Linking Online News and Social Media

Manos Tsagkias
ISLA, University of Amsterdam
e.tsagkias@uva.nl

Maarten de Rijke
ISLA, University of Amsterdam
derijke@uva.nl

Wouter Weerkamp
ISLA, University of Amsterdam
w.weerkamp@uva.nl

ABSTRACT

Much of what is discussed in social media is inspired by events in the news and, vice versa, social media provide us with a handle on the impact of news events. We address the following linking task: given a news article, find social media utterances that implicitly reference it. We follow a three-step approach: we derive multiple query models from a given source news article, which are then used to retrieve utterances from a target social media index, resulting in multiple ranked lists that we then merge using data fusion techniques. Query models are created by exploiting the structure of the source article and by using explicitly linked social media utterances that discuss the source article. To combat query drift resulting from the large volume of text, either in the source news article itself or in social media utterances explicitly linked to it, we introduce a graph-based method for selecting discriminative terms.

For our experimental evaluation, we use data from Twitter, Digg, Delicious, the New York Times Community, Wikipedia, and the blogosphere to generate query models. We show that different query models, based on different data sources, provide complementary information and manage to retrieve different social media utterances from our target index. As a consequence, data fusion methods manage to significantly boost retrieval performance over individual approaches. Our graph-based term selection method is shown to help improve both effectiveness and efficiency.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.3 Information Search and Retrieval

General Terms

Algorithms, Experimentation

Keywords

Linking, online news, social media, user generated content

1. INTRODUCTION

A symbiotic relation has emerged between online news and social media such as blogs, micro-blogs, social bookmarking sites, news comments and Wikipedia. Much of what is discussed in social media is inspired by the news (e.g., 85% of Twitter statuses are news-related [22]) and, vice versa, social media provide us with a handle on the impact of news events [5, 25, 27, 46]. Understanding the relationship between news and social media has become an area of significant research interest. A key ingredient is to discover and establish links between individual news articles and the social media that discuss them.

Social media utterances (such as blog posts, tweets, diggs, etc) may be linked *explicitly* or *implicitly* to a news article. In explicitly linked utterances there is a hyperlink pointing to the article; automatic discovery of such utterances is trivial. In implicitly linked utterances, however, there is no hyperlink to the source article—the utterance is not merely about the same topic as the source news article but it directly discusses the article's content. Consider an utterance discussing the FIFA World Cup 2010 final, expressing the utterance writer's opinion on the match. This is not considered an implicitly linked utterance; would this utterance criticize the match report given in a news article, however, then it would be an implicitly linked utterance for this news article.

The task on which we focus in this paper is *discovering implicitly linked social media utterances*: For a given news article we discover social media utterances that discuss the article. Both the notion of relevance (detailed above) and the fact that, to address the task, one needs to cross from edited content to the unedited and strongly subjective language usage of user generated content, make the task challenging. To quantify the potential "vocabulary gap" [8, 9, 11] we conducted an exploratory experiment. We considered a set of news articles plus a range of social media platforms; for each news article we computed the (symmetric) Kullback-Leibler (KL) divergence between the article and the social media utterances explicitly linked to it (grouped by platform) as a way of approximating the difference in vocabularies; see Fig. 1 for a visualization. We observe (varying levels of) difference in vocabulary between news and social media. The vocabularies of blog posts, Digg and Twitter seem relatively close to that of the news articles—anecdotal evidence suggests that this is due to these sources copying parts of the original news article. Moreover, the social media platforms show varying degrees of difference between their vocabularies.

When attempting to link social media utterances to a given news article, the main question is: how do we represent the article as a query? Typically, the article itself has a fielded structure (title, lead, body, headers, etc) that can be exploited [2, 8, 23]. Which of these is helpful in identifying implicitly linked social media utterances? Alternatively, one can try to identify a selection of "representative" terms from the article [2, 5, 17]. Given the noisy or unedited character of many social media utterances, the selection procedure needs to be very robust. There is a third alternative, based on the

WSDM'11, February 9–12, 2011, Hong Kong, China.
Copyright 2011 ACM 978-1-4503-0493-1/11/02 ...$10.00.

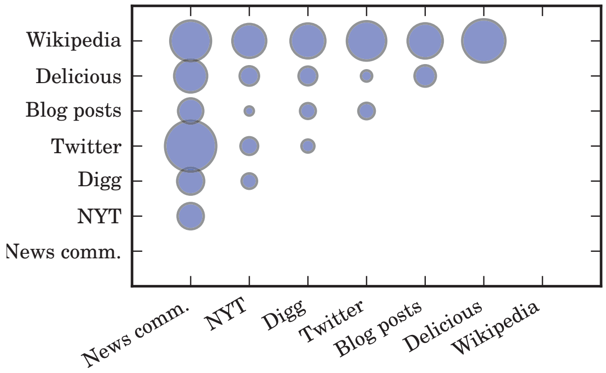

Figure 1: Avg. symmetric KL-divergence between New York Times articles and explicitly linked social media utterances from Digg, Twitter, blog posts, New York Times comments, Delicious, Wikipedia. Larger circles indicate a higher degree of divergence and hence a bigger difference in vocabulary.

observation that there may be many social media utterances that *explicitly* reference a given news article. For a sample of news articles (described in §5.3), Table 6 displays the number of articles that are explicitly referenced by the six social media platforms considered above. What if we used representations of a news article generated from social media utterances that explicitly link to it?

Given these options, we approach the task of discovering implicitly linked social media utterances for a news article as a data fusion problem. We generate multiple query models for an article, based on three strategies: (i) its internal document structure, (ii) explicitly linked social media utterances, and (iii) term selection strategies. This yields ranked lists per strategy and these ranked lists are then merged using data-fusion methods. The research questions we aim to answer are the following:

1. Does the internal document structure of a news article help to retrieve implicitly linked social media utterances?
2. Do query models derived from social media models outperform models based on internal document structure?
3. Is implicilt link discovery effectiveness affected by using reduced query models that only use a limited selection of words?
4. How can ranked lists from individual strategies be fused to improve performance?
5. Does it help effectiveness when making the fusion strategy dependent on the news article for which we are seeking implicitly linked utterances?

When talking about effectiveness of a method, we consider the performance of the method in terms of recall or precision-oriented metrics. Efficiency on the other hand deals with a method's performance in terms of speed.

Our main contributions are the following: (a) we introduce the task of discovering social media utterances implicitly linked to a news article; (b) we offer a comparison of query models derived from (i) the document itself and (ii) auxiliary social media platforms in terms of the effectiveness of finding implicitly linked utterances; (c) we propose a robust graph-based term selection method, apply it to document and social media models, and compare the effectiveness and efficiency of these reduced models to the original models; and (d) we compare three types of late data fusion methods for combining ranked lists: (i) without training, (ii) query independent training, and (iii) query dependent training.

The rest of the paper is organized as follows: We report on related work in §2, we present our approach in §3, our models are presented in §4, our experimental setup is described in §5, we report on results and discuss our findings in §6, and conclude in §7.

2. RELATED WORK

News and social media. Much of what is discussed in social media is inspired by the news [4, 18, 22, 30, 39, 40, 45]. Even search in social media is to an important degree influenced by news events [36]. As a consequence, the text analysis and retrieval communities have begun to examine the relationship between the two—news and social media—from a range of angles. Recent emerging interest concerns work on predicting the response to a news article in social media [20, 43, 46].

We consider a task that is different from the ones mentioned so far: discovering implicitly linked social media utterances that discuss a news article. Link identification has been used to track short information cascades through the blogosphere [1, 15, 21, 24]. Of particular relevance to us, though, is the work by Ikeda et al. [17] who use similarity between term vectors that represent news articles and blog posts to decide on the existence of links between the two. On top of that, Takama et al. [44] use the difference between publication times of news articles and blog posts to decide on the existence of a link. Gamon et al. [12] use a graph-based approach to create context for news articles out of blog posts. We are interested in discovering utterances that implicitly link to a specific news article and not to the news event(s) that the article is about.

Blog post retrieval. Viewed abstractly, the task we consider—discovering social media utterances that are (implicitly) linked to a given news article—is similar to the (topical) blog post finding that has been examined at the TREC Blog track between 2006 and 2009 [38]. There are important differences, though, that motivate approaches to the task of discovering social media utterances that are technically and conceptually different from existing approaches to the blog post finding task. For a start, the information need, and therefore the notion of relevance is different: instead of posts that discuss a topic, we seek to identify utterances that reference a specific article—not a different article that is possibly about the same topic. Among other things, this leads to a dramatically different technical setting, with elaborate information needs (the source article) as opposed to the typical two or three word queries or two or three line narratives. Moreover, relevant utterances are necessarily published after the source news article and tend to be published reasonably shortly after the source article [25]. Conceptually, we are crossing genres, from edited news (on the query side) to user generated content (on the result side).

One of the themes that has emerged around blog (post) retrieval is the use of non-content features. Timeliness is one such feature that is particularly relevant for our setting. Another one concerns quality indicators; we use the credibility indicators in [47].

Combining multiple representations. From work on the topical blog post retrieval task mentioned above, we borrow the insight that social media retrieval benefits from elaborate query modeling [3, 48]. One pertinent type of query representation considered in the literature exploits query structure—we use the structure of the source article, "our query," to obtain different ways of representing the source news article for which we are seeking to identify linked social media utterances. Another pertinent type of query representation known concerns the use of "external corpora," where terms are sampled from documents that are not in the target collection from which items need to be retrieved; a prominent example of such an external corpus in the setting of blog retrieval is Wikipedia.

The idea of using multiple representations of a query or its underlying information need has a long history; Belkin et al. [7] summarize work on the theme that builds off the early TREC collections. More broadly, combinations of approaches—either at the level of queries, sources, or result rankings—have been met with different degrees of success. Snoek et al. [42] identify two types of combination approaches depending on whether the combination occurs at the query level (*early fusion*) or at the result level (*late fusion*). In the setting of blog post retrieval, Weerkamp et al. [48] show that the use of multiple query representations (in the form of complex query models) helps improve blog post retrieval effectiveness. Interestingly, Beitzel et al. [6] find that combinations of highly effective systems hurt performance as compared to the performance of the individual approaches. McCabe et al. [28] find that combinations of a poorly performing approach with a good system, using weights where the good system is weighted highly, leads to performance gains over the good system. We apply standard (late) data fusion approaches [41], re-examine insights on data fusion from the literature and shed new light on the effectiveness of combinations in the context of our finding linked utterances task; see §4.3, 5.4.

3. APPROACH

Starting from a *source* news article, we need to identify, in a *target* index, utterances that reference the source article. We view this task as a *data fusion problem*: starting from the source article, we derive and apply query models to generate multiple queries, which are then used to retrieve utterances from the target index, resulting in multiple ranked lists that we then merge into a single result list; see Fig. 2. Let us motivate these steps.

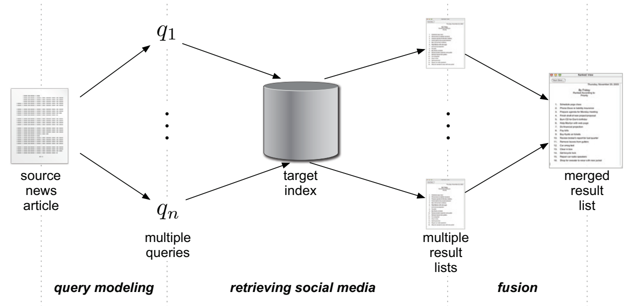

Figure 2: Approach to finding linked social media utterances.

Most of our attention in this paper will be devoted to the *query modeling* step. Importantly, in the process of identifying social media utterances that reference a given source news article, we are crossing genres, from news to social media. When crossing genres, the vocabulary gap between source article ("the query") and target utterances ("the documents") is wider than within genres, especially when one of the genres involved is a social media genre [48]. To bridge the gap, we follow multiple alternative routes: starting from the source article, we consider multiple query modeling strategies, i.e., ways of arriving at a query to be fired against the target index. First, we consider different representations of the source news article itself. It is a semi-structured document that features elements such as title, lead and body. Derived representations such as the named entities mentioned in the article and quotations from interviews are also used to represent the article and generate queries. Second, to help generate queries that represent the source article we also use auxiliary social media. Intuitively, to bridge between the language usage of the source article and that of the utterances in the target index, we can exploit social media where the two types of language usage are, to some degree, mixed. E.g., a Digg story usually consists of the news article title and summary (edited content) and the user comments (unedited content), tweets mix the article title (edited) with the twitterer's opinion/comment (unedited).

The textual representations from which queries are derived may be quite long as compared to, for example, article titles. E.g., when focusing on the source news article, the entire title and body of the article can be used as a query [31]; such long queries, however, are costly to process and may introduce noise and cause topic drift. For this reason, we identify and extract terms that are discriminative and characteristic of language usage pertinent to the source article (or auxiliary social media) and use these to derive a query.

In the *retrieval* step, we submit queries representing the source news article to an index of social media utterances, and retrieve ranked lists for each of these queries.

In the *fusion* step, we use late data fusion methods [41, 42] to merge results lists produced by alternative query modeling methods. For the methods that support weighted fusion, we investigate two approaches for weight optimization: query independent and query dependent. In the former approach, the system learns weights for each query model from a training set so a given metric is maximized, and then these weights are used for fusing ranked lists in response to future articles. In the latter approach, weights are learned per source article ("query") so the given metric is maximized for an article-specific training ground truth.

4. METHODS

We describe the methods used in addressing the three steps identified in the approach outlined in §3: retrieving social media utterances, query modeling and data fusion.

4.1 Retrieval model

For the retrieval step, we use a language modeling approach. We compute the likelihood of generating a news article a from a language model estimated from an utterance u:

$$P_{lm}(a|u) = \prod_{w \in a} P(w|u)^{n(w,a)}, \tag{1}$$

where w is a query term in a, $n(w,a)$ the term frequency of w in a, and $P(w|u)$ the probability of w estimated using Dirichlet smoothing:

$$P(w|u) = \frac{n(w,u) + \mu P(w)}{|u| + \mu}, \tag{2}$$

where μ is the smoothing parameter, $|u|$ is the utterance length in words, and $P(w)$ is the probability of w in the collection.

We impose two constraints on our content-based model expressed in Eq. 1. The first is on the publication date of utterances potentially discussing the source news article. The second is on the "quality" of utterances being retrieved. Both are modeled in a probabilistic fashion so they can be incorporated in our content-based model.

As to the first constraint, we want to favor utterances published close to the source news article, mainly due to the volatility of the news; most social media utterances are generated around the news article publication date [25]. Given a date range t of length $|t|$ in days, an utterance can or cannot appear in t, therefore:

$$P_{date}(u|t) = \begin{cases} \frac{1}{n(u,t)}, & \text{if } u \text{ occurs in } t \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

where r is a time unit in t, $n(u,\cdot)$ is the number of utterances occurring in r or in t. We want to avoid discarding potentially relevant utterances that occur outside t, while still favoring those published in t. Therefore, we follow the language modeling paradigm and

Table 1: Models for individual credibility factors. $|u|$ is the utterance length in words, $n(X, u)$ is the number of X for utterance u, where $X = \{r, e, o, z, l\}$, and r is comments, o is pronouns, e is emoticons, z is capitalized words, and l is misspelled (or unknown) words.

$$\begin{aligned}
p_{comments}(u) &= \log(n(r,u)) \\
p_{emoticons}(u) &= 1 - n(e,u) \cdot |u|^{-1} \\
p_{post\_length}(u) &= \log(|u|) \\
p_{pronouns}(u) &= 1 - n(o,u) \cdot |u|^{-1} \\
p_{shouting}(u) &= 1 - n(a,u) \cdot |u|^{-1} \\
p_{spelling}(u) &= 1 - n(m,u) \cdot |u|^{-1}
\end{aligned}$$

derive an estimate for $P_{date}(u|t)$ based on Dirichlet smoothing:

$$\hat{P}_{date}(u|t) = \frac{1 + \mu P(u)}{n(u,t) + \mu}, \tag{4}$$

where μ is a smoothing parameter as in Eq. 2, and $P(u) = 1/n(u)$ is the a priori probability of an utterance to occur anytime and $n(u)$ is the total number of utterances in the collection.

Our second refinement of the retrieval model aims to account for adversarial social media utterances and for utterances that do not provide informative context for the article. We incorporate the credibility factors introduced in [47] as quality indicators. Specifically, we implement the following topic independent factors on the level of utterances: comments, emoticons, post length, pronouns, shouting, and spelling; Table 1 shows the model for each factor. All factors are given equal importance and are put together for the estimation of a global credibility prior probability $P_{cred}(u)$ for an utterance u:

$$P_{cred}(u) = \tfrac{1}{|F|} \textstyle\sum_{f \in F} p_f(u), \tag{5}$$

where $F = \{$*comments*, *emoticons*, *post_length*, *pronouns*, *shouting*, *spelling*$\}$.

Finally, the content-based, recency and credibility models are combined through their geometric mean in one score for ranking an utterance u given a source news article a and a date range t:

$$Score(u, a, t) = \sqrt[3]{P_{lm}(a|u) \cdot P_{date}(a|t) \cdot P_{cred}(u)} \tag{6}$$

4.2 Query modeling

Most of our methodological contributions concern query modeling: building a representation of news article a to be used for retrieval (see Eq. 1). We explore three families of query models, for which we consider (i) the source news article itself as a "generator" of query models, (ii) social media as such a generator, and (iii) "reducing" the sources from which we generate query models to single out target terms.

Exploiting the source article. Obviously, the source news article itself is an important source of information for creating query models that represent it. News articles typically feature a title, lead and body as structural elements. The title is indicative of the article's main topic and summarizes the article. The lead consists of a few sentences, gives insight on what will follow and includes the main actors of the article. The body is the main content. Following the probabilistic model in [31], the contents of these structural elements are mapped to queries in a straightforward manner: we use the entire contents of a selected element. For article title, we tested the effectiveness of using exact phrases for modeling, however, plain title content outperformed exact phrases and, hence, we use the plain title content to model title.

Table 2: Query models grouped by source; in addition, TH-Rank is applied to the following models: full, digg, and nytc.

Query Model	Source	Elements
Exploiting the source article		
title	Article	Title
lead	Article	Lead
body	Article	Body
metadata	Article	Author (byline), news agent
ne	Article	Named entities
quote	Article	Quotations
full	Article	Title and body
Exploiting social media		
digg	Digg	Title, description and comments
delicious	Delicious	Title, tags and their frequency
twitter	Topsy	Tweet
nytc	NYTC	Comment title and body
wikipedia	Wikipedia	Full article
blogposts	Blogs	Feed item in RSS

In addition to structural elements, we use two extra features as a source for query modeling: *named entities* and *quotations*. A great majority of articles refer to and discuss people, organizations, and locations. Given a news article a, we identify named entities in a by extracting sequences of capitalized words. Quotations are text passages from interviews with people inside the article and as such are likely to remain intact throughout information spread [25]. This characteristic renders them viable surrogates for an article. Starting from the two extra features, we arrive at query models by constructing exact phrases from the named entities and the quotations.

As a final step, we model article metadata, consisting of the byline that represents authorship, and the news agent. The byline consists of the first and last name of the author. For the news agent, we create a basic list of potential synonyms by examining how social media refer to the news agent. For example, New York Times is mapped with three synonyms: "New York Times," NYTimes, NYT. Content from the byline is combined with list of synonyms to produce the final query.

Table 2 (top) lists query models derived from the source article.

Exploiting social media. We consider a second family of query models, obtained from social media platforms that explicitly link to the source article. Examples include Digg stories (that have a URL), tweets that include a URL, etc. Consequently, it is possible to track a source news article to social media utterances via its URL. The idea is to create query models by aggregating content from a range of social media sources, for two reasons:

1. not all sources cover all news articles with the same intensity;
2. different social media may exhibit different language usage around the same source article.

By sampling content from multiple social media sources we increase the possibility of capturing the creativity in the language usage. We use a small number of social media platforms with frequent explicit links to news articles: Digg, Delicious, Twitter and NYT Community (NYTC); see §5 for details. We also use content from blog posts that hyperlink to a source article and Wikipedia articles relevant to the article [48].

Data harvested from social media platforms that explicitly links to a source news article is used as follows for the purposes of query modeling. Similarly to how we modeled internal structure elements, we use the entire contents from all elements in a source to model the news article. E.g., for a Digg story that links to a news article, we take all text from the story title, from the story description and from all comments, if any, attached to the story. For blog posts that include a hyperlink to the article, we consider the text of the post in the blog's feed. For Wikipedia, we use the source article's title to retrieve the ten most relevant Wikipedia articles from a Wikipedia index and use their content to model the news article.

Using social media for query modeling purposes raises issues. First, accumulating content from multiple blog posts and Wikipedia articles can lead to noisy queries. We reduce the model size by applying a graph-based term selection method (see below). Second, looking at other social media platforms, some news articles are "comment magnets," accumulating thousands of comments. Third, with platforms that allow for the creation of hierarchical discussion threads, the relevancy of a comment to the source news article is dependent on its level in the thread. To limit potential topical noise, we perform comment selection (dependent on the platform) based on comment metadata. Next, we look at two methods for ranking comments for Digg and NYTC.

For a Digg comment dc, we consider the number of positive (up) and negative ($down$) votes, the number of replies ($replies$) to the comment and the depth ($level$) of the comment in the thread:

$$Rank(dc) = \frac{(replies + 1) \cdot (up - down)}{e^{level}}$$

The formula rewards comments with a high number of positive votes that triggered further discussion ($replies$) and that are more likely to be about the article than about other comments ($level$).

For a NYT comment nc, we consider the number of recommendations (rec), and whether nc was selected from the editors (se):

$$Rank(nc) = 2 \cdot (se + 1) \cdot rec$$

where se is a binary variable of value 1 when the comment is selected by the editors and 0 otherwise. The formula biases comment selection to highly recommended comments that are boosted further when selected from the NYT editors.

Table 2 (bottom) lists query models derived using social media.

Reduced query models. So far, we have used any and all the data identified for a data source above as "the query model generated from the source." As a consequence, these query models (when viewed as lists of words) may be lengthy, which may have a negative impact on retrieval efficiency and potentially also on effectiveness; see Table 6 (top half) for the average query length per source. Next, we aim to identify and extract terms that are discriminative, either for the source news article at hand or for the discussion surrounding it. To this end we introduce TH-Rank ("TextHitsRank"), a variation of TextRank [34]. TextRank and other graph-based ranking methods are based on the idea of "recommendation," where the importance of a vertex within a word-graph is computed using global information recursively drawn from the entire graph. Our modifications to TextRank are three-fold: how the graph is constructed, the scoring algorithm, and the cutoff threshold for the returned terms.

To construct a directed (word) graph for a document, the text is tokenized and stemmed and multi-word named entities are collapsed into a single word. Unlike TextRank (where only nouns are considered for constructing the graph), we use all terms due to the low recognition accuracy of nouns in noisy text [10]. For each token a vertex is created and an edge is added between tokens that co-occur within a window of two words. Intuitively, the edges are weighted according to the number of occurrences of a pair of tokens in the text. Words at sentence boundaries are not connected to avoid accidental recommendations.

We are not only interested in the most discriminative words, but also in their context. For this purpose, instead of the PageRank algorithm used by TextRank, we use the HITS algorithm, which makes a distinction between "authorities" and "hubs" [19], for scoring. In our setting, the authority score determines how important a word is for the article (proceeded by how many words) and the hub score reflects the word's contribution to the article's context (how many words follow it).

Table 3: Data fusion methods used in the paper.

Method	Gloss
combMAX	Maximum of individual scores
combMIN	Minimum of individual scores
combSUM	Sum of individual scores
combMNZ	combSUM × number of nonzero scores
combANZ	combSUM ÷ number of nonzero scores
WcombSUM	weighted sum of individual scores
WcombMNZ	WcombSUM × number of nonzero scores
WcombWW	WcombSUM × sum of individual weights
RR	Round-robin
RR-W	Round-robin weighted

We use a document-dependent threshold for which terms to select: from each set (authorities or hubs), we only return terms whose score is of the same magnitude as the highest scored term.

In §6, we apply TH-Rank to the following models: full, digg, nytc, wikipedia, and blogposts (Table 2).

4.3 Late fusion

Different query models potentially give rise to different ranked result lists. To arrive at a single merged result list, we use late data fusion methods. In particular, we consider the methods listed in Table 3; see [41] for a survey of these and other methods.

Let N be the set of all ranked lists n_i resulting from different query models. Let $s_{n_i}(a, u)$ be the score of an utterance u (from the target index) given a source news article a, w_{n_i} a weight assigned to n_i and N_{ret} a subset of N consisting of ranked lists that returned u. Then, combMAX considers the highest score from N, combMIN considers the lowest score from N, WcombSUM sums up all scores factored by their weight [16]:

$$score_{WcombSUM}(a, u) = \sum_{i=1}^{|N|} w_{n_i} \cdot s_{n_i}(a, u)$$

if $w_{n_i} = 1$ (for all n_i), it becomes combSUM. WcombWW is similar to WcombSUM except that final scores are multiplied by the sum of weights of the runs that returned the utterance:

$$score_{WcombWW}(a, u) = \sum_{m \in N_{ret}} w_m \times \sum_{i=1}^{|N|} w_{n_i} \cdot s_{n_i}(a, u)$$

for the special case where $w_m = 1$ (for all m), we get WcombMNZ. If we further assume $w_{n_i} = 1$ (for all n_i), we arrive at combMNZ. combANZ is similar to combMNZ but final scores are averaged over the number of runs that return the utterance $|N_{ret}|$:

$$score_{combANZ}(a, u) = \frac{1}{|N_{ret}|} \cdot \sum_{i=1}^{|N|} s_{n_i}(a, u)$$

Round-robin (RR) chooses one utterance from each ranked list, deleting any utterance if it has occurred before. Weighted round-robin (RR-W) is similar except that not all ranked lists are available at each round. Each ranked list is assigned a sampling frequency, defining every how many rounds it will be sampled.

Normalization of scores between ranked lists is required before producing the final rankings [37]. A standard practice is to first normalize the document scores per run and then merge them:

$$s_{normed,n_i}(a,u) = \frac{s_{n_i}(a,u) - \min(s_{n_i}(a))}{\max(s_{n_i}(a)) - \min(s_{n_i}(a))}.$$

We also consider a second normalization method, based on z-scoring, inspired from work in topic detection and tracking [2]:

$$s_{z\text{-}score,n_i}(a,u) = \frac{s_{n_i}(a,u) - \mu}{\sigma},$$

where μ is the mean of the document score distribution for source news article a in ranked list n_i, and σ is the standard deviation.

5. EXPERIMENTAL SETUP

We present our research questions, experiments, dataset and evaluation method. For the purpose of finding social media utterances that reference individual news articles, we choose to focus on a single target collection in our experimental evaluation, namely the blogosphere. Nothing depends on this particular choice, though. Our choice is based on the observation that blogs, unlike many other social media, are not limited to a single dominant platform like Digg or Twitter. Content found on individual social media platforms can be biased according to the platform's user demographics.

5.1 Experiments

To answer our research questions in §1 we conduct two sets of experiments, aimed at (i) query modeling and (ii) late fusion.

Performance of three families of query models In this set of experiments we answer research questions 1–3. For each of the three families (document structure, social media, and reduced models) we construct queries, and submit them to an index of blog posts. We measure the performance of each model individually, and compare the results. Analysis of the results reveals differences in performance between the individual models, and the families of models.

Performance of three late fusion types The second set of experiments is aimed at answering research questions 4 and 5. Here, late fusion techniques are applied to the ranked lists produced by the individual models. We experiment with 10 fusion methods from three types: (i) no training required, (ii) query independent training, and (iii) query dependent training. Finally, we test the utility of two different score normalization methods.

5.2 Data set and data gathering

The data set that we use as our target social media collection is the Blogs08 collection provided by TREC; the collection consists of a crawl of feeds, permalinks, and homepages of 1.3M blogs during early 2008–early 2009. This crawl results in a total of 28.4M blogs posts (or permalinks). We only used feed data, the textual content of blog posts distributed by feeds and ignored the permalinks. Two main reasons underly this decision: (i) our task is precision-oriented and benefits from a clean collection; and (ii) using feed data requires almost no preprocessing of the data. Extracting posts from the feed data gave us a coverage of 97.7% (27.8M posts extracted). As a second preprocessing step we perform language detection and remove all non-English blog posts from the corpus, leaving us with 16.9M blogs posts. Our index is constructed based on the full content of blog posts.

Our news article dataset is based on the headline collection from the top stories task in TREC 2009. This is a collection of 102,812 news headlines from the New York Times and include the article title, byline, publication date, and URL. For our experiments we extended the dataset by crawling the full body of the articles.

As auxiliary collections used in our query modeling experiments, we use data gathered from the following five platforms:

Digg: A collaborative news platform where people submit URLs that they find interesting.[1] We collected 19,608 Digg stories corresponding to the same number of articles. On average each story is associated with 26 comments.

Delicious: A social bookmarking site, where people can store the addresses of web sites they want to keep.[2] We collected 7,275 tagged articles with an average of 3 unique tags per article, summing up to 3,906 unique tags.

Twitter: We use Topsy, a real-time search engine that indexes content from Twitter, a microblogging platform where people can submit short snippets of text 140 characters long.[3] We collected tweets that mention 21,550 news articles, with each article being mentioned in 3 tweets on average.[4]

NYT Community: A web service from New York Times for retrieving comments registered on their site.[5] We collected comments for 2,037 articles with an average of 150 comments per article.

Wikipedia: The collaborative online encyclopedia. We use the Wikipedia dump that is included in the Clueweb09 collection,[6] containing almost 6 million pages.

5.3 Evaluation

The ideal ground truth for our task would consist of tuples consisting of news articles and social media utterances. As a proxy, we follow [14, 33, 35] and use items that are explicitly linked to a given news source. We then remove the explicit links and test our link generation methods by examining to which extent they succeed at identifying those explicit links. The reason for choosing this evaluation scheme is twofold: (i) the generation of such ground truth is cheaper than having human assessors judge whether a blog post is about a news article, and (ii) in this paper we are interested in examining the relative effectiveness of the suggested approaches, not in absolute numbers.

Our ground truth is assembled in two phases. First, for each news article we find blog posts that include the article's URL. Second, for each discovered blog post we look for other blog posts that include its URL. The process continues recursively until no more blog posts are discovered. For our experiments we sample headlines with more than ten explicit links and where social media possibly plays a role. For each news article, we take the temporally first five explicitly linked blog posts for using them in modeling. The remaining blog posts form the article's ground truth. This selection procedure results in 411 news articles with an average of 14 explicitly linked ("relevant") blog posts per article.[7]

[1] http://www.digg.com

[2] http://www.delicious.com

[3] http://www.topsy.com

[4] Topsy limits access to the ten most recent tweets for a URL. Consequently, the reported average might not reflect reality.

[5] http://developer.nytimes.com/docs/community_api

[6] http://boston.lti.cs.cmu.edu/Data/clueweb09/

[7] The complete ground truth may be retrieved from http://ilps.science.uva.nl/resource/linking-online-news-and-social-media.

In our experiments we use the Indri framework [32]. Each experimental condition returns the top 1,000 results. We report on standard IR measures: recall, mean reciprocal rank (MRR), mean average precision (MAP), and r-precision. Statistical significance is tested using a two-tailed paired t-test and is marked as ▲(or ▼) for significant differences for $\alpha = .01$, or △(and ▽) for $\alpha = .05$.

5.4 Weight optimization for late fusion

For late fusion methods that allow for weighted fusion, we estimate a weight w_{n_i} for each ranked list n_i using query independent and query dependent approaches.

Query independent weight optimization. Given a set of news articles and a et of ground truth assessments, we seek weights that maximize MAP over a set of source articles. For this, we conduct two fold cross-validation and split our ground truth in two sets of equal size: training (205 articles) and testing (206 articles). First, we learn weights that maximize MAP on the training set and then use these for evaluation on the test set. For estimating w_{n_i}, we follow He and Wu [16]. First, for each ranked list n_i in the training set, the MAP score map_{n_i} is computed. Then, map_{n_i} is used as weight for n_i in the test set: $w_{n_i} = map_{n_i}$. He and Wu suggest that the weight for the best individual run should be factored several times its MAP score. Fig. 3 shows that, in our setting, increasing the weight of the best individual run hurts performance.

Query dependent weight optimization. Given a news article and a ground truth, we seek weights w_{n_i} that maximize average precision (AP). Since the weights are dependent on the query, the ground truth for training and testing should be different. For building the training ground truth, we look for good surrogates of implicitly linked blog posts to use as proxy. For this purpose, for an article's training ground truth, we consider the temporally first five explicitly linked blog posts. The testing ground truth is kept the same as in query independent optimization for the results to remain comparable. In the training step, the system learns weights such that the blog posts in the training ground truth rank at the top. Then, in the testing step, we report on MAP for the testing ground truth. For estimating w_{n_i} we use maximum AP training and line search [13], where $w_{n_1}, \ldots, w_{n_n}$ is considered a set of directions in the range $[0, 1]$. We move along the first direction in steps of 0.2 so that AP is maximized; then move from there along the second direction to its maximum, and so on. We cycle through the whole set of directions as many times as necessary, until AP stops increasing.

For query dependent and query independent fusion, we combine all available ranked lists except from the blogposts model. The later is excluded because it exploits the same explicitly linked blog posts to model the news article with those used as training ground truth in query dependent fusion. Also, for models that have reduced counterparts, we select the one performing the best. This selection leads to 11 models: title, lead, ne, quote, metadata, full, digg-comm, delicious, nyt-comm, twitter, and wikipedia-graph.

6. RESULTS AND ANALYSIS

We report on our results from the experiments in §5 for query modeling and late fusion and conduct an analysis on our findings.

6.1 Query modeling

We turn to the results of our query modeling approach; each paragraph discusses one of the research questions in §1. Next, we perform an analysis of the results to gain more insight.

RQ1: Internal document structure vs. article title. Our baseline is set to the query model derived from an article's title only. This choice is supported by two reasons: First, the article's title is the most compact representation of the entire article and second, the article's title was chosen in prior research for ranking news headlines according to their mentions in the blogosphere [26].

Table 4: System performance for retrieving blog posts relevant to a source article using credibility priors and models derived from internal document structure and social media, and their reduced counterparts using TH-Rank. Significance tested against baseline (title).

runID	Recall	MRR	Rprec	MAP
Baseline				
(A) title	0.4033	0.3812	0.1488	0.1069
Model based on: **Internal document structure**				
(B) lead	0.2937▼	0.3339▽	0.1276▽	0.0886▼
(C) metadata	0.2206▼	0.1449▼	0.0466▼	0.0275▼
(D) ne	0.3739▼	0.4967▲	0.1787▲	0.1290▲
(E) quote-#1	0.2732▼	0.5101▲	0.1741▲	0.1259△
(F) full	**0.5919**▲	**0.6058**▲	**0.3190**▲	**0.2509**▲
Model based on: **Social media**				
(G) delicious	0.4122	0.2883▼	0.0875▼	0.0677▼
(H) digg	0.1108▼	0.1250▼	0.0433▼	0.0315▼
(I) digg-comm	**0.5797**▲	**0.5490**▲	**0.2508**▲	**0.2010**▲
(J) nytc	0.0072▼	0.0020▼	0.0008▼	0.0006▼
(K) nytc-comm	0.0949▼	0.0644▼	0.0160▼	0.0125▼
(L) twitter	0.1543▼	0.1150▼	0.0545▼	0.0445▼
(M) blogposts	0.1233▼	0.1289▼	0.0424▼	0.0298▼
Model based on: **Reduced using TH-Rank**				
(N) full-graph	**0.4524**	**0.5254**▲	**0.2177**▲	**0.1681**▲
(O) digg-graph	0.2799▼	0.2552▼	0.0890▼	0.0681▼
(P) nytc-graph	0.0691▼	0.0300▼	0.0122▼	0.0077▼
(Q) wikipedia-graph	0.0412▼	0.0142▼	0.0030▼	0.0020▼
(R) blogposts-graph	0.4170	0.4448△	0.1727△	0.1362▲

Table 4 (top) reports on results from models derived from the article's internal document structure. The best performing model is the one that uses the full article, namely, content from the article's title and body. The high performance of full is possibly due to blog posts picking up different aspects of the article that are not available in more compact representations such as title and lead. Both ne and quotes show a precision-enhancing effect over the baseline, at the cost of a drop in recall. Depending on the application, these representations could be an efficient alternative to full.

RQ2: Comparison of social media models over internal document structure models. Turning into models derived from social media, Table 4 (middle) shows that digg-comm, the model from Digg using only five comments (using Appendix 4.2), is performing the best among all social media models and significantly improves over title on all metrics. delicious shows a high recall possibly due to the nature of tags which are more likely to capture the article's theme rather than precisely identify it.

In general, social media models using all available content from the underlying source perform worse than models based on article internal structure. This is possibly due to noise found in user generated content, a claim supported by the improved performance of digg-comm and nytc-comm (which exploit only a subset of available content using comment selection methods) over their respective baselines (using all comments).

RQ3: Reduced query models using TH-Rank. For most query models, TH-Rank leads to improved performance. Among all reduced models, full-graph and blogposts-graph perform the best; both show significant improvements on precision-oriented metrics, without hurting recall. For full-graph, when compared to full, per-

Table 5: System performance for articles present in either twitter or nytc-comm and the baseline.

runID	Recall	MRR	Rprec	MAP
110 common topics between baseline and Twitter				
title	0.4165	0.3876	0.1667	0.1192
twitter	0.5741▲	0.4206	0.2024	0.1654▲
197 common topics between baseline and NYTC				
title	0.4091	0.3576	0.1293	0.0951
nytc-comm	0.1979▼	0.1345▼	0.0334▼	0.0261▼

Table 6: Number of queries per news article model, and their average length for query terms, phrases, and both.

runID	# queries	Average query length		
		Terms	Phrases	Total
Query based on: **Internal document structure**				
title	411	8	0	8
lead	411	23	0	23
metadata	411	8	1	9
ne	410	0	18	18
quote-#1	398	0	10	10
full	411	912	0	912
Query based on: **Social media**				
delicious	411	47	0	47
digg	411	1,476	0	1,476
digg-comm	411	225	0	225
nyt	197	15,048	0	15,048
nytc-comm	197	288	0	288
twitter	111	48	0	48
wikipedia	409	6,912	1,316	8,229
blogposts	408	617	41	658
Query based on: **Reduced using TH-Rank**				
full-graph	411	27	2	29
digg-graph	395	37	1	38
nytc-graph	197	131	1	132
wikipedia-graph	409	117	10	127
blogposts-graph	408	23	1	25

formance drops by 33% due to a significant reduction (97%) in query size. Given the low noise levels in edited text, TH-Rank sees to discard more words than required. For blogposts-graph, performance increases by an order of magnitude following a 80% reduction in query size. For blog posts, TH-Rank manages to remove noise and select terms helpful to retrieval. In both cases, TH-Rank offers a good balance between efficiency (shorter models are less computationally expensive) and effectiveness.

Next, we take a close look at the query modeling results and we perform an analysis in four directions: (i) uniqueness, (ii) silent models, (iii) NYT comments, (iv) TH-Rank, and (v) opinionatedness of articles.

Uniqueness: Besides looking at the results in terms of precision and recall, we also explore the uniqueness of runs: how many linked utterances are identified by one model in the top X, and not by any of the other models? We do so for the top 10 results and top 1,000 results. First, we observe that all models have unique results; Second, quotes-#1 is able to capture unique results in the top 10, whereas delicious does so in the top 1,000. Finally, title, full, and digg-comm capture most unique results.

Silent models: Table 6 shows that certain models, like Twitter and NYT, are silent for a large number of queries, and it is therefore difficult to assess their utility when looking at overall performance. Table 5 reports on the performance for articles present in the baseline *and* the social media model (twitter or nytc-comm); results show that the Twitter model significantly outperforms the baseline on recall metrics.

NYT comments: An interesting observation from the results is the low performance of nytc and nytc-comm, despite their strong connection to the source news article (see Tables 4 and 5). This strong connection could be the reason for their failure: news comments are usually displayed on the same page as the news article and come after it. Consequently, when people comment, there is no need to explain what news event they are referring to, give context to their opinion, or write full names of entities. This leads to a lack of discriminative and descriptive terms for that news article in the comments, potentially explaining the poor performance of news comments-based query models.

TH-Rank: Why does TH-Rank help performance for blogposts but not for other social media? Comment threads are prone to topic drift as the discussion goes on, while explicitly linked blog posts are more likely to be focusing on one topic, that of the article. Topical focus is likely to enable TH-Rank in one case to reduce noise and improve performance and in the other to capture the "general theme" of the discussion which can be far away from what triggered the discussion initially.

The same can hold for models using comment selection methods which are found to outperform their TH-Rank counterparts. Highly recommended comments are more likely to reflect what is also published in the blogosphere. On the other hand, when TH-Rank is ran on all available data from a source it proves unable to capture accurately discriminative terms for the news article, although it returns more terms for digg and nytc (lower reduction ratio, see Table 6).

Opinionatedness: We measure how opinionatedness of news articles affect the performance of individual models. In order to do so, we split our ground truth of 411 articles into 131 opinionated and 280 non-opinionated articles depending on whether the article title contains the term "OP'ED" (e.g., columns, editorials, . . .).

We perform a two-tailed independent t-test between the opinionated and non-opinionated scores for each model. For most models, performance is stable across the two articles types with full and digg-comm performing the best. In terms of recall, six of the models drop significantly, when crossing from non-opinionated to opinionated articles. title is amongst them, possibly due to static titles assigned to opinionated articles, which usually consist of the column or editorial name with only few additional terms. We also notice that digg-comm sees the highest recall on non-opinionated articles over all models, whereas this is full for opinionated articles. An interesting case is the metadata model for opinionated articles: When compared to non-opinionated articles, the recall shows a large significant increase, which is due to blog posts referring to the article author's name (and agency).

6.2 Late fusion

As before, each paragraph corresponds to one of the remaining research questions in §1. Then, we take a closer look at the results.

RQ4: Query independent late fusion. We experiment with 10 fusion methods and two document score normalization methods for the combination of 11 individual models; see §5.4. Table 7 shows that the best performing method in terms of MAP is WcombMNZ, which yields statistically significant improvements over the full model. WcombSUM, WcombWW, combSUM and combMNZ perform similar to, but slightly less than WcombMNZ, and RR-W outperforms RR.

We investigated the effect on MAP for a range of scale factors of the best individual run (full), when using z-scoring and linear

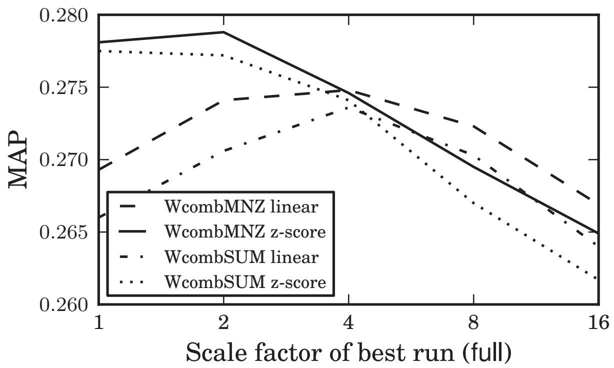

Figure 3: MAP scores for combination of 11 individual runs when increasing the weight of the best individual run for WcombMNZ and WcombSUM methods and using linear and z-score normalization of document scores.

Table 7: System performance for query independent fusion using 10 late fusion techniques on the test set using z-score normalization and combining 11 individual runs for the best scale factor (2) of full. Query dependent fusion results are reported for the best fusion method. Significance tested against full. Results from an oracle run and early fusion are also reported.

Method	Recall	MRR	Rprec	MAP
full	0.5860	**0.6196**	**0.3323**	0.2522
Query independent				
combMAX	0.7214$^\blacktriangle$	0.5871$^\triangledown$	0.2820$^\blacktriangledown$	0.2283$^\blacktriangledown$
combMIN	0.3308$^\blacktriangledown$	0.0766$^\blacktriangledown$	0.0195$^\blacktriangledown$	0.0131$^\blacktriangledown$
combSUM	0.7194$^\blacktriangle$	0.6083	0.3202	0.2665$^\vartriangle$
combMNZ	0.7265$^\blacktriangle$	0.6130	0.3252	0.2722$^\blacktriangle$
combANZ	0.6821$^\blacktriangle$	0.4547$^\blacktriangledown$	0.1574$^\blacktriangledown$	0.1256$^\blacktriangledown$
WcombSUM	0.7190$^\blacktriangle$	0.6141	0.3317	0.2772$^\blacktriangle$
WcombMNZ	0.7248$^\blacktriangle$	0.6123	0.3422	**0.2788**$^\blacktriangle$
WcombWW	0.7169$^\blacktriangle$	0.6129	0.3315	0.2723$^\blacktriangle$
RR	**0.7328**$^\blacktriangle$	0.3990$^\blacktriangledown$	0.2095$^\blacktriangledown$	0.1664$^\blacktriangledown$
RR-W	0.7298$^\blacktriangle$	0.3999$^\blacktriangledown$	0.2358$^\blacktriangledown$	0.1882$^\blacktriangledown$
Query dependent				
WcombMNZ	0.7011$^\blacktriangle$	0.6148	0.3277	0.2646$^\vartriangle$
Analysis				
Oracle	0.6388	0.7727	0.3645	0.3141
Early fusion	0.5331	0.5356	0.5220	0.1956

normalization of document scores. Fig. 3 illustrates that, in our setting, WcombMNZ with z-scoring and the scale factor set to 2 achieves the best MAP among all fusion methods.

RQ5: Query dependent late fusion. For this experiment we use the best performing late fusion method from RQ4: WcombMNZ with z-scoring. The goal here is to learn weights that maximize average precision for a training ground truth.

From Table 7 we can see that query dependent fusion significantly outperforms full, but performs slightly worse than query independent fusion. One reason for this can be that the nature of relevant blog posts is evolving as we move farther in time from the source article publication date.

Next, we proceed with an analysis of: (i) query dependent vs. independent fusion, (ii) an oracle run, and (iii) early fusion.

Query dependent vs. independent fusion: In theory, query dependent fusion was expected to outperform other methods because of how weights were optimized. For each individual article, weights were estimated to maximize average precision. However, query independent fusion showed to perform better. The two methods differ in the training set. For query independent fusion each article in the training set was on average associated with 14 blog posts. For query dependent fusion, weights were estimated for a ground truth of 5 blog posts per article. It is, therefore, interesting to explore the utility of a larger sample of explicitly linked blog posts as training ground truth or to seek time dependent evolution patterns in the weights assigned to each ranked list.

Oracle run: For each source article we take the ranked list produced from the best performing model according to average precision, and combine these into a final "oracle" run. Since we only use one model per source news article and no mixture of models, this run does not achieve the maximum performance possible. Still, the oracle run gives an indication of what scores are achievable. Comparing the performance of the oracle run (Table 7) to WcombMNZ, the best performing query independent fusion method, we observe that the latter arrives remarkably close to the oracle run.

Early fusion: Belkin et al. [7] conducted thorough experiments comparing performance of early and late fusion techniques. They found that combining individual models at query time performance increased compared to individual models. As a check we use the best performing model from each family of query models and combine them in a query. For the reduced models of Wikipedia and blog posts, each term is weighted according to its hub or authority score. The performance of this run (again, Table 7) is less than the best individual run (i.e., full) and the query independent and dependent fusion methods (i.e., WcombMNZ). The lower performance is likely due to noise brought in after combing all models and suggests that term selection and term weighting methods on the combined query hold potential for improving retrieval effectiveness.

7. CONCLUSIONS

Most of the methodological contributions of this paper lie in news article modeling for the task of *discovering implicitly linked social media utterances*. We study retrieval effectiveness of multiple query models that exploit content from individual elements in article's internal structure and from explicitly linked utterances from six social media platforms. Experimental evidence shows that query models based on the entire article perform the best. However, query models from social media bring in previously unseen utterances. Query models trained on anchor text from explicitly linked blog posts are interesting to explore, however our current experimental setup constraints us from further investigating their utility. For lengthy models, we introduce TH-Rank, an unsupervised graph-based method for selecting the most discriminative terms from each model. We demonstrate that content selection helps to improve both effectiveness and efficiency. Next, we study the effect of combining ranked lists from individual query models and we experiment with ten late data fusion methods and two document score normalization methods. We also study the impact on effectiveness of query dependent and query independent weight optimization schemes. We found that fusion methods improve significantly when using z-scoring normalization. Query independent weight optimization helped WcombMNZ to outperform all individual and fusion runs and to achieve performance remarkably close to an oracle run.

In future work we plan stricter recency conditions to our retrieval model, study the potential of query dependent fusion in more detail, compare our models to typical IR approaches such as BM25F, and experiment with additional target indexes such as Twitter. Results from this work and its future extensions lay the ground work for discovering social media utterances related to a topic of a group of news stories.

Acknowledgments. This research was supported by the European Union's ICT Policy Support Programme as part of the Competitiveness and Innovation Framework Programme, CIP ICT-PSP under grant agreement nr 250430 (GALATEAS), by the 7th Framework Program of the European Commission, grant agreement no. 258191 (PROMISE), by the DuOMAn project carried out within the STEVIN programme which is funded by the Dutch and Flemish Governments under project nr STE-09-12, by the Netherlands Organisation for Scientific Research (NWO) under project nrs 612.-066.512, 612.061.814, 612.061.815, 640.004.802, 380-70-011 and by the Center for Creation, Content and Technology (CCCT).

8. REFERENCES

[1] E. Adar, L. Zhang, L. Adamic, and R. Lukose. Implicit structure and dynamics of blogspace. In *WWE '04*, 2004.

[2] J. Allan, editor. *Topic detection and tracking: event-based information organization*. Kluwer Acad. Publ., 2002.

[3] J. Arguello, J. L. Elsas, J. Callan, and J. G. Carbonell. Document representation and query expansion models for blog recommendation. In *ICWSM '08*, 2008.

[4] K. Balog, G. Mishne, and M. de Rijke. Why are they excited? In *EACL '06*, pages 207–210, 2006.

[5] H. Becker, M. Naaman, and L. Gravano. Learning similarity metrics for event identification in social media. In *WSDM '10*, pages 291–300. ACM, 2010.

[6] S. Beitzel, E. Jensen, A. Chowdhury, D. Grossman, O. Frieder, and N. Goharian. Fusion of effective retrieval strategies in the same information retrieval system. *J. Amer. Soc. Inform. Sci. and Techn.*, 55:859–868, 2004.

[7] N. Belkin, P. Kantor, E. Fox, and J. Shaw. Combining the evidence of multiple query representations for information retrieval. *Inform. Proc. and Manag.*, 31:431–448, 1995.

[8] A. Bell. *The Language of News Media*. Language in Society. Blackwell, Oxford, September 1991.

[9] H. Chen. Collaborative systems: Solving the vocabulary problem. *Computer*, 27(5):58–66, 1994. ISSN 0018-9162.

[10] L. Dey and S. K. M. Haque. Studying the effects of noisy text on text mining applications. In *AND '09*, pages 107–114, New York, NY, USA, 2009. ACM.

[11] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. *Commun. ACM*, 30(11):964–971, 1987.

[12] M. Gamon, S. Basu, D. Belenko, D. Fisher, M. Hurst, and A. C. König. Blews: Using blogs to provide context for news articles. In *ICWSM '08*, 2008.

[13] J. Gao, H. Qi, X. Xia, and J. yun Nie. Linear discriminant model for information retrieval. In *SIGIR '05*, pages 290–297. ACM Press, 2005.

[14] S. Geva and A. Trotman. INEX 2010 Link-The-Wiki Track, 2010. `http://www.inex.otago.ac.nz/`.

[15] D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins. Information diffusion through blogspace. In *WWW '04*, pages 491–501, New York, NY, USA, 2004. ACM.

[16] D. He and D. Wu. Toward a robust data fusion for document retrieval. In *IEEE NLP-KE'08*, 2008.

[17] D. Ikeda, T. Fujiki, and M. Okumura. Automatically linking news articles to blog entries. In *AAAI Spring Symp.*, 2006.

[18] A. Java, X. Song, T. Finin, and B. Tseng. Why we twitter: understanding microblogging usage and communities. In *WebKDD/SNA-KDD '07*, pages 56–65. ACM, 2007.

[19] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[20] A. C. König, M. Gamon, and Q. Wu. Click-through prediction for news queries. In *SIGIR '09*, 2009.

[21] R. Kumar, J. Novak, P. Raghavan, and A. Tomkins. Structure and evolution of blogspace. *Com. ACM*, 47(12):35–39, 2004.

[22] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *WWW '10*, pages 591–600, New York, NY, USA, 2010. ACM.

[23] LDC. The New York Times Annotated Corpus, 2008.

[24] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst. Cascading behavior in large blog graphs. In *SIAM International Conference on Data Mining 2007*, 2007.

[25] J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD '09*, pages 497–506, New York, NY, USA, 2009. ACM.

[26] C. Macdonald, I. Ounis, and I. Soboroff. Overview of the TREC 2009 blog track. In *TREC '09*, 2010.

[27] M. Mathioudakis, N. Koudas, and P. Marbach. Early online identification of attention gathering items in social media. In *WSDM '10*, pages 301–310. ACM, 2010.

[28] M. McCabe, A. Chowdhury, D. Grossman, and O. Frieder. System fusion for improving performance in information retrieval systems. In *ITCC 2001*, 2001.

[29] R. McCreadie, C. Macdonald, and I. Ounis. News article ranking: Leveraging the wisdom of bloggers. In *RIAO '10*, 2010.

[30] J. McLean. State of the blogosphere, 2009. `http://technorati.com/blogging/article/state-of-the-blogosphere-2009-introduction`.

[31] D. Metzler, Y. Bernstein, W. B. Croft, A. Moffat, and J. Zobel. Similarity measures for tracking information flow. In *CIKM '05*, pages 517–524, New York, NY, USA, 2005. Acm.

[32] D. Metzler and W.B. Croft. Combining the Language Model and Inference Network Approaches to Retrieval IPM Special Issue on Bayesian Networks and Information Retrieval, 40(5), pages 735–750, 2004.

[33] R. Mihalcea and A. Csomai. Wikify!: linking documents to encyclopedic knowledge. In *CIKM '07*, pages 233–242, 2007.

[34] R. Mihalcea and P. Tarau. TextRank: Bringing order into texts. In *EMNLP '04*, July 2004.

[35] D. Milne and I. H. Witten. Learning to link with Wikipedia. In *CIKM '08*, pages 509–518, 2008.

[36] G. Mishne and M. de Rijke. A study of blog search. In *ECIR 2006*, volume 3936 of *LNCS*, pages 289–301. Springer, 2006.

[37] M. Montague and J. A. Aslam. Relevance score normalization for metasearch. In *CIKM '01*, pages 427–433. ACM, 2001.

[38] I. Ounis, M. de Rijke, C. Macdonald, G. Mishne, and I. Soboroff. Overview of the Trec-2006 blog track. In *TREC 2006*. NIST, 2007.

[39] O. Phelan, K. McCarthy, and B. Smyth. Using twitter to recommend real-time topical news. In *RecSys '09*, pages 385–388, New York, NY, USA, 2009. ACM.

[40] H. Sayyadi, M. Hurst, and A. Maykov. Event detection and tracking in social streams. In *ICWSM '09*, 2009.

[41] J. A. Shaw and E. A. Fox. Combination of multiple searches. In *TREC 1992*, pages 243–252, 1993.

[42] C. G. M. Snoek, M. Worring, and A. W. M. Smeulders. Early versus late fusion in semantic video analysis. In *MULTIMEDIA '05*, pages 399–402. ACM, 2005.

[43] G. Szabó and B. A. Huberman. Predicting the popularity of online content. *CoRR*, abs/0811.0405, 2008.

[44] Y. Takama, A. Matsumura, and T. Kajinami. Visualization of news distribution in blog space. In *WI-IATW '06*, pages 413–416, Washington, DC, USA, 2006. IEEE Computer Society.

[45] M. Thelwall. Bloggers during the london attacks: Top information sources and topics. In *WWE '06*, 2006.

[46] E. Tsagkias, M. de Rijke, and W. Weerkamp. Predicting the volume of comments on online news stories. In *CIKM '09*, Hong Kong, November 2009. ACM.

[47] W. Weerkamp and M. de Rijke. Credibility improves topical blog post retrieval. In *ACL-08: HLT*. ACL, June 2008.

[48] W. Weerkamp, K. Balog, and M. de Rijke. A generative blog post retrieval model that uses query expansion based on external collections. In *ACL-ICNLP 2009*, August 2009.