Real-Time Recommendation of Diverse Related Articles

Sofiane Abbar
QCRI, Doha, Qatar
sabbar@qf.org.qa

Sihem Amer-Yahia
CNRS, LIG, France
Sihem.Amer-Yahia@imag.fr

Piotr Indyk
MIT, Cambridge, USA
indyk@csail.mit.edu

Sepideh Mahabadi
MIT, Cambridge, USA
mahabadi@mit.edu

ABSTRACT

News articles typically drive a lot of traffic in the form of comments posted by users on a news site. Such user-generated content tends to carry additional information such as entities and sentiment. In general, when articles are recommended to users, only popularity (e.g., most shared and most commented), recency, and sometimes (manual) editors' picks (based on daily hot topics), are considered. We formalize a novel recommendation problem where the goal is to find the closest most diverse articles to the one the user is currently browsing. Our diversity measure incorporates entities and sentiment extracted from comments. Given the real-time nature of our recommendations, we explore the applicability of nearest neighbor algorithms to solve the problem. Our user study on real opinion articles from *aljazeera.net* and *reuters.com* validates the use of entities and sentiment extracted from articles and their comments to achieve news diversity when compared to content-based diversity. Finally, our performance experiments show the real-time feasibility of our solution.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Selection process

General Terms

Algorithms, Design, Experimentation

Keywords

Diversity, Recommender System, Social Media

1. INTRODUCTION

People are more inclined than ever to express their opinion online and news articles are typical pieces that drive traffic in the form of comments posted by users on news sites. Such comments contain rich information such as entities (e.g., Person, Organization), and often express an agreeing or disagreeing opinion on the content of the article. Yet, the most common dimensions used to recommend articles to users are popularity (e.g., most shared and most commented), recency, and (manual) editors' picks (based on daily hot topics). We propose to account for users' comments in choosing articles to recommend. While recommended restaurants or products are ones that received mostly positive reviews, news articles that drive most traffic may be ones that are most diverse in the sentiment or entities expressed in their comments. We propose to explore the effect of sentiment and entities mentioned in comments on news diversity.

We are interested in a common scenario in which a user is reading a news article and related articles are recommended on-the-fly. Such recommendations are typically chosen based on overlap in content with the current article, recency, user activity (such as emailing and commenting) and on that day's editor's picks. Figures 1, 2 and 3 show screenshots of related articles on popular news sites, Al Jazeera English, CNN, and the BBC, respectively. As it can be seen on the examples, a variety of tabs that reflect different semantics such as "Most Viewed" and "Most Watched" enable the exploration of articles related to the one currently being browsed. Our discussions with Al Jazeera editors and journalists led to the conclusion that none of the existing news sites uses the content of users' comments to select the set of related articles to recommend. We conjecture that the use of sentiment and entities expressed in comments will enable a good diversification of recommended articles. We use that as a basis to define *a novel news recommendation problem* that relies on *achieving a balance between relevance and diversity*. Relevance is computed with respect to an article currently browsed by the user. Diversity is based on a pairwise distance between articles according to user-generated content in the form of entities and sentiment contained in comments. To the best of our knowledge, our work is the first formalization of the problem of real-time recommendation of diverse news articles that accounts for relevance to an input article and for users' comments to achieve diversity between recommended articles.

Given a set of articles R_a that are relevant to an input article a, we would like to identify a subset $S_a \subseteq R_a$ of size k such that the audience's comments for the collection of articles S_a are *most attractive* among all possible subsets of R_a of size k. We capture different ways of expressing news attractiveness by incorporating entities and sentiment in a pairwise distance between articles. Our problem then becomes finding candidate articles that are within a relevance distance r from a given article a and among those,

Copyright is held by the International World Wide Web Conference Committee (IW3C2). IW3C2 reserves the right to provide a hyperlink to the author's site if the Material is used in electronic media.
WWW 2013, May 13–17, 2013, Rio de Janeiro, Brazil.
ACM 978-1-4503-2035-1/13/05.

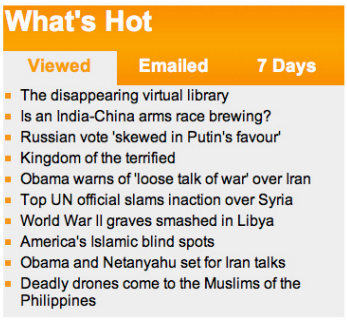

Figure 1: What's Hot on Al Jazeera English

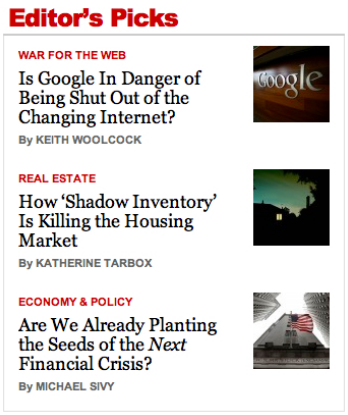

Figure 2: Editor's Picks on CNN

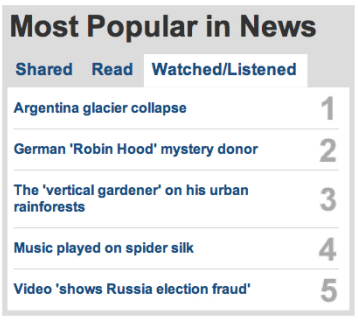

Figure 3: News Popularity on the BBC

identifying *the k most diverse articles* according to a diversity measure that captures coverage of different entities or user sentiment in comments associated to recommended articles. Section 2 contains our formalism, the different diversity functions that we propose to explore, a definition of our problem and a study of its complexity.

Given the real-time nature of our recommendations and the high complexity of our problem [1], we propose to adapt Locality Sensitive Hashing (LSH) techniques [11] to find candidate nearest neighbors of an input article a. LSH precomputes buckets of similar articles based on pre-defined hash functions. Diversity is achieved in a second stage by looking for a subset of k articles among related ones that maximizes a diversity distance. This problem refers to the dispersion problem under MAX-MIN criterion known to be NP-hard [13]. Hence, we adopt Ravi et al. algorithm [13] similar to Ganzalez's algorithm [10] that achieves a performance guarantee of 2, i.e., returns a set of k articles that 2-approximately maximize overall diversity. Obtaining a performance guarantee less than 2 is proven to be NP-hard [10, 13]. We refer to the resulting algorithm as $dLSH$.

In applications with a large number of similar articles, the size of precomputed buckets will be large and will affect the performance of finding candidate articles in $dLSH$. To overcome this problem, we develop $dLSH^*$, an algorithm that pushes diversity into building buckets. The intuition behind $dLSH^*$ is to reduce the size of candidate articles by precomputing only *potentially interesting* articles. For each bucket, it precomputes and stores at most k articles that (d-approximately) maximize the diversity of the bucket. $dLSH^*$ retrieves at most $L \times k$ articles where L is the number of buckets thereby reducing the complexity of $dLSH$. Section 3 contains a detailed description of our algorithms.

Our user study on real opinion articles on Al Jazeera English and on articles from Reuters compared the use of entities and sentiment contained in users' comments for diversifying recommended articles to diversification based on articles' content and showed the superiority of users' comments. Our performance experiments show the superiority of our algorithms when compared to naive ones without compromising precision. In particular, in many cases, pushing diversity into LSH buckets ($dLSH^*$) results in better diversity and execution time improvement with a small relevance loss. Section 4 provides a detailed summary of our experiments. Finally, related work is given in Section 5, and conclusion and future work in Section 6.

2. DATA MODEL AND PROBLEM

We define a set $\mathcal{A}$ of articles and a set $\mathcal{U}$ of users. Each article $a \in \mathcal{A}$ has a unique identifier a_{id} and a set of users $a_U \subseteq \mathcal{U}$ who posted comments on a. A comment is of the form $[a_{id}, u_{id}, text]$ where u_{id} is a unique user identifier. Without loss of generality, we assume a user posts at most one comment per article. The set of all comments on an article a form a discussion denoted by $discuss(a)$.

A news article a could also be characterized by a set of attributes such as topic, date, authors, length, and nature (e.g., opinion article, survey). Similarly, a user u could carry demographics information such as geographic location, gender, age and occupation.

2.1 Pairwise Relevance and Diversity

We are interested in two kinds of distances between articles. Relevance distance determines dissimilarity between two articles, an input article a and an article in $\mathcal{A}$, and is used to find a set of articles related to a. Diversity distance, on the other hand, determines how "different" two articles are. In this section, we start with a definition of pairwise relevance, then we define five different distances to measure diversity between two articles. The four former diversities rely on user comments on articles. The latter is defined on articles' content and will serve as a basis for comparison to assess the effectiveness of using comments to achieve diversity between retrieved articles (see Section 4.5).

2.1.1 Pairwise Relevance

Given an article a and a threshold radius r, the set of articles relevant to a is defined as the set of all articles $a_i \in \mathcal{A}$ within relevance distance r from a, i.e., $d_{rel}(a_i, a) \leq r$. For example, the distance between two articles (represented by their sets of features x and y) could be defined as $d_{rel}(a_i, a) = 1 - Jaccard(x, y)$. Jaccard coefficient measures the similarity between two sets as the size of the intersection divided by the size of the union of the sets. In this paper, features are extracted by running Open Calais(OC) [1] on an article and extracting entities mentioned in them. There-

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

fore,
$d_{rel}(a_i, a) = 1 - Jaccard(OC(a_i), OC(a))$.

2.1.2 Pairwise Comment-based Diversity

We define diversity between two articles a_i and a_j as a function of discussions engaged by users through their comments on those articles.

Using Entities: For each article a, we characterize its discussion, $discuss(a)$, with a set of features: topics (e.g., Politics, Sport, etc.) and named entities (Persons, Countries, Cities) to which it refers. Recall that $discuss(a)$ is a unique document containing the set of all comments posted on a. Features are extracted by running Open Calais(OC) on $discuss(a)$. The pairwise diversity distance between two articles is defined as the Jaccard ratio between feature sets extracted from their corresponding discussions, i.e., $d_{div}(a_i, a_j) = 1 - Jaccard(OC(discuss(a_i)), OC(discuss(a_j)))$.

The intuition behind this distance is that news readers are more likely to reveal and amplify subtle differences (e.g., *journalist's leaning toward a given topic*) between two similar articles (e.g. same topic, same persons, etc.) than any content-based diversification technique will do.

Using Sentiment: Diversity between two articles a_i and a_j is a function of sentiment extracted from their comments. Given an article a, each comment $c_i \in discuss(a)$ is assigned a sentiment value $s_i \in [-1, 1]$ which states whether the comment expresses a negative ($s_i < 0$), positive ($s_i > 0$), or neutral ($s_i = 0$) opinion. A typical method for extracting sentiment from text is to use a dictionary of positive and negative words and count the number of occurrences of those words in the text. Such dictionaries (e.g., UPitt [2], SentiStrengh [3]) and sentiment extraction implementations based on them [4], are widely available. Each article a is assigned a unique sentiment score obtained by averaging the scores s_i of all comments $c_i \in discuss(a)$. Diversity between two articles is given by: $d_{div}(a_i, a_j) = |s_{a_i} - s_{a_j}|$.
Another option to use sentiment for diversity would be to compute two sentiment scores (s_a^+, s_a^-) for each article a. s_a^+ (resp. s_a^-) is the percentage of positive (resp. negative) comments posted on a. Here, diversity between a_i and a_j could be defined as the Euclidean distance over their sentiment coordinates
$d_{div}(a_i, a_j) = \sqrt{(s_{a_i}^+ - s_{a_j}^+)^2 + (s_{a_i}^- - s_{a_j}^-)^2}$.

The intuition is that two articles are diverse if they drive different users' sentiments. In other words, the more polarized two articles are, the more diverse they are considered.

Using user IDs: Each article a is assigned a set of user identifiers $user\_IDs(a)$ who posted at least one comment on a. Every user is considered only once even if she posted more than one comment. Diversity between two articles a_i and a_j is a function of the overlap in users commenting on them, i.e. $d_{div}(a_i, a_j) = 1 - Jaccard(user\_IDs(a_i), users\_IDs(a_j))$.

The intuition here is that similar users tend to read similar articles. Indeed, the similarity of two articles read by exactly the same group of users is likely to be higher than the similarty of articles that interest different users.

Using user locations: Each article a is assigned a set of Countries $user\_countries(a)$ extracted from the profiles of users in $user\_IDs(a)$. Every country is considered only once even if many users from that country posted a comment on a. Diversity between two articles a_i and a_j is a function of the overlap in countries from where comments are issued on both of them, i.e. $d_{div}(a_i, a_j) = 1 - Jaccard(user\_countries(a_i), users\_countries(a_j))$.
The intuition here is that articles that interest users from the same region of the world are more likely to be similar than articles that interest users from different regions.

2.1.3 Pairwise Content-based Diversity

Here, we define the diversity between two articles a_i and a_j as a function of entities extracted from their content. This definition will serve as a basis in our experiments to assess the effectiveness of comment-based diversity in finding attractive articles. In particular, each article a is assigned a set of features (topics, persons, and locations) extracted from its content using Open Calais. Diversity between two articles a_i and a_j is a function of the overlap in features cited in both of them, i.e. $d_{div}(a_i, a_j) = 1 - Jaccard(OC(a_i), OC(a_j))$. Thus, two news articles on US election may be similar because they refer to the same topic (Politics) and location (USA), but still different (diverse) if one is about *Mitt Romney* and the other on *Barack Obama*.

2.2 Set-based Relevance and Diversity

The relevance $Rel(R)$ of a set of articles R with respect to an input article a is defined as

$$Avg_{a_i \in R}(1 - d_{rel}(a_i, a))$$

Set diversity answers the following question: given a set R, if we were to pick a set $S \subseteq R$ containing the k "most distinct" (as measured by the diversity distance) articles, how distinct those articles would be?

The pairwise k-diversity $div_k(R)$ of a set R is defined as

$$\max_{S \subset R, |S|=k} \min_{a_i, a_j \in S} d_{div}(a_i, a_j)$$

In above formulation, diversity is defined as the maximum over any set of size k, of the minimum pairwise distance between articles. Here, a_i plays the role of the "aggregate" or "centroid", and diversity measures the minimum distance to the centroid.

2.3 Problem and Complexity

Our goal is to return a set S_a of k most diverse articles among a set R_a formed by articles whose relevance distance to an input article a is at most r. That is, for a distance r and given a point a, we would like to find a set of k points within distance r from a (according to the relevance distance) that maximizes their pairwise k-diversity (according to the diversity distance). Given the complexity of this problem [1], we define an approximate version where the goal is to find a set of k points that d-approximates their pairwise k-diversity (i.e., the k-diversity is at least $1/d$ times the best possible). Finally, we can define the bi-criterion approximate version of the problem, where for approximation factors c and d, our goal is to find a set of k points S within distance cr from a such that $div_k(S) \geq 1/d \cdot div_k(R_a)$, where R_a is the set of points within distance r from a.

A simple way of solving this problem (for $c = 1$ and $d = 2$) would be to (i) find the set R_a of all points within relevance distance r from a, and (ii) 2−approximate $div_k(R_a)$. The former task can be done using LSH algorithm in $O(n^e)$ time

[2] http://mpqa.cs.pitt.edu/#subj_lexicon
[3] http://sentistrength.wlv.ac.uk/

for some exponent $e \prec 1$[11], while the latter task can be done using e.g., the 2-approximate *GMM* algorithm [10] in $O(|R_a|k)$ time. A nice feature of the latter algorithm is that it works for any diversity distance function satisfying metric properties (i.e., symmetry and triangle inequality). Our first solution is to implement and combine such above algorithms into a single algorithm we call $dLSH$ and that runs in $O(n^e + |R_a|k)$ time. As, the $O(n^e + |R_a|k)$ time can be too large in some applications. we present another algorithm ($dLSH*$) to improve the time to $O(L^2)$, where L is the number of *locality-sensitive hash functions* needed to return all points within distance (approximately) r from a. Typically, we have $L << n$.

3. ALGORITHMS

Our approach has two steps: *i.* identify the set R_a of candidate articles which are within a relevance distance r from an input article a, and *ii.* find among R_a the subset S_a of articles that maximizes diversity. We propose two algorithms $dLSH$ and $dLSH^*$. $dLSH$ is a direct application of Locality-Sensitive Hashing (LSH) [11] on top of which we implemented a greedy algorithm to solve the maximization of minimum pairwise distances problem. In $dLSH^*$, we pushed diversity into LSH by precomputing the k most diverse elements of each LSH bucket. This makes the algorithm faster than $dLSH$. Furthermore, we show in [1] that $dLSH^*$ achieves a 6-approximation of $dLSH$.

3.1 $dLSH$: Diversity-aware LSH Algorithm

We start with an overview of the LSH algorithm [11]. Essentially, the algorithm attempts to find all candidate articles composing R_a. This is done as follows: during an offline process, LSH creates l hash functions $g_1 \ldots g_l$, and their corresponding hash tables $A_1 \ldots A_l$. Each function g_i is a combination of k hash functions h_j (i.e., $g_i = h_1 \ldots . h_k$.) Then each article a is stored in bucket $g_i(a)$ of A_i for all $i = 1 \ldots L$ (lines 1-4, Algorithm 1). The hash functions have the property that, for any article a, we have

$$R_a \subset A_1(g_1(a)) \cup \ldots \cup A_l(g_1(a))$$

It should be noted that functions with the above property (for low values of k and l) are known to exist only for specific relevance distance functions, such as the Euclidean distance and dot product (see [5] for more details).

At runtime, for any input article a, LSH simply recovers all points in $A_1(g_1(a)) \cup \ldots A_l(g_l(a))$ (lines 1-3, Algorithm 3), and retains those that belong to R_a (lines 4-7).

Algorithm 1 dLSH: offline process

Input: $\mathcal{G} = \{g_1, \ldots, g_L\}$: set of L hashing functions, $\mathcal{D}$: collection of articles. s: size of precomputed diverse buckets.
Output: $\mathcal{A} = \{A_1, \ldots, A_L\}$: set of L hash arrays.
1: $R_a \leftarrow \emptyset$
2: **for all** $a \in \mathcal{D}$ **do**
3: **for all** $g_i \in \mathcal{G}$ **do**
4: $A_i[g_i(a)] \leftarrow A_i[g_i(a)] \cup \{a\}$
5: **return** $\mathcal{A}$

As pointed in Section 2, *relevance* distance between two articles a_i and a_j both characterized by a set of features is defined as: $d_{rel}(a_i, a_j) = 1 - Jaccard(OC(a_i), OC(a_j))$. An efficient way to estimate the Jaccard coefficient is to use min-hash functions based on min-wise independent permutations. A min-hash function $h_\pi \in \mathcal{H}$ could be defined as follows: Let $OC(a_i)$ be the set of features (topics, persons, and locations) extracted from article a_i, $\mathcal{E} = OC(a_1) \cup \ldots OC(a_n)$ the union of sets of features of all articles, $S_\mathcal{E}$ the set of all permutations of $\mathcal{E}$, and π a permutation uniformly chosen at random over $S_\mathcal{E}$.

$$h_\pi(a) = Min\{\pi(x_i), \forall x_i \in OC(a)\}$$

Where $\pi(x_i)$ is the index of the entity x_i within permutation π. Such min-hash functions guarantee the following property [7]:

$$Pr(h_\pi(a_i) = h_\pi(a_j)) = Jaccard(NE(a_i), NE(a_j))$$

Based on $\mathcal{H}$, we define each hash function g_j as a composition of K min-hash functions $h_\pi i$,

$$g_j(a) = [h_{\pi 1}(a) \times h_{\pi 2}(a) \cdots \times h_{\pi k}(a)] \,\%\Phi$$

where Φ is a predefined prime.

Enabling Diversity: Greedy Max-Min Algorithm.

To enable diversity we need an algorithm that identifies a subset of k most diverse articles among a set of n articles. According to our definition in Section 2.2, given a set S of n articles, and a diversity distance d_{div}, the most diverse subset of k articles is the one that maximizes the minimum pairwise (diversity) distance. This problem refers to the dispersion problem under MAX-MIN criterion known to be NP-hard [13]. To solve this problem, we adopted the greedy algorithm proposed by Ravi et al. [13] similar to Ganzalez's algorithm [10] that achieves a performance guarantee of 2 in the case of distances satisfying the triangular inequality. Note that obtaining a performance guarantee less than 2 is proven to be NP-hard [10, 13].

Algorithm 2 *GMM*: Greedy MaxMin algorithm

Input: D: set of articles, d_{div}: distance function, k: number of requested articles
Output: C: set of k articles which x-approximately maximizes the min pairwise distance
1: $C \leftarrow \emptyset$
2: find $\{a, b\} | (a, b) \in D^2 \wedge \forall (c, d) \neq (a, b), d_{div}(a, b) \geq d_{div}(c, d)$
3: $C \leftarrow C \cup \{a, b\}$
4: **repeat**
5: find $e | e \in D - C \wedge \forall f \in D - C, min_{x \in C}\{d(e, x)\} \geq min_{y \in C}\{d(f, y)\}$
6: $C \leftarrow C \cup \{e\}$
7: **until** $|C| = k$
8: **return** $\mathcal{C}$

Algorithm 2 optimizes for the maximal minimum pairwise distance. It starts by adding to the result set the two articles with the maximum pairwise distance (lines 2-3, Algorithm 2). Then, it adds at each iteration the article which maximizes the minimum pairwise distance with articles already selected (lines 4-6). The algorithm stops when the number of articles in the result set is k (line 7).

Online recommendation.

Algorithm 3 describes the way diverse recommendations of related articles are processed. Given an input article a, a distance radius r, and hash tables $\{A_1, \dots A_L\}$ created in the offline step; the algorithm does the following: First, it retrieves all candidate articles that are hashed into the same buckets as a (lines 1-3). Second, it filters those articles that are out of the ball $B(a, r)$ (lines 4-6). Finally, the algorithm runs *GMM* to pick the d-approximately k most diverse articles (lines 7-8).

Algorithm 3 dLSH: online process

Input: $\mathcal{G} = \{g_1, \dots, g_L\}$: set of L hashing functions, $\mathcal{A} = \{A_1, \dots, A_L\}$: set of L hash arrays, $\mathcal{D}$: collection of articles, a: input article, r: radius distance, k:number of requested articles.

Output: S_a: set of d-approximately k most diverse articles within a distance r from a.

1: $R_a \leftarrow \emptyset$
2: **for all** $g_i \in \mathcal{G}$ **do**
3: $\quad R_a \leftarrow R_a \cup A_i(g_i(a))$ {retrieve all candidate answers}
4: **for all** $p_j \in R_a$ **do**
5: $\quad$ **if** $dis_{div}(a, p_j) \succ r$ **then**
6: $\quad\quad R_a \leftarrow R_a - \{p_j\}$ {Remove irrelevant answers from result set}
7: $S_a \leftarrow GMM(R_a, k)$
8: **return** S_a

3.2 $dLSH^*$: Pushing diversity into LSH buckets

As pointed earlier, the main drawback of $dLSH$ is that in several cases the $O(n + |R_a|\, k)$ time complexity is too high, especially in applications where the data distribution results in many dense buckets. To overcome this problem, we present $dLSH^*$, an algorithm that considers diversity earlier in the recommendation process. The intuition behind $dLSH^*$ is to reduce the size of R_a by storing only *potentially interesting* articles in hash tables $A_1, \dots A_L$. Suppose that we request k articles that d-approximately maximize the diversity of R_a. For each bucket $A_i[j]$, we precompute and store the set $A'_i[j]$ of at most k articles that (d-approximately) maximize the diversity of $A_i[j]$. This is done by using *GMM* algorithm during the offline process of $dLSH^*$ (lines 5-7, algorithm 4). The online process of $dLSH^*$ is similar to the one of $dLSH$ (see Algorithm 3) except that $dLSH^*$ retrieves at most $L \times k$ articles stored in buckets $A'_i(g_i(a)), i = 1 \dots L$. Here again, *GMM* is used to find the d-approximately k most diverse articles among those $L \times k$ article. By doing so, the complexity in time drops to $O(Lk)$, where L is the number of *locality-sensitive hash functions* and k is the number of desired articles.

4. EVALUATIONS

We conducted different experiments to assess the efficiency and effectiveness of our proposals.

Implementation setup Our algorithms are implemented in JDK 6 with 2GB virtual memory. All experiments were conducted on an Intel dual core workstation with 4GB Memory, running Windows 7 OS (64-bit). Obtained results are the average of three separate runs.

Algorithm 4 $dLSH^*$: offline process

Input: $\mathcal{G} = \{g_1, \dots, g_L\}$: set of L hashing functions, $\mathcal{D}$: collection of articles. s: size of precomputed diverse buckets.

Output: $\mathcal{A} = \{A_1, \dots, A_L\}$: set of L hash arrays.

1: $R_a \leftarrow \emptyset$
2: **for all** $a \in \mathcal{D}$ **do**
3: $\quad$ **for all** $g_i \in \mathcal{G}$ **do**
4: $\quad\quad A_i[g_i(a)] \leftarrow A_i[g_i(a)] \cup \{a\}$
5: **for all** $A_i \in \mathcal{A}$ **do**
6: $\quad$ **for** $j = 0; j < size(A_i); j++$ **do**
7: $\quad\quad A_i[j] \leftarrow GMM(A_i[j], s)$
8: **return** $\mathcal{A}$

Datasets We conducted our experiments on two different datasets: a collection of AlJazeera English (AJE) opinion articles and a collection of Reuters news articles.

- ***AJE opinion articles***[4] This dataset contains 2040 articles published between April 24^{th}, 2010 and February 07^{th}, 2012. Each article comes with a set of comments ($389k$ in total) posted by $35k$ different users from 179 different countries. We characterized each article by its features (*topics, persons*, and *locations*) extracted using Open Calais. On average, each article had around 7.5 different features distributed as follows: 1.42 *topics*, 2.58 cited *persons*, and 3.5 cited *locations*.

- ***Reuters news articles***[5] We crawled 13K news articles published by Reuters between December 4^{th}, 2010 and December 20^{th}, 2011. We ran Open Calais [6] to identify article features. For each article, we identified an average of 5.19 features with 1.6 *topics*, 1.85 cited *persons*, and 1.74 cited *locations*. Note that user comments are not available for Reuters articles.

4.1 Our Baselines

We introduce in this section three algorithms that we implemented and used in the experiments to evaluate different aspects of our diversification proposals.

Exact Near Neighbors Algorithm Referred to as r_NN is an algorithm that given an input article a and a distance radius r, scans the whole data collection and retrieves *all* those articles within a distance r from a.

Approximate k Nearest Neighbor Algorithm Referred to as k_NN is an algorithm that given an input article a and a list of candidate relevant articles R_a (generated by LSH), ranks articles in R_a according to their relevance to a and picks the top k among them. This algorithm is approximate because in relies of R_a rather than the whole data collection.

MMR **Algorithm** MMR is an iterative algorithm that strives to reduce redundancy while maintaining relevance of retrieved articles[8]. MMR is based on the concept of Marginal Relevance (MR) that is a linear combination of the relevance of an articles wrt an input one and its novelty wrt already selected ones. The MR score of article a_i w.r.t.

[4]http://www.aljazeera.com/indepth/opinion/
[5]http://www.reuters.com
[6] http://www.opencalais.com

an input article a and a set of articles already selected S is given by:

$$MR(a_i) = \lambda \times sim(a_i, a) - (1 - \lambda) \times max_{a_j \in S} sim(a_i, a_j)$$

Then, at each iteration, the algorithm picks the article not yet selected whose MR score is maximal.

4.2 Results Summary

Our first experiment (Section 4.3) showed that $dLSH$ exhibits a strong performance in terms of execution time compared to a brute force r_NN. $dLSH$ is 22 times faster in nearest neighbor retrieval and 30.38 times faster in recommending diverse related articles compared to MMR. This result makes our approach suitable for a real-time recommendation scenario. Second, exploiting user-generated data, comments in our case, to diversify articles leads to a good balance between relevance and diversity (Section 4.4). The comparison result of $dLSH$ and MMR strengthens this observation as it shows clearly that $dLSH$ preserves relevance while providing an acceptable diversification compared to MMR. Moreover, we found that the diversity gain induced by $dLSH$ when compared to k_NN, is much more important than relevance loss especially in the case of low values of r. For instance, $dLSH@r = 0.5$ under *sentiment*-based diversification increases recommendation diversity by 37.46% while average relevance drops down by only 5% compared to the TopK lists. Finally, the user study (Section 4.5) revealed that users prefer sentiment-based diversity.

4.3 Nearest Neighbors Retrieval

We ran both LSH and an exact brute-force algorithm, r_NN, to request the set of nearest neighbor articles, R_a, that are within a relevance distance r of an input article a where a is instantiated to every article in the AJE and Reuters collections. In each run, the relevance distance r took each of these values: $\{0, 0.2, 0.4, 0.6, 0.8\}$.

For each input article a, r_NN accesses all articles in the corresponding collection and retrieves those within distance r from a. For LSH, the parameters are set as follows: $L = 1$, $K = 2$, and $\Phi = 307$ (i.e. $g(p) = h_{\pi 1}(p) \cdot h_{\pi 2}(p)\%307$). So, for each article a, the algorithm first retrieves all articles stored in the bucket $(A\,[g(a)])$, then it filters articles out of the ball (R_a).

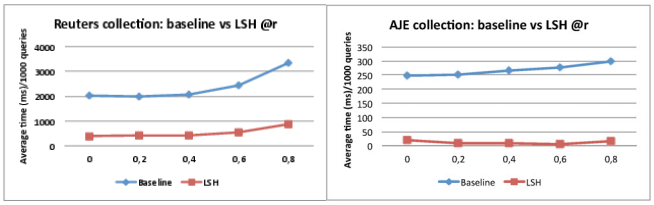

Figure 4: *exact r_NN vs. LSH @r*

In Figure 4, we show the evolution of the average execution time per 1000 input articles in both Reuters (left-hand side) and AJE (right-hand side). Not surprisingly, LSH is much faster than r_NN. On average, LSH is 4.5 times faster for Reuters, 22.5 times faster for AJE. This gain is relatively stable for all values of r. The difference in gain achieved in the two collections is due to the distribution of articles in the distance space. By analyzing the two collections, we observed that AJE articles are scattered whereas Reuters has many dense regions representing similar articles. The presence of dense regions in Reuters is reflected by the generation of a small number of LSH buckets each of which with a high filling rate. In contrast, in the sparse AJE collection, LSH generates a large number of buckets with a very low filling rate. From this observation, it becomes obvious that Reuters requires more processing (to filter out irrelevant articles) than AJE.

However, as expected, Table 1 shows that the gain in execution time affects recall. In fact, by considering the set of articles returned by r_NN as our ground truth, we computed $recall@r$ for each collection as the ratio between the number of articles returned by LSH and the number of articles returned by r_NN. For instance, Table 1 shows that for $r = 0$, LSH returns the same results as r_NN for both collections. For this specific value of r, we observed an average of 0.28 nearest neighbor articles for each AJE article and 38.3 nearest neighbors for each article in the Reuters collection. We can also see that the recall curves are drastically decreasing as r increases. This is due to the offline process in LSH during which only articles with relatively highly similarity are hashed into the same bucket. For example, when we request the set of nearest neighbors within a distance $r = 0.4$, only 30% of them are returned by LSH in both collections. To improve the recall score, one can reduce K and increase L hence creating simpler hash functions. However, for the data collection sizes we have, this will result in a significant increase in time necessary to filter out irrelevant articles.

Table 1: LSH recall@r

Dataset/ r	0.0	0.2	0.4	0.6	0.8
AJE	1.0	0.63	0.31	0.17	0.09
Reuters	1.0	0.78	0.30	0.27	0.25

4.4 Diversification Techniques

We evaluate the efficiency and effectiveness of $dLSH$ and $dLSH^*$ algorithms. As we did not have access to user comments on Reuters, we used AJE only.

Evaluation Method

For each input article a, we identify the set of candidate related articles R_a using $dLSH$ or $dLSH^*$. The radius r takes one of these values: $0.5, 0.6, 0.7$, and 0.8. For each set of candidate answers we compute two sets $divK$ and $topK$. The former is the set of k articles that d-approximately maximizes diversity generated by $dLSH$ whereas the latter is the set of k most relevant articles within R_a generated using an k_NN algorithm (both defined in section 2.2.) Basically, the approximate k_NN algorithm accesses all articles in R_a, ranks them in terms of their relevance to a, and returns the top K among them. Here also, we varied k to take different values ranging from 3 to 20 for both $divK$ and $topK$. For each set ($divK$ and $topK$), we report its diversity and average relevance scores obtained in both $dLSH$ and/or $dLSH^*$. Thus, we can estimate the loss of relevance implied by each diversification technique as well as the impact of pre-computing $divK$ of each $dLSH^*$ bucket. In what follows, we first report the results of $dLSH$ in terms of relevance and diversity of recommendations compared to k_NN. Then, we report the impact of $dLSH^*$ on results obtained by $dLSH$.

Comparing $dLSH$ to approximate k_NN

An important issue of diversification in general, is to control the loss in relevance implied by diversification. We now show that $dLSH$ leads to a substantial increase in diversity of $divK$ sets while maintaining a reasonable relevance compared to $topK$ sets (Table 2). As diversity is measured using different distances in different spaces (entities, sentiments, users, and locations), scores are not comparable. For instance, the same set $divK$ of k articles may get a high diversity score in the space of user's locations and a low score in the space of sentiments.

In order to enable the comparison of different diversification techniques, we consider content-based diversity as the ground truth. For every input article a, we retrieve its corresponding $divK$ sets using entities, sentiments, users, and locations for diversification. Then, for every $divK$ set, we report its actual diversity score measured in terms of articles content (as defined in Section 2.1.3.) This allow us to check if there is any correlation between the diversity of articles obtained in different comment spaces and the diversity of articles in the content space. More specifically, we check whether a set of articles diverse in terms of sentiments, entities, users, or user's locations is also diverse in term of articles' content (less redundancy and more novelty).

Table 2 reports the gain/loss in diversity/relevance achieved by $dLSH$ compared to approximate k_NN. For every radius value r, we report the average of scores obtained at different values of $k \in \{3, 5, 10, 15, 20\}$.

Table 2: *$dLSH$ vs. r_NN: Percentage of Gain/Loss in Diversity/Relevance @r*

	Entity-based		Sentiment-based		UserID-based		geoLocation-based	
Radius	div.	rel.	div.	rel.	div.	rel.	div.	rel.
0.5	24.05	-6.37	37.46	-5.07	31.08	-3.84	29.14	-3.87
0.6	6.63	-5.38	13.04	-4.73	10.26	-4.42	9.92	-4.34
0.7	15.63	-8.8	8.83	-8.47	11.95	-7.88	11.54	-7.87
0.8	19.64	-20.79	18.29	-20.33	7.99	-19.44	7.99	-19.5

Not surprisingly, we notice that all diversification techniques achieve a positive gain in diversity of $divK$ sets compared to $topK$ sets returned by r_NN algorithm. This means that there is a clear correlation between comment diversity as acquired through entities, sentiments, users, or locations from one side and content diversity from another. To confirm this observation, we note that maximizing the diversity with any of the techniques we proposed results in dropping the average relevance of $divK$ sets compared to $topK$ sets. Also, it seems that the best compromise between diversity and relevance is achieved in the case of $r = 0.5$ where the gain in diversity is maximized in all cases (entities, sentiments, users, and locations) while the loss in relevance is minimized in almost all cases.

There are several observations to be made regarding the pairwise comparison of the four diversification techniques. First, The *sentiment*-based approach outperforms other approaches as it implies 19.4% of diversity gain and -9.6% of relevance loss in average. This confirms the existance of a strong correlation between content diversity and opinion (sentiment) diversity, i.e., two diverse articles are more likely to generate different sentiment values than two similar articles. Second, *userID*-based and *geoLocation*-based diversification techniques achieve almost the same results. This observation is confirmed in Figure 5 where one can see that these two techniques lead to almost the same curves. Thus, we argue that it is preferable to substitute users to their locations (extracted from ip addresses) as the average number of locations per article is much smaller than the number of users who commented on that article, hence improving execution time. Finally, there is no obvious relationship between the radius r and the diversity gain achieved by the different techniques. For instance, one would expect that diversity scores increase as the value of r increases. That is not the case here because the diversity of $divK$ sets is optimized using one of the diversification techniques we proposed (using entities, sentiments, user ids, or locations) whereas diversity scores reported in Table 2 reflects the content diversity of $divK$ sets.

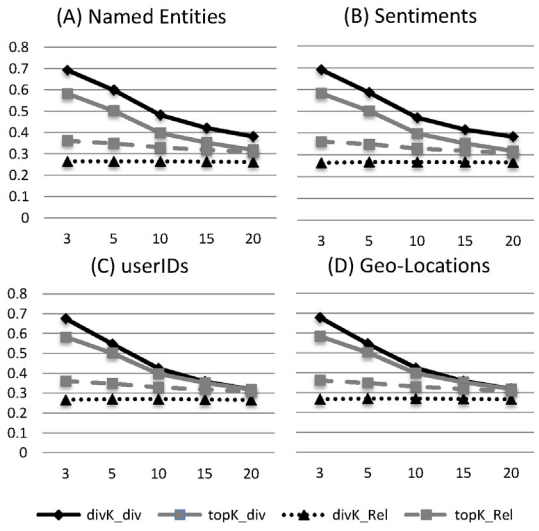

Figure 5: *Diversity of different diversification techniques*

Figure 5 summarizes results presented in the last line ($r = 0.8$) of Table 2 breakdown by value of k. Black (resp. grey) curves correspond to $divK$ (resp. $topK$) while continuous (resp. dashed) line curves correspond to diversity (resp. relevance). From this figure we observe the following. First, in all diversification techniques and for all values of k, $dLSH$ achieves a real content diversity compared to approximate k_NN. The diversity achieved is more important when using entities (A) and sentiments (B) than in the two remaining cases (C and D). Second, the average relevance of $divK$ sets is not affected by the number of requested articles (k). This unexpected behavior is due to the offline step of $dLSH$ algorithm where only relatively highly similar articles are hashed into the same bucket. That is, for a given bucket $A_i[j]$ whose size is m, and any two different numbers K_1, K_2 such that $K_1 \leq m, K_2 \leq m$ we have $relevance(divK_1) \approx relevance(divK_2)$. Note that this observation holds for $topK$ sets as well. This further corroborates our conclusion made in Nearest Neighbor retrieval experiment (section 4.3) related to the recall scores achieved by LSH

Comparing *dLSH* to *MMR*

In the previous experiments, we investigated the impact of $dLSH$ on the relevance and diversity of recommendation compared to $Topk$ approach. We have shown that in most of cases, $dLSH$ implies a significant gain in diversity against a relatively small loss in relevance. In order to have a com-

plete picture on $dLSH$, we compared it to a well-known diversification approach called Maximal Marginal Relevance (MMR) [8] (see section 4.1).

The evaluation process may be summarized as follows. For each input article, we requested different lists of $divK$ articles using $dLSH$ and MMR. All parameters (K, r, and λ) are varied to capture the different trends of each algorithm.

In the case of $dLSH$, we reported the results of the four diversity distances proposed. Figure 6 summarizes the results of comparing $dLSH$ to MMR in terms of execution time (figs.6(a), and (b)), relevance (figs.6(c),(c'),(e), and (g)), and diversity (figs.6(d), (d'), (f), and (h)).

In figs.6 (c), (c'), (d) and (d'), we plotted relevance and diversity scores achieved by $dLSH$ under different diversity distances and MMR when the number of requested articles was $K = 10$. The x-axis corresponds to the different values of $dLSH$ radius r in figures (c) and (d) whereas it corresponds to the MMR parameter λ in figurs (c') and (d'). Not surprisingly, figs. 6 (c) and (c') show that $dLSH$ relevance curves decrease as the radius r increases while the MMR curve slightly increases as λ increases. One should expect that the relevance of MMR @$\lambda = 0.9$ (≈ 0.347) should be greater than the relevance achieved by $dLSH$ @$r = 0.5$ (≈ 0.525). However, contrarily to MMR where scores are obtained by averaging relevance of recommendation lists of all articles in the dataset collection, $dLSH$ scores are obtained only from those articles having at least $K+1$ nearest neighbors within a radius r. This results in a higher relevance in the case of $dLSH$ that is always greater than or equal to r. In the remaining figs, we decided to plot results of $dLSH$ under different diversity distances and values of r along with results of MMR@0.7. Setting MMR parameter λ to 0.7 is a reasonable choice as it provides a good balance between relevance and diversity.

Figs 6 (e) and (g) strengthen the previous remark and show clearly that $relevance$@k (number of requested articles) obtained by $dLSH$ (with $r = 0.7$) under all diversity distances is substantially greater than $relevance$@k achieved by MMR. In contrast to that, figs. 6 (f) and (h) show that MMR achieves a better diversity compared to $dLSH$. In fact, as reported before, AJE collection is so scattered that there're many articles that don't have any relevant article (within a reasonable distance radius). However, unlike $dLSH$ which may return an empty list of recommendation for this kind of articles, MMR will anyways recommend a list of articles that minimize the intra-redundancy even if none of these articles is relevant enough to the input article. To summarize this experiment part, we can say that $dLSH$ is a good fit in cases where a minimum relevance threshold should absolutely be satisfied while RMM is better in cases where we are more flexible with the relevance.

Yet, another interesting result concerns the time performance of $dLSH$. Figs 6 (a) and (b) present the execution time (in milliseconds) necessary to the different algorithms in order to find recommendation lists for 100 input articles. In the former fig, we fixed $r = 0.8$ and $\lambda = 0.7$ and varied the number of requested articles K whereas in the latter fig, we fixed $K = 10$ and $\lambda = 0.7$ and varied the radius r of $dLSH$. From fig 6 (a) we can see that $dLSH$ is much more efficient than MMR. For instance, when we requested 10 ($= K$) articles, the best gain of $dLSH$ is achieved at *sentiment*-based diversification which was 30.38x times faster than MMR while the worst gain is recorded by *userID*-based $dLSH$ which was only 7.48x times faster. The slowness of *userID*-based diversification is due to the high number of unique users commenting on each article making the distance calculation more expensive. However, one can use *location*-based instead which is 24.68x faster than MMR and achieves almost the same relevance and diversity scores than *userID*-based diversification. In cases where we requested more articles ($K = 15$ and $K = 20$), the gain achieved by $dLSH$ was even bigger. This gain is due to the nature of MMR which for a given input article, scans the collection of articles K times, at each iteration it updates the MR scores of the hole dataset and picks only the $Top1$.

Comparing *dLSH* to *dLSH**

This section reports results of comparing $dLSH$ to $dLSH^*$. The comparison is conducted to measure the impact of precomputing diverse buckets on the diversity and relevance of recommended related articles. As pointed before, LSH algorithm reduces significantly the execution time compared to the exact r_NN algorithm. However, when the size of data collections is important LSH may become slow. In such cases, one can use $dLSH^*$ to reduce the execution time.

Table 3: Execution time($dLSH^*$ vs. $dLSH$)

Tech	Entities	Sentiments	Users	Locations
Gain	27.49x	5.5x	21.34x	22.82x

Table 3 summarizes the gain in execution time obtained by $dLSH^*$ over $dLSH$. For instance, in the case of entity-based diversification, $dLSH^*$ is 27.49 times faster than $dLSH$ whereas in the case of sentiment-based diversification, $dLSH^*$ is only 5.5 times faster. The difference in gain depends on the nature of diversity distance used. From a computational point of view, it is clear that Euclidean distance used in sentiment-based diversification is much faster than Jaccard correlation used in the Entity (UserId and location)-based diversifications. Thus, reducing buckets size in $dLSH^*$ leads to a higher gain in the case of entity, users, and location -based diversification.

Table 4: $dLSH^*$ vs. $dLSH$: diversity/relevance gain

Tech	Entities	Sentiments	Users	Locations
Rel. gain	-1.42%	-9.83%	+3.41%	+4.52%
Div. gain	+58.49%	+11.73%	-19.91%	-17.84%

The main concern of $dLSH^*$ is the way it impacts relevance and diversity compared to $dLSH$. For instance, one would expect that diversity scores of $dLSH^*$ $divK$ sets be lower than diversity scores of $dLSH$ $divK$ sets (because $dLSH^*$ relies on subsets of $dLSH$ buckets). We will see that this observation is not hold when the diversity is measured on articles content. Table 4 presents the overall gain and/or loss in relevance and diversity caused by $dLSH^*$ when compared to $dLSH$. These overall scores are obtained by averaging scores of each diversification technique obtained at different values of $k = \{3, 5, 10, 15, 20\}$ and $r = \{0.5, 0.6, 0.7, 0.8\}$. It is worth to notice that diversity gains presented in the figure are calculated over *content*-based diversity achieved by $dLSH$ and $dLSH^*$. As we can see, $dLSH^*$ did well in the case of entity-based diversification as the diversity of returned sets increases by 58.50% while the relevance decreases

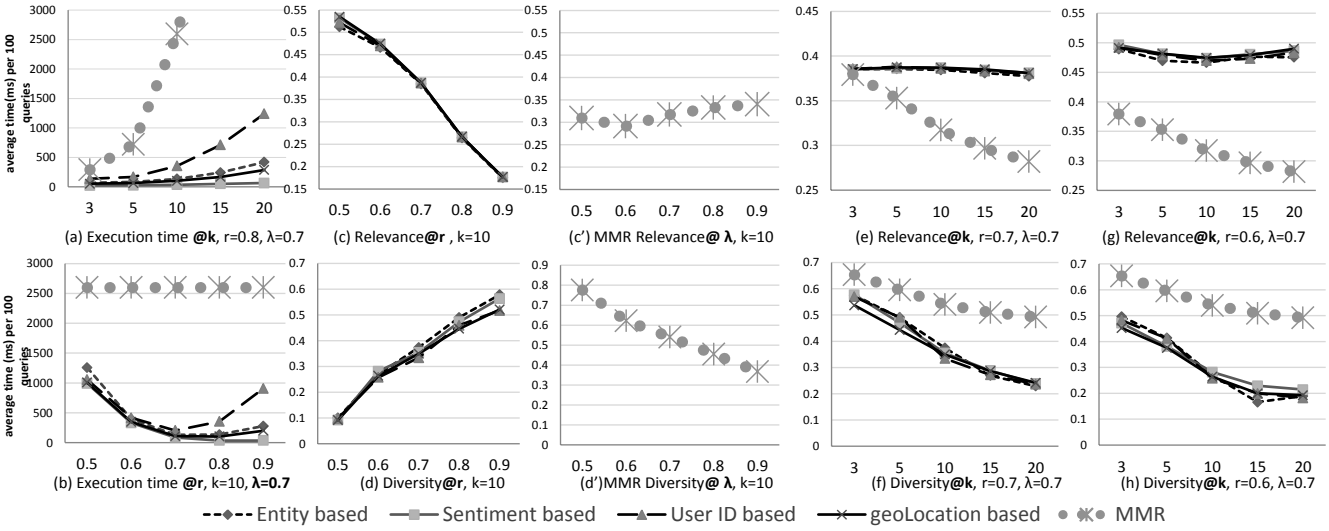

Figure 6: *Comparing dLSH to MMR in terms of execution time, relevance, and diversity*

by only 1.42% compared to sets returned by $dLSH$. That is, there's a real benefit to use $dLSH^*$ rather than $dLSH$ for this kind of diversification. In the case of sentiment-based diversification, the results are encouraging although the advantage of using $dLSH^*$ is less noticeable as the resulting 11.7% gain in diversity is almost paid with a loss of 9.8% in terms of relevance. In contrast to that, we observe in the two remaining diversification techniques, namely *user*-based and *location*-based, that $dLSH^*$ leads to a diversity decrease and a relevance increase. Moreover, the derived loss in diversity (user-based: -19.9%, location-based:-17.8%) is much more important than the gain in relevance obtained ((user-based: 3.4%, location-based: 4.5%)).This unexpected results lead to think that there is no obvious relationship between $dLSH^*$ and $dLSH$ for two reasons: (i.) The diversity is maximized for some comment-based distance its quality is measured in terms of content. (ii.) The 6-approximation doesn't hold in the case of Jaccard-based distances which doesn't satisfy the triangular inequality.

4.5 User study

To evaluate the quality of our diversification techniques, we conducted a user study using Amazon Mechanical Turk system (AMT). The goal of this survey is twofold: (i.) Compare lists of related articles generated by $dLSH$ and MMR. (ii.) Compare different diversity distances to each other. For this purpose, we randomly picked two articles from two main topics: Politics and Business-Finance. For each input article we generated four related articles lists each of size 5. Three lists are generated using $dLSH$ with *entity*-, *sentiment*-, and *userId*- diversity distance whereas the fourth list is generated using MMR. *location*-based diversification is intentionally omitted because it usually generates the same lists than *userID*-based diversification.

Comparative experiment.

We design a survey including two questions of interest: In question Q_1, we ask workers to rate the overall diversity of one list of articles (let say list A) wrt. another list (say list B) using five comparative measures: articles in list A are *significantly more diverse*, *somewhat more diverse*, *equally diverse*, *somewhat less diverse*, or *significantly less diverse* compared to articles in list B. In question Q_2, we simply ask them to say which list of articles they do prefer.

We build for each input article 6 different groups of HITs each of which composed of 11 identical HITs (the total number of HITs is 132). In the first three groups we compare diversified list of articles (say, $divK$ lists) generated by our diversification techniques (entity-based, sentiment-based, and userID-based) to the list generated by MMR (say, $mmrK$ lists.) The three remaining groups are devoted to compare $divK$ lists two by two. In order to make the user study realistic, we omitted to provide any extra information about related articles but their titles while the input article was presented with its title, key features (topic, persons, locations), a small abstract, and a *Read more* link. As a consequence, all workers answers will rely on the titles only.

Results To quantify the responses given by workers, we use two metrics called Mean Response Volume (MRV) and Mean Weighted Response (MWR) [9]. For a given question, MRV measures the number of responses received per option divided by the total number of responses whereas MWR provides an aggregated quantitative measure about workers responses. To aggregate workers responses, we number options from 1 (worst) to 4 or 5 (best) according to the number of options provided.

Overall, the user study revealed that only 28.5% of workers preferred (in $Q2$) the list of related articles they identified as being more diverse than the other one (in $Q1$). This leads to think that users are somehow reluctant to diversity as they don't understand the reason for which certain articles are recommended to them. An explanation effort should be undertaken here to overcome this problem (e.g., display the related features under article titles).

Figure 7-(a) shows the overall MRV of each option in response to Q1 in HITs comparing $divK$ lists to $mmrK$ lists. In the case of *economics* input article, more than 65% of workers found $dLSH$ related articles (significantly or somewhat) more diverse than MMR list while in the case of *politics* input article only 30% of workers said that $dLSH$

generated lists were more diverse than MMR list. These contradictory results are mainly due to the fact that users are restricted to article titles to judge the diversity of lists. It turns that in the former case, all the five articles in the MMR list contained the word *China* in their title while the input article was about *The asset crisis of emerging economies*. $dLSH$ did better in this case, as the list of articles it generated contained more varied titles. In the latter case, we think that the actual (content) diversity of articles was better reflected in the titles in both MMR and $dLSH$ lists. Therefore, a majority of workers noticed that MMR list is more diverse than $dLSH$ lists.

Figure 7-(B) reports the MWR breakdown by diversification technique. It shows that the three diversification techniques produce lists that are more diverse ($score \succ 3$) compared to $mmrK$ lists in the case of *economics* input article. In the case of *politics* input article, userID and entity-based diversification got almost the same diversity scores than MMR while sentiment-based lists were definitely less diverse.

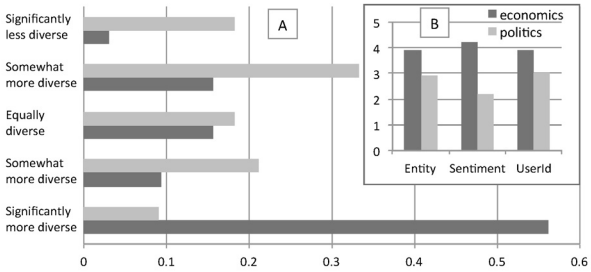

Figure 7: *Diversity of dLSH vs. MMR*

In terms of pairwise comparison of $dLSH$ diversification techniques, results are as follows: In the case of *economics* article sentiment-based diversification was preferred to both userId-based (55% vs. 45% of votes) and entity-based (73% vs.27% of votes), and user-based diversification was preferred to the entity-based (82% vs. 18% of votes). In the case of *politics* article, entity-based diversification did almost as well as sentiment-based diversification, both ahead userID-based technique. Overall, the most preferred diversification was sentiment-based distance which surprisingly got a good results in both *economics* and *politics* input articles. That is, people tend more than ever to express their sentiment on news articles of different topics, in particular those directly impacting their lives.

5. RELATED WORK

Search result diversification has received extensive attention in both research and industry. Carbonell and Goldstein introduced the concept of diversity in text retrieval and summarization [8]. Authors used two similarities to define the Maximal Marginal Relevance criterion. The first similarity is used to evaluate document relevance given a query and the second to estimate document novelty. Since then, several diversification techniques have been proposed which fall into two main approaches: content-based and intent-based. In the former [8, 12, 6] the goal is to reduce the redundancy of a set of results by selecting the most relevant document and greedily adding documents which maximize relevance and minimize redundancy. In the latter [3, 14] the goal is to cover subtopics (intents) behind queries. This approach is particularly effective in the case of ambiguous queries. It starts by identifying the set of all possible query subtopics and uses probabilities with which the query belongs to each subtopic to rank documents.

In recommender systems, diversification was approached through two paradigms: local and aggregate. Local diversity [18, 17, 15, 16] aims at increasing the intra-diversity of a set of items within a recommendation list and is usually achieved using attributes [18, 17] or explanations [16]. Unlike local diversity, aggregate diversity aims at increasing diversity [2] so that users do not see the same top k items.

Jain et al. address the problem of providing diversity in K-nearest neighbor queries [12]. Authors present an algorithm (MOTLEY) which relies on a user-tunable parameter to control the minimun diversity that any pair of results should satisfy. The main idea consists in finding the set of candidate answers which satisfies the diversity constraint, then access these candidates in a smart way (using an R-tree) to pick the set of k nearest neighbors. Besides the fact that we are using social data to recommend diverse news, there are some major differences between our work and that of Jain et al. *i.* In our solution, only those items which are within a relevance distance are candidates, whereas [12] consider items which satisfy a diversity constraint, i.e., MOTLEY provides no guarantee on relevance. *ii.* Unlike MOTLEY that uses R-trees to retrieve the set of k-NN, we are using LSH techniques [5] that are faster and more robust under high dimensionality [11].

6. CONCLUSION AND FUTURE WORK

We addressed the problem of providing diverse news recommendations related to an input article. We leveraged user-generated data to refine lists of related articles. In particular, we explored different diversity distances that rely on the content of user comments on articles such as sentiments and entities. Given the real-time nature of our recommendations, we applied locality-sensitive hashing techniques to find candidate articles that are within a relevance distance r from an input article a. Then, we adapted a greedy algorithm to identify the k articles that collectively 2-approximately maximize diversity. Our user study showed that user-generated data achieves a good balance between relevance and diversity compared to other diversification techniques. We also showed with an extensive set of experiments on Al Jazeera English and Reuters collections, that using LSH makes our proposals effective in an real-time environment.

We would like to investigate the relationship between LSH parameters L and K and the relevance of nearest neighbor articles retrieved. The intuition we have is that the diversity of two articles can be a function of the number of times ($\alpha = 1 \ldots L$) they fall into the same bucket (i.e. number of hash functions that hash both articles to the same bucket). The lower that number, the higher the diversity.

7. REFERENCES

[1] S. Abbar, S. Amer-Yahia, P. Indyk, S. Mahabadi, and K. Varadajaran. K-diverse near neighbor problem, to be submitted.

[2] G. Adomavicius and Y. Kwon. Improving aggregate recommendation diversity using ranking-based techniques. *Knowledge and Data Engineering, IEEE Transactions on*, PP(99):1, 2011.

[3] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, WSDM '09, pages 5–14, New York, NY, USA, 2009. ACM.

[4] S. Amer-Yahia, S. Anjum, A. Ghenai, A. Siddique, S. Abbar, S. Madden, A. Marcus, and M. El-Haddad. Maqsa: a system for social analytics on news. In *SIGMOD Conference*, pages 653–656, 2012.

[5] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 51(1):117–122, Jan. 2008.

[6] A. Angel and N. Koudas. Efficient diversity-aware search. In *Proceedings of the 2011 international conference on Management of data*, SIGMOD '11, pages 781–792, New York, NY, USA, 2011. ACM.

[7] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. Min-wise independent permutations (extended abstract). In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, STOC '98, pages 327–336, New York, NY, USA, 1998. ACM.

[8] J. Carbonell and J. Goldstein. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '98, pages 335–336, New York, NY, USA, 1998. ACM.

[9] M. D. Choudhury, M. Feldman, S. Amer-Yahia, N. Golbandi, R. Lempel, and C. Yu. Automatic construction of travel itineraries using social breadcrumbs. In *HT*, pages 35–44, 2010.

[10] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.*, 38:293–306, 1985.

[11] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, STOC '98, pages 604–613, New York, NY, USA, 1998. ACM.

[12] A. Jain, P. Sarda, and J. R. Haritsa. Providing diversity in k-nearest neighbor query results. In *PAKDD*, pages 404–413, 2004.

[13] S. Ravi, D. Rosenkrantz, and G. Tayi. Facility dispersion problems: Heuristics and special cases. *Algorithms and Data Structures*, pages 355–366, 1991.

[14] M. J. Welch, J. Cho, and C. Olston. Search result diversity for informational queries. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 237–246, New York, NY, USA, 2011. ACM.

[15] C. Yu, L. Lakshmanan, and S. Amer-Yahia. It takes variety to make a world: diversification in recommender systems. In *Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology*, EDBT '09, pages 368–378, New York, NY, USA, 2009. ACM.

[16] C. Yu, L. V. S. Lakshmanan, and S. Amer-Yahia. Recommendation diversification using explanations. In *ICDE*, pages 1299–1302, 2009.

[17] M. Zhang and N. Hurley. Avoiding monotony: improving the diversity of recommendation lists. In *Proceedings of the 2008 ACM conference on Recommender systems*, RecSys '08, pages 123–130, New York, NY, USA, 2008. ACM.

[18] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, WWW '05, pages 22–32, New York, NY, USA, 2005. ACM.