# Blog Distillation via Sentiment-Sensitive Link Analysis

Giacomo Berardi, Andrea Esuli, Fabrizio Sebastiani, and Fabrizio Silvestri

Istituto di Scienza e Tecnologie dell'Informazione,
Consiglio Nazionale delle Ricerche, 56124 Pisa, Italy
firstname.lastname@isti.cnr.it

**Abstract.** In this paper we approach blog distillation by adding a link analysis phase to the standard retrieval-by-topicality phase, where we also we check whether a given hyperlink is a citation with a positive or a negative nature. This allows us to test the hypothesis that distinguishing approval from disapproval brings about benefits in blog distillation.

## 1 Introduction

Blog distillation is a subtask of blog search. It is defined as the task of ranking in decreasing order of relevance the set of blogs in which the topic expressed by the query $q$ is a recurring and principal topic of interest. Blog distillation has been intensively investigated within the TREC Blog Track [4], where participants have experimented with various combinations of (i) methods for retrieval by topicality and (ii) sentiment analysis methods. Retrieval by topicality is needed since topicality is a key aspect in blog distillation, while sentiment analysis is needed since blogs tend to contain strongly opinionated content.

We test a method for blog distillation in which, on top of a standard system for retrieval by topicality, we add a link analysis phase meant to account for the reputation, or popularity, of the blog. However, due to the highly opinionated nature of blog contents, many hyperlinks express a *rebuttal*, and not an approval, of the hyperlinked post on the part of the hyperlinking post. In this work we test the hypothesis that distinguishing hyperlinks expressing approval from ones expressing rebuttal may benefit blog distillation. We thus define a *sentiment-sensitive link analysis* method, i.e., a random-walk method on which the two types of hyperlinks have a different impact. We detect the sentimental polarity of a given hyperlink (i.e., detect if it conveys a positive or a negative endorsement) by performing sentiment analysis on a text window around the hyperlink.

## 2 Sentiment-Sensitive Link Analysis for Blog Ranking

In the following we discuss the model we have adopted to compute the sentiment-sensitive, link-analysis-based ranking of blogs and blog posts. We rely on a graph-based model in which nodes represent either blogs or blog posts, and the weights attached to nodes represent their importance.

G. Bouma et al. (Eds.): NLDB 2012, LNCS 7337, pp. 228–233, 2012.

**A Graph-Based Model of the Blogosphere.** Let $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ be a set of blog posts, partitioned into a set of blogs $\mathcal{B} = \{b_1, b_2, \ldots, b_m\}$. Let $G_P = (\mathcal{P}, E_P)$ be a graph, where the set of nodes $\mathcal{P}$ is as above and $E_P$ is a set of edges corresponding to hyperlinks between posts, i.e., edge $e_{xy}$ from post $p_x \in \mathcal{P}$ to post $p_y \in \mathcal{P}$ denotes the presence of at least one hyperlink from $p_x$ to $p_y$. Similarly, let $G_B = (\mathcal{B}, E_B)$ be a graph, where the set of nodes $\mathcal{B}$ is as above and $E_B$ is a set of edges corresponding to hyperlinks between blogs, i.e., edge $e_{ij}$ from blog $b_i \in \mathcal{B}$ to blog $b_j \in \mathcal{B}$ denotes the presence of at least one hyperlink from a post $p_x \in b_i$ to the homepage of blog $b_j$. Let $w_P : E_P \rightarrow \mathbb{R}$ and $w_B : E_B \rightarrow \mathbb{R}$ be two weighting functions (described in detail in the next section), where $\mathbb{R}$ is the set of the reals. Informally, the weight assigned to an edge models the importance the corresponding hyperlink confers onto the hyperlinked post (for $E_P$) or blog (for $E_B$).

Let $q$ be a query. Our blog distillation method comprises a first step consisting in running a standard (i.e., text-based) retrieval engine on $\mathcal{P}$, yielding a ranked list of the $k$ top-scoring posts for $q$. Let $L = l_1, l_2, \ldots, l_k$ be this ranked list (with $l_1$ the top-scoring element), and let $s_1, s_2, \ldots, s_k$ be the scores returned for the posts in $L$ by the retrieval engine.

**Weighting the Hyperlinks.** We define the weighting functions $w_P$ and $w_B$ on the basis of a sentiment-based analysis, whose aim is to determine if an edge $e_{xy}$ denotes a *positive* or a *negative* attitude of post $p_x$ towards post $p_y$ (for $w_P$) or of post $p_x \in b_i$ towards blog $b_j$ (for $w_B$). A positive (resp., negative) value of $w_P(e_{xy})$ will indicate a positive (resp., negative) attitude of the linking document $p_x$ toward the linked document $p_y$, and the absolute value of $w_P(e_{xy})$ will indicate the intensity of that attitude. The same goes for $w_B$.

For determining $w_P(e_{xy})$, all the hyperlinks from $p_x$ to $p_y$ are taken into account; similarly, for determining $w_B(e_{ij})$, all the hyperlinks from any post $p_x \in b_i$ to the homepage of blog $b_j$ are taken into account. For determining the impact of a hyperlink on the weighting function, the anchor text and the sentence in which the anchor text is embedded are analysed. This analysis begins with POS-tagging the sentence in order to identify *candidate sentiment chunks*, i.e., all the sequences of words matching the `(RB|JJ)+` and `NN+` patterns.

We assign a sentiment score to each term in the chunk by using SentiWordNet (SWN) [2] as the source of sentiment scores. From SentiWordNet we have created a word-level dictionary (that we call SentiWordNet$_w$) in which each POS-tagged word $w$ (rather than each word sense $s$, as in full-fledged SentiWordNet) is associated to a score $\sigma(w)$ that indicates its sentimental valence, averaged across its word senses. We have heuristically obtained this score by computing a weighted average $\sigma(w) = \sum_i \frac{1}{i}(Pos(s_i(w)) - Neg(s_i(w)))$ of the differences between the positivity and negativity scores assigned to the various senses $s_1(w), s_2(w), ...$ of $w$. In this weighted average the weight is the inverse of the sense number, thus lending more prominence to the most frequent senses.

All candidate sentiment chunks only composed by terms that have been assigned a sentiment score equal to 0 (i.e., sentiment-neutral words) are discarded from consideration. Each of the remaining *sentiment chunks*, together with a

portion of text preceding it, is then checked for the presence of *sentiment modifiers*, i.e., negators (e.g., "no", "not") or intensifiers / downtoners (e.g., "very", "strongly", "barely", "hardly"). As the resource for sentiment modifiers we have used the appraisal lexicon defined in [1]. When a modifier is found, the score of the word that follows it is modified accordingly (e.g., "very good" is assigned a doubly positive score than "good"). We then assign a sentiment score to a chunk by simply summing the sentiment scores of all the words in a chunk, taking into account the modifiers as described above.

In order to determine a sentiment score for the hyperlink, we compute a weighted sum of the sentiment scores of all the chunks that appear in the sentence containing the anchor text. We use weights that are a decreasing function of the distance of the chunk from the anchor text, according to the assumption that that the closer a chunk is to the hyperlink, the more it is related to it. This distance is itself computed as a weighted sum, where each token between the anchor text and the chunk has its own weight depending on its type; for instance, mood-changing particles such as "instead" are assigned a higher weight.

Finally, we compute $w_P(e_{xy})$ as the mean of the sentiment scores assigned to the links from $p_x$ to $p_y$; if both positive and negative links are present, they thus compensate each other. Similarly, we compute $w_B(e_{ij})$ as the mean of the values associated to hyperlinks from posts in $b_i$ to the homepage of blog $b_j$.

Using the $w_P$ function we then split $G_P$ into two graphs $G_P^+ = \left(\mathcal{P}, E_P^+\right)$ and $G_P^- = \left(\mathcal{P}, E_P^-\right)$, where $E_P^+$ and $E_P^-$ are the sets of edges $e_{xy}$ such that $w_P(e_{xy}) \geq 0$ and $w_P(e_{xy}) < 0$, respectively. Analogously, we split $G_B$ into $G_B^+ = \left(\mathcal{B}, E_B^+\right)$ and $G_B^- = \left(\mathcal{B}, E_B^-\right)$, where $E_B^+$ and $E_B^-$ are the sets of edges $e_{ij}$ such that $w_B(e_{ij}) \geq 0$ and $w_B(e_{ij}) < 0$, respectively.

**Ranking the Nodes.** We use the graphs $G_P^+$, $G_P^-$, $G_B^+$, $G_B^-$ in order to compute the ranking of posts and blogs based on sentiment-sensitive link analysis. We use an algorithm known as *random walk with restart* (RWR), also known as *personalized* (or *topic-sensitive*) *random walk* [3], which differs from more standard random walk algorithms such as PageRank for the fact that the $\mathbf{v_P}$ and $\mathbf{v_B}$ vectors (see below) are not uniform. The values in the latter vectors are sometimes referred to as the *restart probabilities*. Two RWR computations are run on $G_P^+$ and $G_B^+$, respectively, yielding $\mathbf{r_P}$ and $\mathbf{r_B}$ (i.e., the vectors of scores for posts and blogs) as the principal eigenvectors of the matrices $\mathbf{P} = (1-a) \cdot \mathbf{A_P^+} + \frac{a}{k} \cdot \mathbf{v_P}$ and $\mathbf{B} = (1-a) \cdot \mathbf{A_B^+} + \frac{a}{k'} \cdot \mathbf{v_B}$, where $\mathbf{A_P^+}$ (resp., $\mathbf{A_B^+}$) is the adjacency matrix associated with graph $G_P^+$ (resp., $G_B^+$), and $a$ is the damping factor (i.e., the factor that determines how much backlinks influence random walks). Vector $\mathbf{v_P}$ is the preference vector for blog posts, i.e., a vector whose entries corresponding to the $k$ pages in $L$ are set to 1 and whose other entries are set to 0. Vector $\mathbf{v_B}$ is obtained in a slightly different way. We first group together posts belonging to the same blog and build a vector $\overline{\mathbf{v}}_\mathbf{B}$ whose entries count the number of retrieved posts belonging to the blog corresponding to the entry. Vector $\overline{\mathbf{v}}_\mathbf{B}$ is then normalized into vector $\mathbf{v_B}$ and, in this case, $k'$ is the number of entries in $\mathbf{v_B}$ greater than zero. In order to find the principal eigenvectors of $\mathbf{P}$ and

$\mathbf{B}$ we solve the eigenproblems $\mathbf{r}_P^+ = \mathbf{P} \cdot \mathbf{r}_P^+ = (1-a) \cdot \mathbf{A}_\mathbf{P}^+ \cdot \mathbf{r}_P + \frac{a}{k} \cdot \mathbf{v}_\mathbf{P}$ and $\mathbf{r}_B^+ = \mathbf{B} \cdot \mathbf{r}_B^+ = (1-a) \cdot \mathbf{A}_\mathbf{B}^+ \cdot \mathbf{r}_B + \frac{a}{k'} \cdot \mathbf{v}_\mathbf{B}$. We calculate the vectors of negative scores $\mathbf{r}_P^- = \mathbf{A}_\mathbf{P}^- \cdot \mathbf{r}_P^+$ and $\mathbf{r}_B^- = \mathbf{A}_\mathbf{B}^- \cdot \mathbf{r}_B^+$, and we normalize them so that the sum of their negative components is -1. $\mathbf{A}_\mathbf{P}^-$ and $\mathbf{A}_\mathbf{B}^-$ contain the negative values associated with the edge weights of $G_P^-$ and $G_B^-$. Finally, the scoring vectors $\mathbf{r}_P$ and $\mathbf{r}_B$ that result from taking into account both positive and negative links are given by $\mathbf{r}_P = (1-\theta) \cdot \mathbf{r}_P^+ + \theta \cdot \mathbf{r}_P^-$ and $\mathbf{r}_B = (1-\theta) \cdot \mathbf{r}_B^+ + \theta \cdot \mathbf{r}_B^-$, where $\theta$ is a coefficient for tuning the relative impact of positive and negative links on the overall score of a post (or blog). In our tests we have set $\theta = \frac{1}{2}$.

**Scoring Blogs against a Query.** We are now in a position to describe our blog distillation method. Let $q$ be a query whose aim is to rank the blogs in descending order of relevance to the query. Our method consists of three steps.

As described in Section 2, in the 1st step a standard (i.e., text-based) retrieval engine is run on $\mathcal{P}$, yielding a ranked list of the $k$ top-scoring posts for $q$. Let $L = l_1, l_2, \ldots, l_k$ be this ranked list (with $l_1$ the top-scoring element), and let $s_1, s_2, \ldots, s_k$ be the corresponding scores returned by the retrieval engine.

The 2nd step consists of combining the retrieval scores $s_x$ with the link-based scores. We use the combination $w_x = (1-\alpha) \cdot s_x + \alpha \cdot \mathbf{r}_{Px}$ for all $x \in [1, k]$, where $\alpha$ is a coefficient used to balance the weights of link-based and text-based scores and $\mathbf{r}_{Px}$ is the $x$-th component of the vector $\mathbf{r}_P$ as computed in Section 2.

The 3rd step consists of merging the scores computed for each post according to the blog the post itself comes from. Obviously the choice of the merging function is an important issue. We simply weight each blog $b_i$ using the average of the scores $w_x$ for each post $p_x \in b_i$ plus the static score $\mathbf{r}_{Bi}$ of blog $b_i$ smoothed using the actual number of posts retrieved for blog $b_i$. More formally, we use the same $\alpha$ coefficient as above to balance the weight of average post score and the link-based score of blog $b_i$; to score blog $b_i$ we use the equation $\omega_i = \frac{(1-\alpha)}{|b_i|} \cdot \sum_{x:p_x \in b_i} w_x + \alpha \cdot \mathbf{r}_{Bi} \cdot \frac{|L \cap b_i|}{|b_i|}$ where by $|L \cap b_i|$ we denote the number of posts of blog $b_i$ retrieved as top-$k$ posts in the list $L$. Eventually, our blog retrieval system returns blogs sorted by the $\omega_i$ scores.

## 3 Experiments

We have tested our method on the Blogs08 collection used in the 2009 and 2010 editions of the TREC Blog Track [4]. Blogs08 consists of a crawl of 1,303,520 blogs, for a total of 28,488,766 blog posts, each identified by a unique URL. We have followed the protocol of the 2009 Blog Track, using the 50 queries of 2009.

We have processed all the hyperlinks via our hyperlink polarity scoring method (see Section 2). The processing of hyperlinks relative to the graph of blog posts resulted in the identification of 8,947,325 neutral links (i.e., polarity weight equal to zero), 1,906,182 positive links, and 1,780,281 negative links.

As the system for performing the first step of our method we have used Terrier [5]. With Terrier we have built a baseline for the comparison of our results, based on ranking the documents with respect to each query via the well-known

**Table 1.** Ranking effectiveness on the Blogs08 collection. "Base" indicates the baseline; "$\alpha = r$" indicates our method, where $r$ is the value of $\alpha$ which has returned the best result; "Mean" indicates the average performance of our method across the four tested values of $\alpha$. **Boldface** indicates the best result.

| $k$ | 1,000 | | | 2,000 | | | 3,000 | | | 4,000 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Base | $\alpha = 0.85$ | Mean | Base | $\alpha = 0.75$ | Mean | Base | $\alpha = 0.80$ | Mean | Base | $\alpha = 0.65$ | Mean |
| MAP | 0.1775 | 0.1802 | 0.1801 | 0.1914 | 0.1943 | 0.1942 | 0.1958 | **0.1986** | 0.1983 | 0.1949 | 0.1977 | 0.1976 |
| P@10 | 0.2878 | **0.2898** | 0.2880 | 0.2796 | 0.2837 | 0.2819 | 0.2653 | 0.2735 | 0.2719 | 0.2592 | 0.2673 | 0.2650 |
| bPref | 0.2039 | 0.2056 | 0.2050 | 0.2203 | 0.2222 | 0.2220 | 0.2226 | **0.2247** | 0.2242 | 0.2210 | 0.2230 | 0.2224 |

BM25 weighting method. These rankings have been used also as the input to the random-walk-based reranking phase of our method. For each query we retrieve the first hundred posts, which are set as the nodes with non-null restart probabilities given as input to the RWR method. As the damping factor $a$ (see Section 2) we have used the value 0.85, a fairly typical value in these methods [3]. The value used for stopping the iterative process is set to $\epsilon = 10^{-9}$.

We have evaluated the results of our experiments via the well-known *mean average precision* (MAP), *binary preference* (bPref), and *precision at 10* (P@10).

We have tested various values for parameters $\alpha$ (see Section 2) and $k$ (the number of posts retrieved in the 1st step of our method). For $\alpha$ we have tested the values 0.65, 0.75, 0.80, 0.85; preliminary experiments with values outside this range had shown clear deteriorations in effectiveness. For $k$ we have tested the values 1000, 2000, 3000, 4000; here too, preliminary experiments with values outside this range had shown clear deteriorations.

The different values for the $\alpha$ parameter do not yield substantial differences in performance. This can be gleaned from Table 1 by looking at the differences among best and mean values obtained by our method (2nd and 3rd column of each block), which are always very small.

The results do show an improvement over the retrieval-by-topicality baseline, but this improvement is very small. A closer inspection of the results reveals that this is due to the sparsity of the graphs resulting from the collection of posts retrieved in the first step: these posts are often isolated nodes in the posts graph, which means that the random-walk step only affects a small subset of the results. Increasing the value of the $k$ parameter determines an increase of the improvement over the baseline, since more posts are affected by the random walk scores, but the relative improvement is still small anyway. We have seen that increasing $k$ beyond the value of 4000, instead, introduces a higher number of irrelevant posts in the results, which decreases the magnitude of the improvement.

We have also tried different values for $\theta$ (see Section 2), but this has not brought about any substantial improvement. The main reason is that, due to the above mentioned sparsity of the posts graph, the impact of the link analysis phase on the final ranking is low anyway, regardless of how this link analysis balances the contribution of the positive and negative links.

In order to obtain a further confirmation of the fact that using link analysis does not impact substantively on the final ranking, we have run an experiment in which the RWR random-walk method has been replaced by standard PageRank.

**Table 2.** MAP evaluations for our random walk with restart (RWR) and PageRank (PR), with $k = 3,000$. **Boldface** indicates best results for each weight.

| $\alpha$ | 0.65 | 0.75 | 0.80 | 0.85 |
|---|---|---|---|---|
| RWR | **0.1981** | **0.1983** | **0.1986** | 0.1983 |
| PR | 0.1978 | **0.1983** | 0.1985 | **0.1988** |

Here, $\mathbf{r_P}$ and $\mathbf{r_B}$ are computed similarly to our method but for the fact that the entire $G_P$ and $G_B$ graphs are used, i.e., without differentiating positive and negative links. Edges are weighted with uniform probability of transition equal to $1/outdegree(node)$. The differences in the final results are negligible (see Table 2). The very slight improvement brought about by PageRank over our method is probably due to the fact that the two graphs used by PageRank have a larger proportion of non-isolated nodes than each of the graphs on which our method operates. Our algorithm, however, is faster than PageRank, because it works on the reduced transition matrices given by $G_B^+$ and $G_P^+$.

## 4 Conclusions

We can conclude that the hypothesis according to which it makes sense to distinguish positive from negative endorsements in blog analysis has neither been confirmed nor disconfirmed. To see this, note that the literature on blog search has unequivocally shown that the best results are obtained when sentiment analysis is performed not on the entire blogosphere, but on the subset of blogs which have been top-ranked by a standard retrieval-by-topicality engine [4]. The essential conclusion that can be drawn from our results is that the blogs (and their posts) retrieved in the retrieval-by-topicality phase contain too few hyperlinks / endorsements, no matter whether positive or negative, for a link analysis phase to have a substantial impact on retrieval.

## References

1. Argamon, S., Bloom, K., Esuli, A., Sebastiani, F.: Automatically Determining Attitude Type and Force for Sentiment Analysis. In: Vetulani, Z., Uszkoreit, H. (eds.) LTC 2007. LNCS, vol. 5603, pp. 218–231. Springer, Heidelberg (2009)
2. Baccianella, S., Esuli, A., Sebastiani, F.: SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In: Proceedings of LREC 2010 (2010)
3. Haveliwala, T.H.: Topic-sensitive PageRank: A context-sensitive ranking algorithm for Web search. IEEE TKDE 15(4), 784–796 (2003)
4. Macdonald, C., Santos, R.L., Ounis, I., Soboroff, I.: Blog Track research at TREC. SIGIR Forum 44(1), 58–75 (2010)
5. Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., Lioma, C.: Terrier: A high performance and scalable information retrieval platform. In: Proceedings of the SIGIR 2006 Workshop on Open Source Information Retrieval, Seattle, US (2006)