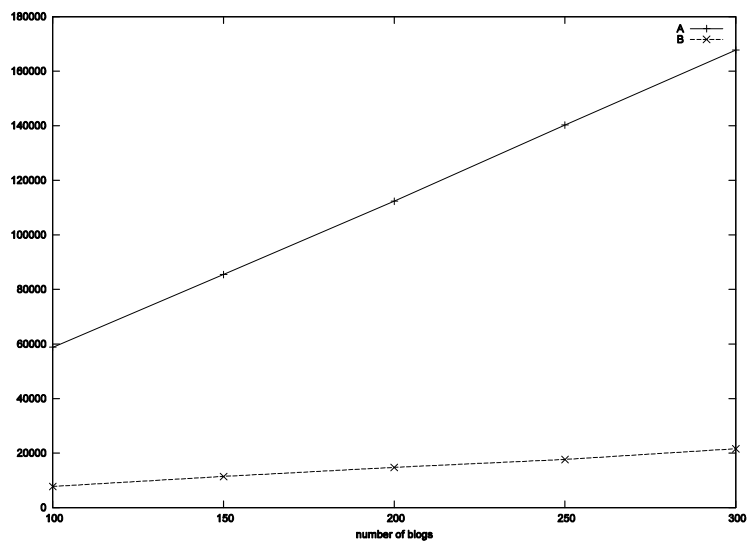


**Figure 5.** A – number of features, B – number of themes with respect to the number of blogs

blocks needed to store discovered feature and theme vectors in the relational database. The increase in the number of feature and theme tokens discovered in blogs causes storage requirements to grow fast. In the implementation, discovered feature and theme tokens are physically stored as index-organized tables (IOT) with minimal storage overhead. Nevertheless, the number of disc blocks required to store feature vectors and theme vectors is significant. For instance, storing only 300 blogs after transformation to vector representation requires 12.5 MB of disk space.

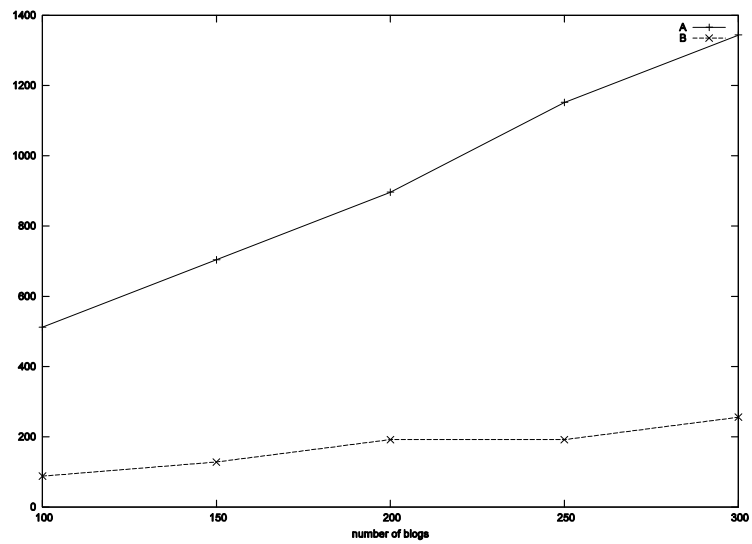


**Figure 6.** A – number of 8KB disk pages for features, B – themes with respect to the number of blogs

To conclude the experiment section, we present the example of results returned from the algorithm. A set of 200 blogs have been collected and processed, resulting in 4000 blog posts. The majority of processed blogs originated from the domain of blogs.oracle.com, or were linked to from an Oracle corporation blog. Apparently, Oracle corporation blogs have a predefined number of blog posts visible through the RSS channel set to 20 blog posts, so the average number of blog posts per blog was significantly higher than the average computed from a random part of the blogosphere. Table 3 presents the top 30 themes obtained for the query *database*, *oracle*, ordered by the blog ranking algorithm.

Obviously, the theme processor is not perfect and its heuristics sometimes lead to erroneous results. For instance, the processor uses consecutive capitalized words as a strong indication of a named entity. Words *labour* and *party* are common nouns, but *Labour Party* is a theme and should be treated as such. Unfortunately, capitalizing words may mislead the processor as shown by theme at position 2, where *Memory Management* is recognized as a theme. It is difficult to judge to which extent this behavior of the processor is wrong, because *Automatic Memory Management* is ac-

**Table 3.** Ranked results for the query *database*, *oracle*

| Pos. | Theme | Pos. | Theme | Pos. | Theme |
|---|---|---|---|---|---|
| 1 | Inside Oracle | 11 | Automatic Memory Management | 21 | Head Quarters |
| 2 | Memory Management | 12 | Oracle University | 22 | Steven Feuerstein |
| 3 | Sonnenberg | 13 | DBAs | 23 | MS-Word |
| 4 | Steven | 14 | Jean-Francois Verrier | 24 | VPN-based |
| 5 | SQL Server | 15 | RAC | 25 | Oracle Apps |
| 6 | Emulators | 16 | Licensing Oracle Clusterware | 26 | Better Than Oracle |
| 7 | Unix | 17 | Oracle PreSales | 27 | THC |
| 8 | BCHR | 18 | Single Instance Oracle Database | 28 | Leyla Alpaslan G |
| 9 | TCP-C | 19 | Berlin | 29 | Oracle University Turkey |
| 10 | Eddy Awad | 20 | Oracle University Germany | 30 | July On July |

tually a theme (AMM is an option to the Oracle database server). A similar rule may apply to movie titles, book titles, music albums and songs, product and brand names, and many more. Table 3 also reveals several people's names discovered as themes. Again, it is a purposeful choice of the author to treat last names as themes. There are three suspicious themes at positions 3, 4, and 22, namely *Steven*, *Sonnenberg*, and *Steven Feuerstein*. Steven Feuerstein is a famous PL/SQL guru, an Oracle evangelist, a popular blogger, and a highly acclaimed expert in SQL programming. Some Oracle-related blogs were discussing his seminar held for Oracle University Switzerland in Sonnenberg convention center in Zurich.

## 6. Conclusions

In this paper we have presented the entire framework for mining the blogosphere in search for interesting and emerging themes. Our solution relies on the dual representation of document term vectors, namely, in the distinction between features and themes. Features are used to provide context for the query, whereas themes are returned as the answer to the query. The algorithm works under assumption that influential and active bloggers are the first ones to discover and report interesting themes. To better rank the results we have introduced the relation of influence between blog posts, and we have aggregated the influence relation to entire blogs. Quantitative measures of the strength of the influence were introduced. Next, we have discussed general architectural issues pertaining to the blog processing framework. The chapter concludes with the description of initial experiments conducted on the gathered samples of blogs.

Still, many research challenges remain unsolved. Our prototype opens several promising directions for further research. We outline some of them below. Filters implemented in the framework are rudimentary, and much work is required to prepare robust and flexible anti-spam filters. Fighting spam is of crucial importance due to significant overhead introduced by the need to process junk blogs. As we have mentioned before, theme recognition heuristic is inchoate at this stage of research and will be improved to correctly handle most common errors. Our blog ranking algorithm is also incipient and requires further improvement. Although we are certain that the basic assumptions underlying the algorithm are correct, in particular, we firmly uphold the notion of the influence relation, the algorithm needs to be trained on large real-world datasets consisting of various blogs to properly rank blogs. A serious limitation of the algorithm stems from the fact that the computation of the influence relation implies a pairwise comparison of all blog posts in the database, resulting in the complexity of $O(n^2)$, where $n$ is the number of blog posts stored in the database. Currently, we are working on an additional filter for automatic tagging of blog posts in order to limit the pairwise comparison of blog posts only to the blogs that share a tag in common. We are also considering the idea to merge blog ranking produced by the algorithm with external ranking services, such as Alexa or Technorati. The above directions can be perceived as our future work agenda in the area of blog mining.

## References

Agarwal N., Liu H., Tang L., Yu P.S. (2008), Identifying the influential bloggers in a community, [in:] *Proceedings of the WSDM'08, Palo Alto, USA, February 11-12*, Eds. M. Najork, A.Z. Broder, A. Chakrabarti, ACM, pp. 207-218.

Bachnik W., Szymczyk S., Leszczyński P., Podsiadlo R., Rymszewicz E., Kuryło L., Makowiec D., Bykowska B. (2005), Quantitative and sociological analysis of blog networks, *Acta Physica Polonica* B, Vol. 36, No. 10, pp. 2435-2446.

Backstrom L., Huttenlocher D.P., Kleinberg J.M., Lan X. (2006), Group formation in large social networks: Membership, growth, and evolution, [in:] *Proceedings of the ACM SIGKDD'06, Philadelphia, USA, August 20-23*, Eds. T. Eliassi-Rad, L.H. Ungar, M. Craven, D. Gunopulos, ACM, pp. 44-54.

Chaua M., Xu J. (2007), Mining communities and their relationships in blogs: A study of online hate groups, *International Journal of Human-Computer Studies*, Vol. 65, No. 1.

Chin A., Chignell M.H. (2006), A social hypertext model for finding community in blogs, [in:] *Proceedings of the ACM Hypertext'06, Odense, Denmark, August 22-25*, Eds. U. Kock Wiil, P.J. Nürnberg, J. Rubart, ACM,pp. 11-22.

Conrad J.G., Schilder F. (2007), Opinion mining in legal blogs, [in:] *Proceedings of the ICAIL'07, Stanford, USA, June 4-8*, ACM, pp. 231-236.

Dave K., Lawrence S., Pennock D. M. (2003), Mining the peanut gallery: Opinion extraction and semantic classification of product reviews, [in:] *Proceedings of the WWW'03, Budapest, Hungary, May 20-23*, ACM, pp. 519-528.

Facca F.M., Lanzi P.L. (2005), Mining interesting knowledge from weblogs: A survey, *Data & Knowledge Engineering*, Vol. 53, No. 3.

Fujimura K., Toda H., Inoue T., Hiroshima N., Kataoka R., Sugizaki M. (2006), Blogranger – a multifaceted blog search engine, [in:] *Proceedings of the 'WWW'06, Edinburgh, UK, May 23*, Eds. L. Carr, D. De Roure, A. Iyengar, C.A. Goble, M. Dahlin, ACM.

Gruhl D., Guha R., Liben-Nowell D., Tomkins A. (2004), Information diffusion through blogspace, [in:] *Proceedings of the WWW'04, New York, USA, May 20-29*, Eds. S.I. Feldman, M. Uretsky, M. Najork, C.E. Wills, ACM, pp. 491-501.

Java A. (2007), A framework for modeling influence, opinions and structure in social media, [in:] *Proceedings of the AAAI AI'07, Vancouver, Canada, July 22-26*, AAAI Press, pp. 1933-1934.

Java A., Kolari P., Finin T., Mayfield J., Joshi A., Martineau J. (2007), Blogvox: Separating blog wheat from blog chaff, [in:] *Proceedings of the IJCAI'07, Hyderabad, India, January 6-12*, Ed. M.M. Veloso.

Java A., Kolari P., Finin T., Oates T. (2006), *Modeling the spread of influence on the blogosphere. Technical report*, University of Maryland, Baltimore County.

Joshi M., Belsare N. (2006), Blogharvest: Blog mining and search framework, [in:] *Proceedings of the COMAD'06, Delhi, India, December 14-16*.

Kempe D., Kleinberg J. M., Tardos E. (2003), Maximizing the spread of influence through a social network, [in:] *Proceedings of the ACM SIGKDD'03, Washington, USA, August 24-27*, Eds. L. Getoor, T.E. Senator, P. Domingos, C. Faloutsos, ACM, pp. 137-146.

Kleinberg J.M. (2007), Challenges in mining social network data: Processes, privacy, and paradoxes, [in:] *Proceedings of the ACM SIGKDD'07, San Jose, USA, August 12-15*, Eds. P. Berkhin, R. Caruana, X. Wu, ACM, pp. 4-5.

Kossinets G., Watts D.J. (2006), Empirical analysis of an evolving social network, *Science*, Vol. 311, No. 5757, AAAS, pp 88-90.

Kumar R., Novak J., Tomkins A. (2006), Structure and evolution of online social networks, [in:] *Proceedings of the ACM SIGKDD'06, Philadelphia, USA, August 20-23*, Eds. T. Eliassi-Rad, L.H. Ungar, M. Craven, D. Gunopulos, ACM, pp. 611-617.

Leskovec J., Kleinberg J.M., Faloutsos C. (2005), Graphs over time: Densification laws, shrinking diameters and possible explanations, [in:] *Proceedings of the ACM SIGKDD'05, Chicago, USA, August 21-24*, Eds. R. Grossman, R.J. Bayardo, K.P. Bennett, ACM, pp. 177-187.

Li X., Liu B. (2005), Mining community structure of named entities from free text, [in:] *Proceedings of the ACM CIKM'05, Bremen, Germany, October 31-November 5*, Eds. O. Herzog, H-J. Schek, N. Fuhr, A. Chowdhury, W. Teiken, ACM, pp. 275-276.

Mishne G., De Rijke M. (2006), A study of blog search, *Advances in Information Retrieval*, *Lecture Notes in Computer Science*, Vol. 3936, Springer Berlin–Heidelberg, pp. 289-301.

Nanno T., Fujiki T., Suzuki Y., Okumura M. (2004), Automatically collecting, monitoring, and mining Japanese weblogs, [in:] *Proceedings of the WWW Alt.'04, New York, USA, May 17-20*, Eds. S. Feldman, M. Uretsky, ACM, pp. 320-321.

Salton G., Wong A., Yang C.S. (1975), A vector space model for automatic indexing, *Communications of ACM*, Vol. 18, No. 11.

Salvetti F., Nicolov N. (2006), Weblog classification for fast splog filtering: A url language model segmentation approach, [in:] *Proceedings of the HLT-NAACL'06, New York, USA, June 4-9*, R.C. Moore, Eds. J.A. Bilmes, J. Chu-Carroll, M. Sanderson, The Association for Computational Linguistics, pp. 137-140.

Suhara Y., Toda H., Sakurai A. (2008), Extracting related named entities from blogosphere for event mining, [in:] *Proceedings of the ICUIMC'08, Suwon, Korea, January 31-February 1*, Eds. W. Kim, H-Jin Choi, ACM, pp. 225-229.

Yang K., Yu N., Valerio A., Zhang H., Ke W. (2007), Fusion approach to finding opinions in blogosphere, [in:] *Proceedings of the ICWSM'07, Boulder, USA, March 25-28*.

## RELACJA WPŁYWU W BLOGOSFERZE

**Streszczenie:** artykuł jest poświęcony problemom rankingu i eksploracji blogów. Prezentujemy wprowadzenie do blogosfery i przegląd wcześniejszych prac naukowych. Przedstawiamy relację wpływu występującą między blogami i wykorzystujemy tę relację do odkrycia podzbioru najbardziej wpływowych blogów w kontekście określonym przez zapytanie użytkownika. Metoda zaprezentowana w artykule wykorzystuje dualną reprezentację blogów, która zakłada podział blogu na dwa rozłączne wektory: wektor tematów i wektor cech. Wektor tematów jest wykorzystywany do tematycznego dopasowania blogu do zapytania użytkownika. Wektor cech stanowi rejestr opisywanych pojęć i służy do skonstruowania wynikowego zbioru pojęć. W artykule prezentujemy wyniki wstępnych eksperymentów, które wskazują, że dualna reprezentacja blogów w połączeniu z relacją wpływu stanowi atrakcyjną alternatywę dla aktualnie wykorzystywanych algorytmów rankingu blogów.