Multi-document summarization exploiting frequent itemsets

Elena Baralis
Politecnico di Torino
Corso Duca degli Abruzzi
Torino, Italy
elena.baralis@polito.it

Luca Cagliero
Politecnico di Torino
Corso Duca degli Abruzzi
Torino, Italy
luca.cagliero@polito.it

Alessandro Fiori
Politecnico di Torino
Corso Duca degli Abruzzi
Torino, Italy
alessandro.fiori@polito.it

Saima Jabeen
Politecnico di Torino
Corso Duca degli Abruzzi
Torino, Italy
saima.jabeen@polito.it

ABSTRACT

A summary is a succinct and informative description of a data collection. In the context of multi-document summarization, the selection of the most relevant and not redundant sentences belonging to a collection of textual documents is definitely a challenging task. Frequent itemset mining is a well-established data mining technique to discover correlations among data. Although it has been widely used in transactional data analysis, to the best of our knowledge, its exploitation in document summarization has never been investigated so far.

This paper presents a novel multi-document summarizer, namely ItemSum (Itemset-based Summarizer), that is based on an itemset-based model, i.e., a model composed of frequent itemsets, extracted from the document collection. It automatically selects the most representative and not redundant sentences to include in the summary by considering both sentence coverage, with respect to a concise and highly informative itemset-based model, and a sentence relevance score, based on tf-idf statistics. Experimental results, performed on the DUC'04 document collection by means of ROUGE toolkit, show that the proposed approach achieves better performance than a large set of competitors.

Categories and Subject Descriptors

I.5.4 [**Pattern Recognitions**]: ApplicationsText Processing; H.2.8 [**Information Systems**]: Database Management-Database ApplicationsData Mining

General Terms

Algorithms

SAC'12 March 25-29, 2012, Riva del Garda, Italy.

Keywords

Multi-document summarization, Text mining, Frequent itemset mining

1. INTRODUCTION

In last years, the increasing availability of textual documents in the electronic form has prompted the need of efficient and effective data mining approaches suitable for textual data analysis. Summarization is a challenging data mining task that focuses on constructing a succinct and informative description of a data collection. In the context of multi-document summarization, a summary is composed of the most representative sentences belonging to a document collection.

A number of different approaches have been proposed to select the most relevant sentences (e.g., [11, 14, 17]). They commonly evaluate sentences according to cluster-based or graph-based models. For instance, the approach recently proposed in [17] exploits an incremental hierarchical clustering algorithm with the two-fold aim at identifying groups of sentences that share the same content and updating summaries over time. Differently, [11] proposed to represent correlations among sentences by means of a graph-based model. Most relevant sentences are selected according to the eigenvector centrality computed by means of the well-known PageRank algorithm [4]. A parallel research effort has been devoted to formalizing the summarization task as a maximum coverage problem with Knapsack constraints based on sentence relevance within each document [14]. However, previous approaches typically focus on single word significance while do not effectively capture correlations among multiple words at the same time.

Frequent itemset mining [1] is a widely exploratory technique to discover hidden correlations that frequently occur in the source data. Although its application to transactional data is well-established, to the best of our knowledge, the usage of frequent itemsets in textual document summarization has never been investigated so far. In recent years, a number of approaches addressed the discovery and selection of the most informative yet non-redundant set of frequent itemsets mined from transactional data (e.g., [7, 15]). Some of them compared the observed frequency (i.e., the support) of each

itemset against some null hypotheses (e.g., its expected frequency) to evaluate its interestingness. Others considered previously selected patterns in itemset evaluation to reduce model redundancy. Among them, [15] effectively exploited an heuristics to solve the maximum entropy model that allows evaluating on-the-fly the significance of an itemset during the itemset mining process.

This paper presents the ItemSum (Itemset-based Summarizer) multi-document summarizer. ItemSum provides two main contributions: (i) the usage of an itemset-based model to represent the most relevant and not redundant correlations among document terms, and (ii) the selection of the minimal set of representative sentences that best covers the itemset-based model. From a transactional representation of the document collection, an highly informative and not redundant itemset-based model is extracted to represent significant higher order correlations among document terms. To address this issue, the algorithm first proposed in [10] in the context of transactional data is adopted. Since it pushes the itemset selection into the mining process, it is particularly suitable for being applied in text summarization. To better discriminate among single word occurrences within each document, ItemSum combines the usage of the itemset-based model with a sentence relevance score, computed from the bag-of-word sentence representation and based on the well-founded tf-idf statistics [8]. The problem of selecting the minimal set of sentences that best covers the itemset-based model is formalized as a set covering problem. To solve the problem efficiently and effectively, a greedy approach is adopted. We validated our approach against a large number of approaches on the DUC'04 [6] document collection. Performance comparisons, in terms of precision, recall, and F-measure, have been performed by means of the ROUGE [9] toolkit. In most cases, ItemSum significantly outperforms the considered competitors. Furthermore, the impact of both the main algorithm parameters and the adopted model coverage strategy on the summarization performance are investigated as well.

The paper is organized as follows. Section 2 describes the main steps of the ItemSum method. Section 3 presents the experiments we performed to validate our approach, while Section 4 draws conclusions and presents future developments.

2. THE SUMMARIZATION METHOD

ItemSum is a novel summarizer that selects the most representative sentences based on both their coverage of an itemset-based model and their single-word statistical relevance. To this aim, it relies on a two-way data representation, which is formally stated in the following section.

2.1 Document representation

Two different document/sentence representations are exploited by ItemSum: (i) the traditional bag-of-word (BOW) sentence representation to evaluate the sentence relevance score and (ii) the transactional data format to compute the itemset-based model. The raw document content is first preprocessed to make it suitable for the data mining and knowledge discovery process. To avoid noisy information stopwords, numbers, and website URLs are removed, while the Wordnet stemming algorithm [3] is applied to reduce document words to their base or root form (i.e., the stem). Let $D=\{d_1,\ldots,d_n\}$ be a document collection, where each document d_k is composed of a set sentences $S_k=\{s_{1k},\ldots,s_{zk}\}$. Documents are composed of a sequence of sentences, each one composed of a set of words. The BOW representation of the j-th sentence s_{jk} belonging to the k-th document d_k of the collection D is the set of all word stems (i.e., terms) occurring in s_{jk}.

Consider now the set $tr_{jk}=\{w_1,\ldots,w_l\}$ where $tr_{jk}\subseteq s_{jk}$ and $w_q \neq w_r \ \forall \ q \neq r$. It includes the subset of distinct terms occurring in the sentence s_{jk}. To tailor document sentences to the transactional data format, we consider each document sentence as a transaction whose items are distinct terms taken from its BOW representation, i.e., tr_{jk} is the transaction that corresponds to the document sentence s_{jk}. A transactional representation T of the document collection D is the union of all transactions tr_{jk} corresponding to each sentence s_{jk} belonging to any document $d_k \in D$.

The document collection is associated with the statistical measure of the term frequency-inverse document frequency (tf-idf) that evaluates the relevance of a single word in the whole collection. A more detailed description of the tf-idf statistics is reported in [8].

2.2 Itemset-based model generation

Frequent itemset mining [1] is a widely exploratory data mining technique that focuses on discovering correlations, i.e., itemsets, that frequently occur in the source data. An itemset I of length k, i.e., a k-itemset, is a set of k distinct items. Let T be the document collection in the transactional data format. We denote as $\mathcal{D}(I)$ the set of transactions supported by I, i.e., $\mathcal{D}(I)=\{tr_{jk}\in T \mid I\subseteq tr_{jk}\}$. The support of an itemset I is the observed frequency of occurrence of I in D, i.e., $sup(I)=\frac{\mathcal{D}(I)}{|T|}$. Since the problem of discovering all itemsets from a transactional dataset is computationally intractable [1], itemset mining is commonly driven by a minimum support threshold.

Given a minimum support threshold min_sup and a model size ms, ItemSum generates an itemset-based model that includes the most informative yet non-redundant set of ms frequent itemsets discovered from the document collection T. Among the large set of previously proposed approaches focused on succinctly representing transactional data by means of itemsets [15], we adopt an algorithm recently proposed in [10]. Unlike previous approaches, it exploits an entropy-based heuristics to drive the mining process and select the most informative yet not redundant itemsets without the need of a postpruning step. Its efficiency and effectiveness in discovering succinct transactional data summaries makes it particularly suitable for the application to text summarization.

2.3 Sentence evaluation and selection

ItemSum exploits the itemset-based model to evaluate and select most relevant sentences to include in the summary. Sentence evaluation and selection steps consider (i) a sentence relevance score that combines the tf-idf statistics [8] associated with each sentence term, and (ii) the sentence coverage of the generated itemset-based model (see Section 2.2). In the following we formalize both sentence relevance score and sentence model coverage.

Sentence relevance score.

The relevance score of a sentence is computed from the BOW sentence representation. It is defined as the sum of the

tf-idf values [8] of each distinct term belonging to a generic sentence s_{jk} in the document collection.

$$SR(s_{jk}) = \frac{\sum_{i \mid w_i \in t_{jk}} tc_{ik}}{|t_{jk}|} \quad (1)$$

where t_{jk} is the set of distinct terms occurring in s_{jk}, and $\sum_{i \mid w_i \in s_{jk}} tc_{ik}$ is the sum of the tf-idf values associated with terms (i.e., word stems) in s_{jk}.

Sentence model coverage.

The sentence coverage measures the pertinence of each sentence to the generated itemset-based model. To this aim, it considers document sentences tailored to the transactional data format. We first associate with each sentence $s_{jk} \in D$ a binary vector, denoted in the following as *sentence coverage vector*, SC_{jk}=$\{sc_1, \ldots, sc_{ms}\}$ where ms is the number of itemsets belonging to the model and $sc_i = \mathbf{1}_{tr_{jk}}(I_i)$ indicates whether itemset I_i supports or not tr_{jk} (see Section 2.2). More formally, $\mathbf{1}_{tr_{jk}}$ is an indicator function defined as follows:

$$\mathbf{1}_{tr_{jk}}(I_i) = \begin{cases} 1 & \text{if } I_i \subseteq tr_{jk}, \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

The coverage of a sentence s_{jk} with respect to the pattern-based model is defined as the number of ones that occur in the corresponding coverage vector SC_{jk}.

We formalize the problem of selecting the most informative and not redundant sentences according to the model as a set covering problem.

The set covering problem.

The set covering optimization problem focuses on selecting the minimal set of sentences, of arbitrary size l and maximal score, whose logic OR of the corresponding coverage vectors, i.e., SC^*=$SC_1 \vee \ldots \vee SC_l$, generates a binary vector with the maximum number of 1's. This implies that each itemset belonging to the model covers at least one sentence. The SC^* vector will be denoted as the *summary coverage vector* in the rest of this section. ItemSum addresses the set covering problem to select sentences with the best model coverage and relevance score.

The set covering optimization problem is known to be NP-hard. To solve the problem, we adopt a greedy strategy similar to that successfully applied in [2] in the context of biological data feature selection. The greedy sentence selection strategy considers sentence model coverage as the most discriminative feature, i.e., sentences that cover the maximum number of itemsets belonging to the model are selected first. At equal terms, the sentence with maximal coverage that is characterized by the highest relevance score SR is preferred. A more detailed description of the adopted greedy strategy follows.

The greedy strategy.

The adopted algorithm identifies, at each step, the sentence s_{jk} with the best complementary vector SC_{jk} with respect to the current summary coverage vector SC^*. The pseudo-code of the greedy approach is reported in Algorithm 1. It takes in input the set of sentence relevance scores SR, the set of sentence coverage vectors SC, and the tf-idf matrix TC. It produces the summary $\mathcal{S}$, i.e., the minimal

Algorithm 1 Greedy sentence selection

Input: set of sentence relevance scores SR, set of sentence coverage vectors SC, tf-idf matrix TC
Output: summary $\mathcal{S}$

```
1: S = ∅
2: ESC = ∅ {set of eligible sentence coverage vectors}
3: SC* = all_zeros() {summary coverage vector with only 0s}
4: {Cycle until either SC* contains only 1s or all the SC vectors
   contain only zeros}
5: while not (all_ones(SC*) or only_zeros(SC*)) do
6:    {Determine the sentences with the highest number of ones}
7:    ESC = max_ones_sentences()
8:    if ESC != ∅ then
9:       {Select the sentence with maximum relevance score}
10:      SC_best = ESC[best] with best = arg_i max SR_i
11:      {Update sets and summary_coverage_vector}
12:      S = S ∪ SC_best
13:      SC*= SC* OR SC_best
14:      ESC = ESC \ SC_best
15:      {Update the sentence coverage vectors belonging to V}
16:      for all SC_i in SC do
17:         SC_i = SC_i AND ¬SC*
18:      end for
19:   else
20:      break
21:   end if
22: end while
23: return S
```

subset of the most representative sentences. The first step is the variable initialization and the sentence coverage vector computation (lines 1-3). Next, the sentence with maximum coverage, i.e., the one whose coverage vector contains the maximum number of ones, is iteratively selected (line 6). At equal terms, the sentence with maximum relevance score (Cf. Formula 1) is preferred (line 10). Finally, the selected sentence is included in the summary $\mathcal{S}$ while the summary and sentence coverage vectors are updated (lines 12-18). The procedure iterates until either the summary coverage vector contains only ones, i.e., the model is fully covered by the summary, or the remaining sentences are not covered by any itemset, i.e., the remaining sentences are not pertinent to the model (line 5).

Experimental results, reported in Section 3.2.3, show that the proposed sentence selection algorithm is more effective and efficient than a branch-and-bound algorithm for text summarization purposes.

3. EXPERIMENTAL RESULTS

We performed a set of experiments to address the following issues: (i) the performance comparison between ItemSum and other text summarizers (Section 3.1), and (ii) the impact of model size, support threshold, and set covering algorithm on the performance of ItemSum (Section 3.2)

We evaluated our method on task 2 of DUC'04 [6], which is the latest DUC dataset on generic text summarization [17]. The provided collection is composed of 50 document groups, each of them including 10 documents. For each group, a golden summary is given. We compared our approach with 30 methods submitted to the DUC conference and two other widely used text summarizers, i.e., the Open Text Summarizer (OTS) [13] and TexLexAn [16]. To compare ItemSum with the other approaches we used the ROUGE [9] toolkit, which has been adopted as official DUC'04 tool for performance evaluation[1]. It measures the quality of a summary

[1]The provided command is: ROUGE-1.5.5.pl -e data -x -m -2 4 -u -c 95 -r 1000 -n 4 -f A -p 0.5 -t 0 -d -a

by counting the unit overlaps between the candidate summary and a set of reference summaries. Intuitively, the summarizer that achieves the highest ROUGE scores could be considered as the most effective one. To perform a fair comparison the golden and generated summaries have been preliminary normalized before using the ROUGE tool.

3.1 Performance comparison and validation

We evaluated the performance of ItemSum in terms of ROUGE-2, ROUGE-3, and ROUGE-4 precision (Pr), recall (R), and F-measure (F). For both OTS and TexLexAn we adopted the configuration suggested by the respective authors. For the DUC competitors the results provided by the DUC'04 system [6] are considered. For ItemSum we set, as standard configuration, the minimum support threshold min_sup=3% and the model size ms=12. A more detailed discussion on the impact of both min_sup and ms on the performance of ItemSum is reported in Sections 3.2.1 and 3.2.2. Table 1 summarizes the achieved results. For the lack of space, among the DUC'04 competitors, we reported just three representative ones, i.e., the two that achieve the best performance in terms of ROUGE-4 F-measure and one that achieves medium performance. ItemSum performs better than OTS, TexLexAn, and all the other considered summarizers for any tested measure, except for ROUGE-2 Precision. To validate the statistical significance of the performance improvements, we used the paired t-test [5] at 95% significance level for each evaluated dataset and measure. ItemSum significantly outperforms all the considered approaches in terms of ROUGE-4 F-measure (25 significant improvements out of 30 DUC'04 competitors) and 4 out 5 in terms of ROUGE-3 F-measure (24 out of 30). Although one of the DUC'04 competitors slightly outperforms ItemSum in terms of ROUGE-2 Precision, the performance worsening is not significant. Furthermore, ItemSum significantly outperforms OTS, TexLexAn, and 24 out of 30 DUC'04 competitors in terms of ROUGE-2 F-measure.

3.2 Parameter analysis

We analyzed the impact of the minimum support threshold and the pattern-based model size, i.e., the number of selected itemsets, on the performance of ItemSum. To also evaluate the impact of the greedy algorithm on the summarization performance, we tested a slightly modified version of ItemSum, which adopts a branch-and-bound algorithm [12] to solve the set covering problem, as well. Furthermore, to perform a fair comparison the golden and generated summaries have been preliminary normalized before using the ROUGE tool. For the lack of space, we only reported, in Figures 1(a) and 1(b), the achieved ROUGE-3 F-measure values. Analogous results have been obtained for the other ROUGE scores, for precision and recall measures, and for all other configurations. In the following, we discuss the impact of each considered factor separately.

3.2.1 Impact of the support threshold

When higher support thresholds (e.g., 10%) are enforced, many informative itemsets are discarded, thus the itemset-based model becomes too general to yield high summarization performance. Oppositely, when very low support thresholds (e.g., 0.1%) are enforced, data overfitting occurs, i.e., the model is too much specialized to effectively and concisely summarize the whole document collection content. At medium support thresholds (e.g., 3%) the best balancing between model specialization and generalization is achieved, thus, ItemSum produces succinct yet informative summaries.

3.2.2 Impact of the model size

The model size may significantly affect the summarization performance. When a limited number of itemsets (e.g., ms=5) is selected, the relevant knowledge hidden in the document collection is not yet fully covered by the extracted patterns (see Figure 1(b)), thus the generated summaries are not highly informative. The multi-document itemset-based summarizer becomes very effective when the model size is around 12 as selected patterns are representative of the most informative and non-redundant knowledge. Differently, when the model size further increases the quality of the generated summaries worsens as the model is still informative but redundant. The best value of model size depends on the analyzed document term distribution.

3.2.3 Impact of the set covering algorithm

ItemSum exploits a greedy algorithm to solve the set covering optimization problem. Generally speaking, it produces an approximated solution to the problem of selecting the set of sentences that best covers the itemset-based model. Since the set covering problem is a min-max problem, it may be converted to a linear programming problem and addressed by using combinatorial optimization strategies. Thus, we compared the performance achieved by ItemSum with that achieved by a slightly modified version, namely ItemSum$_{B\&B}$, which exploits a branch-and-bound implementation [12] to address the set covering task. In most cases ItemSum outperforms ItemSum$_{B\&B}$ and shows a more stable trend in terms of ROUGE scores. In particular, when high-quality models are generated (e.g., when $p = 12$ and $min\_sup = 3\%$) the best results are achieved by the greedy approach. ItemSum$_{B\&B}$ yields better results when the itemset-based model is low-quality or redundant (e.g., when $p < 6$ or $min\_sup > 5\%$). However, it does not provide any significant performance improvement with respect to ItemSum, while it requires a higher execution time (e.g., around 5% more when $p = 12$ and $min\_sup = 3\%$).

4. CONCLUSIONS AND FUTURE WORKS

In this paper we present a multi-document summarizer that combines the knowledge provided by an itemset-based model with a statistical evaluator, based on tf-idf statistics, to select the most representative and not redundant sentences. To the best of our knowledge, this is the first attempt to exploit frequent itemsets in text summarization. Experiments, conducted on DUC'04 document collection, show the effectiveness of the proposed approach.

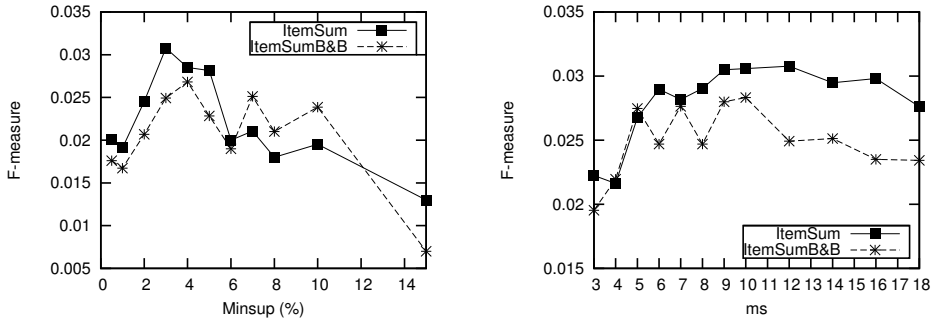
Future works will address: (i) the extension of the proposed approach to address the problem of incremental summary updating, and (ii) the application of entity disambiguation techniques to improve the summarization performance.

5. REFERENCES

[1] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large

Summarizer		ROUGE-2			ROUGE-3			ROUGE-4		
		R	Pr	F	R	Pr	F	R	Pr	F
OTS		0.0746*	0.0740*	0.0743*	0.0236*	0.0234*	0.0235*	0.0087*	0.0086*	0.0087*
TexLexAn		0.0655*	0.0643*	0.0649*	0.0197*	0.0193*	0.0195*	0.0071*	0.0069*	0.0070*
DUC'04 competitors	peer102	0.0840	0.0846	0.0843	0.0264	0.0267	0.0265	0.0103*	0.0104*	0.0104*
	peer103	0.0774*	**0.0896**	0.0826	0.0243*	0.0284	0.0260*	0.0092*	0.0108	0.0099*
	peer140	0.0685*	0.0692*	0.0688	0.0218*	0.0220*	0.0219*	0.0093*	0.0094*	0.0093*
ItemSum		**0.0864**	0.0869	**0.0866**	**0.0307**	**0.0309**	**0.0308**	**0.0135**	**0.0136**	**0.0135**

Table 1: Performance comparison in terms of ROUGE-2, ROUGE-3 and ROUGE-4 scores. Statistically relevant worsening in the comparisons between ItemSum and the other approaches are starred.



(a) ms=12. Impact of the support threshold.

(b) min_sup=3%. Impact of the model size.

Figure 1: Comparison between ItemSum and ItemSum$_{B\&B}$ on Rouge-3 with different parameter settings.

databases. In *ACM SIGMOD Record*, volume 22, pages 207–216.

[2] E. Baralis, G. Bruno, and A. Fiori. Minimum number of genes for microarray feature selection. *30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 5692–5695, 2008.

[3] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python*. O'Reilly Media, 2009.

[4] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the seventh international conference on World Wide Web 7*, pages 107–117, 1998.

[5] T. G. Dietterich. Approximate statistical test for comparing supervised classification learning algorithms. *Neural Computation*, 10(7), 1998.

[6] Document Understanding Conference. HTL/NAACL workshop on text summarization, 2004. `http://www-nlpir.nist.gov/projects/duc/pubs.html`.

[7] S. Jaroszewicz and D. A. Simovici. Interestingness of frequent itemsets using bayesian networks as background knowledge. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 178–186, 2004.

[8] K. S. Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21, 1972.

[9] C.-Y. Lin and E. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 71–78, 2003.

[10] M. Mampaey, N. Tatti, and J. Vreeken. Tell me what I need to know: Succinctly summarizing data with itemsets. In *Proceedings of the 17th ACM Conference on Knowledge Discovery and Data Mining*, 2011.

[11] D. R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22:2004, 2004.

[12] T. Ralphs and M. Guzelsoy. The SYMPHONY callable library for mixed integer programming. *The Next Wave in Computing, Optimization, and Decision Technologies*, 29:61–76, 2006. Software available at `http://www.coin-or.org/SYMPHONY`.

[13] N. Rotem. Open text summarizer (ots). *Retrieved July*, 2006.

[14] H. Takamura and M. Okumura. Text summarization model based on the budgeted median problem. In *Proceeding of the 18th ACM conference on Information and knowledge management*, pages 1589–1592, 2009.

[15] N. Tatti. Probably the best itemsets. In *Proceedings of the 16th ACM international conference on Knowledge discovery and data mining*, pages 293–302, 2010.

[16] TexLexAn. Texlexan: An open-source text summarizer, 2011.

[17] D. Wang and T. Li. Document update summarization using incremental hierarchical clustering. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 279–288, 2010.