Learning Temporal-Dependent Ranking Models

Miguel Costa [1,2]
miguel.costa@fccn.pt

Francisco M Couto [2]
fcouto@di.fc.ul.pt

Mário J. Silva [3]
mjs@inesc-id.pt

[1] Foundation for National Scientific Computing, Portugal
[2] Departamento de Informática, Faculdade de Ciências, Universidade de Lisboa, Portugal
[3] INESC-ID, Instituto Superior Técnico, Universidade de Lisboa, Portugal

ABSTRACT

Web archives already hold together more than 534 billion files and this number continues to grow as new initiatives arise. Searching on all versions of these files acquired throughout time is challenging, since users expect as fast and precise answers from web archives as the ones provided by current web search engines. This work studies, for the first time, how to improve the search effectiveness of web archives, including the creation of novel temporal features that explore the correlation found between web document persistence and relevance. The persistence was analyzed over 14 years of web snapshots. Additionally, we propose a temporal-dependent ranking framework that exploits the variance of web characteristics over time influencing ranking models. Based on the assumption that closer periods are more likely to hold similar web characteristics, our framework learns multiple models simultaneously, each tuned for a specific period. Experimental results show significant improvements over the search effectiveness of single-models that learn from all data independently of its time. Thus, our approach represents an important step forward on the state-of-the-art IR technology usually employed in web archives.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Retrieval models

General Terms

Algorithms; Performance; Experimentation

Keywords

web archives; temporal-dependent ranking

1. INTRODUCTION

Web archive information retrieval (WAIR) addresses the retrieval of document versions from web archives, according to topical and temporal criteria of relevance. WAIR differs from typical IR and web IR, because a web archive corpus is distinctively composed by a stack of content collections harvested from the web over time. Thus, each document may have several versions and the relevance of a version depends on the user's period of interest. Another main difference of WAIR is that its multi-version web collections have different characteristics over time, which causes variations in the discriminative power of features used in ranking.

WAIR has been applied in at least 68 web archive initiatives[1] undertaken by national libraries, national archives and consortia of organizations that are acquiring and preserving parts of the web. Web archives already hold more than 534 billion files (17 PB), of which some are historical records, such as opinions, decisions and photos of events. Pieced together, these records form our collective memory and the possibility of looking into the past opens space for novel applications. In the last years, applications based on data from web archives include tools for assessing the trustworthiness of statements [31], detecting web spam [5], improving web IR [10] or forecasting events [26]. However, despite web archive technology having achieved a good maturity level, the effectiveness of the search services they provide still presents unsatisfactory results [8]. As result, information cannot be found and web archives are useless for their users.

In this work, we cope with the poor retrieval effectiveness of web archives by addressing three identified limitations. First, the ranking relevance of document versions is currently computed based only on the similarity of each content with the query, ignoring many other features which have shown to improve web search engines. By employing state-of-the-art learning to rank (L2R) algorithms on such features we immediately obtained significant improvements, increasing more than three times the search effectiveness of state-of-the-art WAIR. Second, web archives preserve many years of collected web snapshots, but current WAIR approaches ignore the time dimension in such collections. We researched what relevant information to WAIR can be extracted from this time dimension, by exploring, for the first time, the long-term persistence of web documents. In our experiments, conducted over 14 years of web snapshots, we found that for navigational queries, relevant documents tend to have a longer lifespan and more versions. This result enabled us to obtain significant gains by modeling the persistence of web documents into novel ranking features. These features are especially important in web archives, because the query-independent features typically used to identify popular or important documents based on click-through data and the

SIGIR '14, July 06 - 11 2014, Gold Coast, QLD, Australia

[1] http://en.wikipedia.org/wiki/List_of_Web_archiving_initiatives

web-graph, are not available in this context. Web archives receive a much smaller volume of queries and clicks than web search engines, and the web-graphs are sparser since only a small part of the web is commonly collected and preserved by each archive. Third, the characteristics of the web vary over time. For instance, the sites in the 90s did not have the richer layouts and more interactive interfaces of the early 00s with CSS and JavaScript. Other examples include the dynamics of the web link structure, which grows following a power law [19], and the dynamics of language in web contents, which have many terms appearing and disappearing every year [29]. We believe that a single general ranking model cannot predict the variance of web characteristics over such long periods of time. As a result, we present as the main contribution of this paper, an approach that learns and combines multiple ranking models specific for each period. Experimental results show that our approach outperforms the search effectiveness of single-model approaches that fit all data independently of when it was created or updated. We refer to our approach as temporal-dependent ranking.

The remainder of this paper is organized as follows. In Section 2, we cover the related work. Section 3 analyzes the long-term web document persistence, while a temporal-dependent ranking framework is proposed in Section 4. In Section 5, we present the experimental setup and report the results in Section 6. Section 7 finalizes with the conclusions and future work.

2. RELATED WORK

2.1 Web Archives Access

Much of the current effort on web archive development focuses on acquiring, storing, managing and preserving data [22]. However, the data must also be easily accessible to users who need to exploit and analyze them. Full-text search has become the dominant form of information access, especially in web search systems, such as Google, which has a strong influence on the way users search in other settings. Surveys indicate that full-text search is also the preferred tool for accessing web archive data and the most used when supported [7]. Even with the high computational resources required for this purpose, 67% of world-wide web archives support full-text search for at least a part of their collections [13]. However, the large majority of web archives that support full-text search are based on the Lucene search engine[2] or extensions of Lucene to handle web archives, such as NutchWAX[3]. The search services provided by these web archives are visibly poor and frequently deemed unsatisfactory [13]. An WAIR evaluation confirmed the low quality of search results retrieved with such technology [8]. This last work, like ours, focus in navigational queries, since this is the main information need of web archive users [6].

2.2 Temporal Features

Some works leveraged temporal information to improve full-text search results of web search engines. One of the most common ideas is incorporating in language models the heuristic that the prior probability of a document being relevant is higher in the most recent documents [20]. Boosting the most recent documents is desirable for queries where a user intends to find recent events or breaking news. The distribution of the documents' dates can also be explored, since it reveals time intervals that are likely to be of interest to the query [16]. For instance, when searching for *tsunami*, the peaks in the distribution may indicate when tsunamis occurred. Another idea is to favor more dynamic documents, since documents with higher relevance are more likely to change or change to a greater degree [10]. More popular and revisited documents are also more likely to change [1]. On the other hand, the most persistent terms are descriptive of the main topic and likely added early in the life of a document [1]. These persistent terms are especially useful for matching navigational queries, because the relevance of documents for these queries are expected to not change over time. To the best of our knowledge, we are the first studying the relation between long-term web document persistence and relevance for improving search effectiveness.

2.3 Learning Multiple Models

The L2R framework learns ranking models that fit all training data. However, a generic model is not always the best solution and may be overcome by a criteria-dependent model. For instance, Kang and Kim automatically classified queries and created a ranking model for each query type [17]. However, it is often hard to classify a given web search query due to its small number of terms, which makes this technique unfeasible in some cases or imprecise when the wrong model is chosen. To avoid the misclassification problem, Geng et al. created a ranking model for each query q by using the k-nearest training queries of q measured by the similarity of their feature values [12]. The query feature values were computed as the mean of the feature values of the top search results ranked by a reference model (BM25). However, the training time required to create all these models is quite large and each model is learned with just a part of the training data. Bian et al. employed a clustering method to identify a set of query topics based on features extracted from the top search results [3]. Then, a ranking model was learned for each query topic. Each query contributed to learn each model according to the similarity between the query and the respective topic. Dai et al. followed this work, but integrated freshness with relevance to simultaneously optimize both [9]. In a different work, Salles et al. created a classification model per time interval and weighted them based on the temporal distance between the creation time of documents and the interval [28]. Our work is the first that learns ranking models taking into account the specificities of each time period.

3. WEB DOCUMENTS PERSISTENCE

Most ranking models have a static view of web documents and only consider their last version. We posit that web document persistence can be used to create discriminative features for improving the performance of ranking models. In this section, we analyze the correlation between the relevance of web documents and their long-term persistence.

3.1 Collection Description

We chose for this analysis a test collection built for WAIR research [8]. The general statistics are detailed in Table 1. This collection includes a corpus with 269 801 assessed web document versions using a three-level scale of relevance (not-

[2] http://lucene.apache.org/

[3] http://archive-access.sourceforge.net/projects/nutch/

document versions	255 million
data volume	8.9 TB
date range	1996 to 2009
navigational queries	50
average query length	2.23
assessed document versions	269 801
assessment scale of relevance	3-level

Table 1: Test collection statistics.

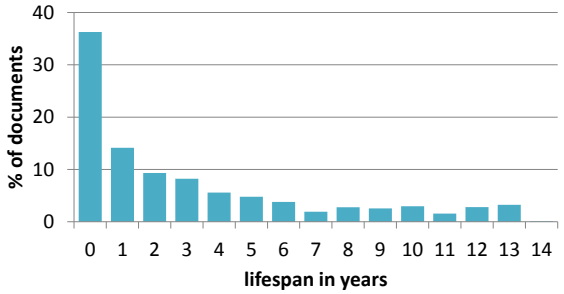

Figure 1: Distribution of the lifespan of documents in years.

relevant, relevant and very relevant). The assessed document versions were returned by different ranking models in response to 50 navigational queries randomly sampled from the logs of a public web archive. This selection strategy enables to get a high coverage of relevant documents, especially because navigational queries tend to have only one (very) relevant document. The queries have 2.23 terms on average and 1/3 are restricted by date range.

The documents range over a period of 14 years, from 1996 to 2009. Such characteristics make this collection unique to study long-term persistence of web documents and their relation to relevance ranking. For instance, to study content change, Elsas and Dumais used a collection of 2 million documents crawled for a period of 10 weeks [10], Adar et al. used 55 thousand documents crawled during 5 weeks [1], Fetterly et al. crawled 150 million documents over a period of 11 weeks [11] and Ntoulas et al. 150 web sites over the course of 1 year [24]. These are much shorter periods of analysis not so adequate to this study.

3.2 Document Persistence

We analyzed the persistence of web documents measured by their lifespan (i.e. difference in days between the first and last versions) and their number of versions. For simplification, we identified the versions of a URL by comparing their MD5 checksums.

Figure 1 shows the lifespan distribution of web documents. Around 36% of documents have less than one year and hence, we assigned a lifespan of 0 years. This percentage is inferior to the 50% reported by Ntoulas et al. [24]. 14% have a lifespan between 1 and 2 years and near 8% have a lifespan longer than 10 years. Figure 2 shows the distribution of the number of versions of documents. Around 36% have just 1 version, 29% have between 2 and 10, and 35% have more than 10.

The lifespan and number of versions present different distributions. While the number of versions fits a logarithmic distribution, the lifespan resembles a long tail distribution. When inspecting the documents, we saw that the document with most versions is the homepage of a newspaper (`http://www.correiomanha.pt/`) with 1301 versions and a lifespan of 12 years and a half. The document with the longest lifespan contains a list of scientific books for the younger (`http://nautilus.fis.uc.pt/softc/Read_c/l_infantis/infantis.html`) with a lifespan of 13 years and 2 months, but with just 8 versions. While all the documents with the highest number of versions have a long lifespan, the opposite is not true. In fact, the top ten documents with the longest lifespans have less than 15 versions. The Pearson correlation coefficient between the number of versions and the lifespan of web documents is 0.52.

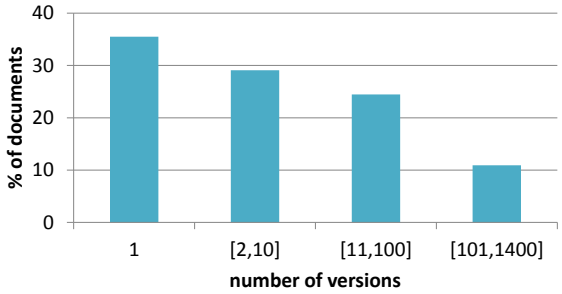

Figure 2: Distribution of the number of versions of documents over 14 years.

3.3 Document Persistence & Relevance

We found some interesting patterns when analyzing the relationship between the long-term persistence of web documents and their relevance. Figure 3 shows the fraction of documents that have a lifespan longer than 1 year for each relevance level, i.e. the number of documents with a given relevance level and a lifespan longer than 1 year, divided by the total number of documents with that same relevance level. The figure shows that these documents are likely to have a higher relevance. The same correlation exists for documents between 1 and 5 years. The percentage of very relevant documents with more than 5 years is only 1% of the total documents for the 50 queries analyzed, which makes it difficult to identify any meaningful correlation. Nevertheless, the sum of the relevant and very relevant fractions of documents is always superior to the not-relevant when considering the documents with a lifespan longer than 1 year. This indicates that the relevant documents tend to have a longer lifespan.

Figure 4 shows the fraction of documents that have more than 10 versions for each relevance level. These documents tend to have a higher relevance, such as the documents between 1 and 30 versions. The percentage of very relevant documents with more than 30 versions is only 1% of the total documents for the 50 queries analyzed. The 1% is the threshold where once again the correlation starts to be insignificant. However, the sum of the relevant and very relevant fractions of documents is always superior to the not-relevant when considering until 300 versions. After this number, the 4% of remaining documents present a different pattern. Even so, in general, these results indicate that relevant documents tend to have more versions.

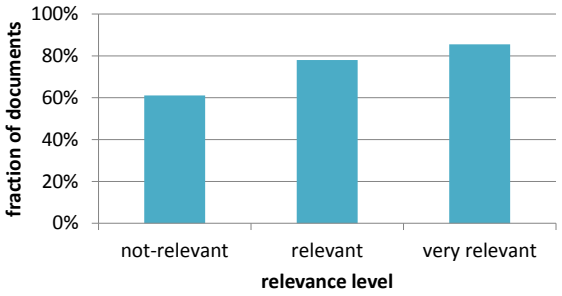

Figure 3: Fraction of documents with a lifespan longer than 1 year for each relevance level.

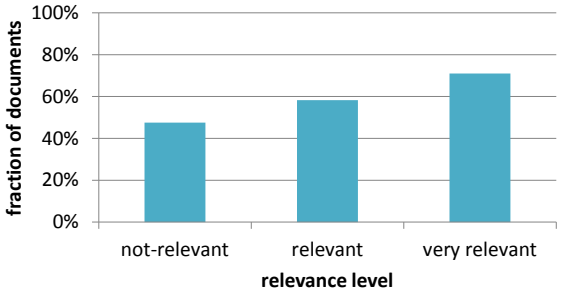

Figure 4: Fraction of documents with more than 10 versions for each relevance level.

3.4 Modeling Document Persistence

The lifespan and number of versions of documents are not correlated between them, but both are correlated with the relevance of documents. Hence, to leverage this correlation we modeled these measures of persistence with a logarithmic function that gives a higher score to: (1) documents with a longer lifespan; (2) documents with more versions. Both are defined by the same function:

$$f(d) = \log_y(x) \tag{1}$$

where, for the first case, x is the number of days between the first and last versions of document d, and for the second case, x is the number of versions of document d. The y is the maximum possible x for normalization. This is just an example of a function that can be used to create ranking features, such as these two features that we will use ahead in this study.

4. TEMPORAL-DEPENDENT RANKING

In this section, we present our temporal-dependent ranking framework for improving search effectiveness. First, we formalize the ranking problem. Second, we explain how to divide the training data by time, and third, how to use these data to create temporal-dependent models. Fourth, we describe how to learn all models simultaneously and how to combine them to produce a final ranking score. Last, we present how to implement our framework.

4.1 Ranking Problem

The traditional ranking problem is to find a ranking model f with parameters ω that receives X as input, where X is an $m \times d$ matrix of m query-document feature vectors of size d. This model f produces a vector $\hat{y}$ of m ranking scores, one per query-document pair $< q, d >$, trying to predict the real relevance of document d for query q:

$$\hat{y} = f(X; \omega) \tag{2}$$

Manually finding and optimizing f is a laborious and prone to error work, especially when f combines multiple features. Hence, L2R algorithms automatically learn the best model $\hat{f}$, such that $\hat{f}$ minimizes the given loss function L:

$$\hat{f} = \arg\min_{f \in F} \sum_{i=1}^{m} L(f(X_i; \omega), y_i) \tag{3}$$

where X_i represents the i^{th} query-document feature vector and y_i the corresponding relevance label. As Eq. 3 shows, the typical L2R outcome is a single general model that ranks documents independently when they were created or updated.

4.2 Temporal Intervals

Contrary to the traditional ranking problem, we learn multiple ranking models, each taking into account the specific characteristics of a period. In order to achieve that, we first identify a set of temporal intervals $T = \{T_1, T_2, ..., T_n\}$, from which we then learn multiple ranking models $M = \{M_1, M_2, ..., M_n\}$. Each interval T_k has associated a set of query-document feature vectors for training, where each feature vector i belongs to T_k if and only if the timestamp of the respective document version $t_i \in T_k$.

There are several timestamps associated to a document version, such as the dates of creation, modification, crawling or archiving. The creation and modification dates are good choices, since they refer to the time when a version was created. However, identifying them is not straightforward. The metadata from the document's HTTP header fields, such as Date, Last-Modified and Expires are not always available, nor reliable. Studies estimate that from 35% to 64% of web documents have valid last-modified dates [14], but these percentages can be significantly improved by using the dates of the web document's neighbors, especially of web resources embedded in the selected document (e.g. images, CSS, JavaScript) [25]. Nevertheless, for simplification, in this work we adopted the crawling date.

4.3 Temporal-Dependent Models

It is hard to establish clear temporal boundaries in web data, because the ranking features tend to change gradually over time rather than abruptly. Thus, a model M_k is learned using all training instances of all intervals T, but each training instance contributes with a different weight to the learning of M_k. The instances of interval T_k contribute with a maximum weight, while the instances of other intervals $T_j \neq T_k$ contribute with a weight defined by their temporal distance to T_k. Consider Figures 5(a), 5(b) and 5(c) as illustrative examples. They depict the weights of a collection with web snapshots between time points t_1 and t_4. Let's assume that we want to create 3 different models, $M = \{M_1, M_2, M_3\}$, taking into account the different characteristics of the web snapshots over time. For that, we divide the collection in 3 time intervals $T = \{T_1, T_2, T_3\}$ or $T = \{[t_1, t_2],]t_2, t_3],]t_3, t_4]\}$. Figure 5(a) shows that the training instances of interval T_1, such as v_1, are used with

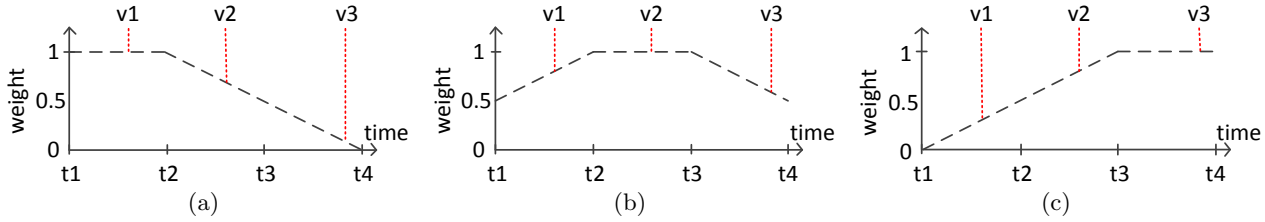

Figure 5: Weights of training instances, such as v_1, v_2 and v_3, when learning ranking models (a) M_1, (b) M_2 and (c) M_3.

weight 1 when learning M_1, while the other instances receive a weight that decreases as the timestamps of the instances move away from T_1, such as v_2 and v_3. Figures 5(b) and 5(c) show the values returned by temporal weight functions when learning M_2 and M_3, respectively.

Contrary to typical learning to rank, our goal is to learn the best model $\hat{f}$ for a temporal interval T_k, such that $\hat{f}$ minimizes the following loss function L:

$$\hat{f} = \arg\min_{f \in F} \sum_{i=1}^{m} L(\gamma(X_i, T_k) f(X_i; \omega), y_i) \quad (4)$$

where γ is the temporal weight function. We can adopt several γ functions with the underlying idea that the weight decreases as the temporal distance increases, such as the following function:

$$\gamma(X_i, T_k) = \begin{cases} 1 & \text{if } X_i \in T_k \\ 1 - \alpha \frac{distance(X_i, T_k)}{|T|} & \text{if } X_i \notin T_k \end{cases} \quad (5)$$
$$s.t.\ 0 \leq \gamma \leq 1$$

where $distance(X_i, T_k)$ is the absolute difference between the date of document version in X_i and the closer date to interval T_k, i.e. to the begin or end of T_k. $|T|$ denotes the total time covered by the collection. The γ function may have a larger or a smaller slope α to learn ranking models with higher or lower contribution of the training instances. For instance, by having a α of 2 instead of 1, the ranking model will be learned with half the contribution of the training instances and will ignore the instances in the half most distant intervals.

4.4 Multi-task Learning

A temporal-dependent model has two advantages over a model that only learns from data of a segment of time. First, solutions where each model learns from a part of the training data tend to present bad performance results, because more data usually beats better machine learning algorithms [2]. Thus, each temporal-dependent model considers all training instances during learning, avoiding the problem of the lack of data. Second, a temporal-dependent model considers the dependency between datasets of different temporal intervals. A model will learn more from instances of closer intervals than from instances of intervals more far apart.

Another important aspect is that we want to minimize the overall prediction error of all temporal-dependent models, since all will be employed to rank query results. Hence, we minimize a global relevance loss function, which evaluates the overall training error, instead of minimizing multiple independent loss functions without considering the correlation and overlap between models, i.e. instead of minimizing Eq. 4 for each model, we minimize:

$$\hat{f}_1, ..., \hat{f}_n = \arg\min_{f_1,...,f_n \in F} \sum_{i=1}^{m} L(\sum_{j=1}^{n} \gamma(X_i, T_j) f_j(X_i; \omega), y_i) \quad (6)$$

where n is the number of temporal-dependent ranking models. The minimization of this global loss function enables learning all models simultaneously to optimize a unified relevance target. Notice that each training instance X_i is shared by each model f_j and the closer the time interval T_j to X_i the greater this sharing. Models based on data learned from time intervals far apart, will share little or no information of X_i. This is important for distant time intervals do not end up influencing negatively each other.

After learning all temporal-dependent models, we employ an unsupervised ensemble method to produce the final ranking score. We run each of the n ranking models f_j against a testing instance X_i multiplied by its temporal weight γ to the corresponding interval T_j. Then, we sum all scores produced by all ranking models:

$$score(X_i) = \sum_{j=1}^{n} \gamma(X_i, T_j) f_j(X_i; \omega) \quad (7)$$

This ensemble method follows the global loss function (Eq. 6) used in the learning phase, trying to minimize the overall prediction error and improve the final search effectiveness.

4.5 L2R Algorithm

Our temporal-dependent ranking framework is quite flexible and can be implemented using different L2R algorithms as long as we adapt them to use the global loss function of Eq. 6. We followed the work of Bian et al. and adapted the RankSVM algorithm [3].

The goal of RankSVM is learning a linear model that minimizes the number of pairs of documents ranked in the wrong relative order [15]. Formally, RankSVM minimizes the following objective function:

$$\min_{\omega, \xi_{q,i,j}} \frac{1}{2}||\omega||^2 + C \sum_{q,i,j} \xi_{q,i,j}$$
$$s.t.\ \omega^T X_i^q \geq \omega^T X_j^q + 1 - \xi_{q,i,j}, \quad (8)$$
$$\forall X_i^q \succ X_j^q, \xi_{q,i,j} \geq 0$$

where $X_i^q \succ X_j^q$ implies that document i is ranked ahead of document j with respect to query q. C is a trade-off coefficient between the model complexity $||\omega||^2$ and the training error $\sum \xi_{q,i,j}$.

We modified the objective function of RankSVM following our global loss function, which takes into account the feature specificities of web snapshots over time. Each temporal-dependent ranking model M_k is learned by minimizing the following objective function:

$$\min_{\omega,\xi_{q,i,j}} \frac{1}{2}\sum_{k=1}^{n}||\omega_k||^2 + C\sum_{q,i,j}\xi_{q,i,j}$$

$$s.t.\ \sum_{k=1}^{n}\gamma(X_i^q,T_k)\omega_k^T X_i^q \geq \sum_{k=1}^{n}\gamma(X_j^q,T_k)\omega_k^T X_j^q + 1 - \xi_{q,i,j},$$

$$\forall X_i^q \succ X_j^q, \xi_{q,i,j} \geq 0 \quad (9)$$

5. EXPERIMENTAL SETUP

This section presents our experimental setup that enabled us to answer the following questions:

1. How much can we improve the search effectiveness of state-of-the-art WAIR using the L2R framework? We believe that the observations made in the context of L2R applied to document retrieval hold in relation to WAIR, but this hypothesis has not been tested.

2. Do temporal features intrinsic to web archives improve WAIR, such as the features based on the long-term persistence of web documents described in Section 3?

3. Does our temporal-dependent ranking framework described in Section 4 improve WAIR over a single general model that fits all data independently of its time?

Next, we give a brief description of the L2R dataset and the ranking features used in the experiments. Then, we present the compared ranking algorithms and models, and for last, the evaluation methodology and metrics.

5.1 L2R Dataset

The L2R dataset is composed by a set of *<query, document version, grade, features>* quadruples, where the grade indicates the relevance degree of the document version to the query. The features represent a vector of ranking feature values, each describing an estimate of relevance for the *<query, document version>* pair.

From the 269 801 *<query, document version>* pairs assessed in the test collection described in Section 3.1, we extracted 39 608 quadruples with 68 features. This is the size of the dataset, which has on average 843 versions per query. 3 queries were excluded from the 50, because their relevant and very relevant versions did not contain all features.

Table 2 shows the distribution of relevance judgments per relevance grade. As expected, the number of relevant and very relevant versions is much less than the not-relevant. Notice that for each of these navigational queries there is usually only one very relevant version and/or one relevant version. The dataset is publicly available for research at `http://code.google.com/p/pwa-technologies/wiki/L2R4WAIR`.

5.2 Ranking Features

The effectiveness of the ranking models greatly depends on the quality of the features they use. We give an overview of the classes of the 68 features released in the L2R dataset.

Grade	very relevant	relevant	not relevant
# judgments	4 610	4 357	30 641

Table 2: Distribution of relevance judgments in the L2R dataset per relevance grade.

Each class explores a different type of data:

term-weighting features estimate the similarity between the query and the different sections of a document version (anchor text of incoming links, text body, title and URL), such as Okapi BM25 [27].

term-distance features use the distance between terms in the different sections of a document version to quantify the relatedness between them, such as the Minimal Span Weighting function [23].

URL features compute an importance measure based on the probability of URLs representing an entry page, using the number of slashes, their length, or if they refer to a domain, sub-domain or page [18].

web-graph features estimate the popularity or importance of a document version inferred from the graph of hyperlinks between versions. These features include the number of inlinks to a version.

temporal features consider the time dimension of the web. They include the age of a document version and the two features described in Section 3.4 based on the long-term persistence of web documents.

Some of these features are typically used in web search engines and their results have been proven over time. The temporal features, however, were implemented specifically for this research. The complete list of features can be consulted online[4]. All feature values were scaled to a range between 0 and 1 using a min-max normalization.

5.3 Ranking Algorithms

The way L2R algorithms learn can be categorized into three approaches: pointwise, pairwise and listwise [21]. We employed three state-of-the-art L2R algorithms that cover the three approaches:

pointwise: *Random Forests* consists of multiple regression trees, where each tree is built from a bootstrap sample of the training data and a random subset of features is selected to split each node of a tree [4]. The relevance score of each document is the average of the outputs of the individual regression trees.

pairwise: *RankSVM* which is described in Section 4.5.

listwise: *AdaRank* is a boosting algorithm that linearly combines "weak learners", which are iteratively selected as the feature that offers the best performance among all others [30]. Each new learner focus on the queries not ranked well on previous iteration, by giving more weight to them.

RankSVM and AdaRank produce linear models, while Random Forests produce nonlinear models. In all experiments we used the RankSVM implementation available in the SVM^{rank} software[5] and the implementation of the other two L2R algorithms in the *RankLib* software[6].

[4] `http://pwa-technologies.googlecode.com/files/featureList.pdf`
[5] `http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html`
[6] `http://www.cs.umass.edu/~vdang/ranklib.html`

5.4 Ranking Models Compared

To compare the search effectiveness of the proposed approaches we evaluated the following ranking models:

1. **Models with manually tuned features:** these are baseline models. For comparison we included the results of three ranking models with manually tuned features, obtained from a related work [8]. The first model is the Okapi BM25 with parameters k1=2 and b=0.75 [27]. The second is Lucene's term-weighting function[7], which is computed over five fields (anchor text of incoming links, text body, title, URL and hostname of URL) with different weights. The third is a small variation of Lucene used in NutchWAX, with a different normalization by field length. These last two models can be considered the state-of-the-art in WAIR, since the most advanced IR technology currently used in web archives is based on the Lucene and NutchWAX search engines [13].
2. **Models with regular features combined with L2R:** these are another class of baseline models, but based on the technology usually employed in web search engines. These models contain all ranking features described in Section 5.2, except the temporal features. The regular features were automatically combined using the L2R algorithms to create a single ranking model. These models are denoted as the single-model approach with regular features.
3. **Models with all features combined with L2R:** these are the same models as in the previous point, but with all ranking features, regular and temporal. All these features were automatically combined by L2R algorithms to create a single ranking model. We refer to these models as the single-model approach with all features.
4. **Models with regular features combined with the temporal-dependent ranking framework:** unlike the previous models created independently of the time of each document version, these ranking models were created using the temporal-dependent ranking framework proposed in Section 4. The framework used equal intervals of time with an approximate number of training instances. The models only contain regular features.
5. **Models with all features combined with the temporal-dependent ranking framework:** these are the same models as in the previous point, but with all ranking features, regular and temporal.

5.5 Evaluation Methodology and Metrics

We chose a five-fold cross-validation to compare the average performance of the different ranking models. The L2R dataset was divided in five folders, where each folder has three subsets: training, validation and testing. Each ranking model was created, for each folder, using the training data. The validation data was used to tune the parameters of the L2R algorithms and the test data was used only on the evaluation of the model to avoid overfitting. The final results are the averages of the five tests.

Each of the 50 evaluated navigational queries may have one very relevant version and several relevant versions. Considering this fact, the ranking models were evaluated with two of the most used evaluation metrics: Precision at three cut-off values (P@1, P@5 and P@10) and the Normalized Discount Cumulative Gain at the same three cut-off values (NDCG@1, NDCG@5 and NDCG@10). $P@k$ measures the relevance of the top k document versions in a ranking list with respect to a query and is calculated as follows:

$$P@k = \frac{\sum_{i=1}^{k} r(i)}{k}$$

where r(i) is the relevance of the document version ranked at position i. Precision works over binary judgments. Due to that, the very relevant and relevant judgments were both taken as relevant when using P@k.

NDCG@k handles multiple levels of relevance and gives a higher score to relevant documents in higher ranking positions. It is calculated as follows:

$$NDCG@k = Z_k \sum_{i=1}^{k} \frac{2^{r(i)}-1}{\log_2(1+i)}$$

where Z_k is a normalization constant for the perfect list to get a NDCG@k of 1.

The past experience in web archive assessment has shown that users do not want to see multiple versions of a URL on the search results, but rather only one URL with a link to a list of all the other versions of that URL [8]. This corresponds to the common behavior already implemented in the user interfaces of existing WAIR systems. Hence, we evaluate only the first document version shown in the search results and ignore all the other versions of the same URL, before applying P@k or NDCG@k.

6. EXPERIMENTAL RESULTS

In this section we report and discuss the results of the tested ranking models, summarized in Table 3.

Baselines.

The NutchWAX model performs better than the Lucene and BM25 models. However, its performance is significantly worse than the models produced by the L2R algorithms using regular features. For instance, the model produced with the Random Forests algorithm, which presents the best results of the three L2R algorithms, has a NDCG@10 of 0.650, while NutchWAX gets 0.174. This is more than a three times increase. All models derived from L2R algorithms achieved better results than NutchWAX in all metrics with a statistical significance of $p<0.01$ using a two-tailed paired Student's t-test. This strongly indicates, as expected, that the use of L2R with ranking features typically used in web search engines, improves the search effectiveness of web archives, but also that the commonly used WAIR engines have a quite poor performance.

Temporal features.

All previous models are baselines. Hence, we compared only against the strongest baseline, i.e. the models with regular features combined with L2R algorithms. We analyzed the discriminative power of the temporal ranking features by running the L2R algorithms with and without these features. We can see a clear pattern. The L2R algorithms almost always present statistically significant improvements for all metrics when using the temporal features. For instance, Random Forests has a NDCG@1 superior in 10% to the same algorithm learning without the temporal features and RankSVM increased 3 percentage points. Therefore, the temporal features intrinsic to web archives can be used to improve WAIR.

[7] http://lucene.apache.org/java/2_9_0/api/all/org/apache/lucene/search/Similarity.html

Metric	models with features manually tuned			models with regular features combined with L2R			models with all features combined with L2R		
	BM25	Lucene	NutchWAX	AdaRank	RankSVM	R. Forests	AdaRank	RankSVM	R. Forests
NDCG@1	0.250	0.220	0.250	0.380 †	0.500 †	0.550 †	0.400 †	0.530 †‡	**0.650** †‡
NDCG@5	0.145	0.157	0.215	0.427 †	0.485 †	0.610 †	0.426 †	0.546 †‡	**0.665** †‡
NDCG@10	0.119	0.133	0.174	0.470 †	0.523 †	0.650 †	0.476 †	0.571 †‡	**0.688** †‡
P@1	0.300	0.280	0.320	0.460 †	0.560 †	0.640 †	0.480 †	0.580 †‡	**0.760** †‡
P@5	0.140	0.164	0.236	0.264 †	0.276 †	0.390 †	0.260 †	0.324 †‡	**0.396** †‡
P@10	0.108	0.132	0.168	0.182 †	0.194 †	0.236 †	0.182 †	0.196 †	**0.238** †

† shows a statistical significance of p<0.01 against NutchWAX with a two-sided paired t-test, while ‡ shows a statistical significance of p<0.05 against the models with regular features combined with L2R (i.e. we compare the same model with and without temporal features). The bold entries indicate the best result achieved in each metric.

Table 3: Results of the tested ranking methods.

Temporal-dependent ranking framework.

Finally, we analyzed the single-model approach versus the temporal-dependent ranking framework, with and without temporal features. Figures 6 and 7 show the NDCG@1, NDCG@5 and NDCG@10 values obtained with the temporal-dependent ranking framework, when using regular features or all features. We tested the framework with different time intervals (1, 2, 4, 7 and 14) and different slopes α in the temporal weight function (0.25, 0.5, 0.75, 1, 1.25 and 1.5). Notice that we used a test collection with 14 years of web snapshots. Thus, when we use 14 or 7 time intervals, it means that a model is created for each year or two years, respectively. The use of 1 time interval is similar to creating just one model, i.e. the single-model approach.

The results show that the proposed temporal-dependent ranking framework outperforms the single-model approach, with and without temporal features. We achieved improvements for all time intervals, but the highest improvements were obtained when we used 4 or 7 intervals. Results depicted in Figure 6 without temporal features, show that the major increase for NDCG@1 was from 0.500 to 0.560 (+6%) when using 4 and 7 intervals, while for a NDCG@5 was from 0.485 to 0.551 (+6.6%) and for NDCG@10 was from 0.523 to 0.572 (+4.9%), both when using 4 intervals. Results depicted in Figure 7 with temporal features, show that the major increase for NDCG@1 was from 0.530 to 0.590 (+6%) when using 7 intervals, while for a NDCG@5 was from 0.546 to 0.583 (+3.7%) and for NDCG@10 was from 0.571 to 0.604 (+3.3%), both when using 4 intervals. All these improvements, which present a statistical significance (p<0.05), indicate that the values of the ranking features change considerably over time in a way that can be learned by ranking models to better differentiate between relevant and not-relevant documents.

The slope α of the temporal weight function in Eq. 5 has an important impact in the final results. We obtained the worst results when α is larger than 1, i.e. when the contribution of the training instances is smaller. On the other hand, a small α, such as 0.25, caused a larger than desired contribution of the training instances. The best results were achieved with α between 0.5 and 1.

The temporal features and the temporal-dependent ranking framework, are independent approaches that demonstrate promising results. However, both approaches also work well together. In fact, the results displayed in Figure 7 show that we achieved the best results when we combined them. The NDCG@1, NDCG@5 and NDCG@10 are superior in 9%, 10% and 8%, respectively, over the single-model approach using just regular features.

BM25 over all fields
TF-IDF over all fields
Number of versions of a URL
TF-IDF over the hostname of URL
Length of the shortest text with all query terms in title
Days between the first and last versions of a URL

Table 4: Top 6 most important ranking features for the temporal-dependent ranking framework.

6.1 Results Analysis

We analyzed why the temporal-dependent ranking framework produces better results than the typical single ranking model created by L2R algorithms. We sorted the ranking features by their importance, measured by the absolute weight assigned by RankSVM. We found that the top features are almost the same, whether using just one model or multiple temporal-dependent models. The difference between ranking models created for different time intervals lies on small changes in the weights of features. This finding corroborates our observations that the characteristics of web documents evolve smoothly rather than abruptly and the temporal-dependent ranking models can adjust the feature weights to provide fine-grained ranking over time.

Table 4 shows the top 6 most important ranking features for the temporal-dependent ranking framework. From this table, we can see that BM25 and TF-IDF over all fields are the features with higher weight. The features based on long-term persistence of web documents, using the number of versions and the number of days between the first and last versions, are also at the top. RankSVM weighted some of these as the best features to identify relevant document versions for navigational queries.

7. CONCLUSION & FUTURE WORK

The retrieval effectiveness of state-of-the-art WAIR systems is poor, preventing users from unfolding the full potential of web archives. This work made a few contributions to face this problem. We studied, for the first time, the effects of long-term web document persistence in relevance ranking. In our experiments, conducted over 14 years of

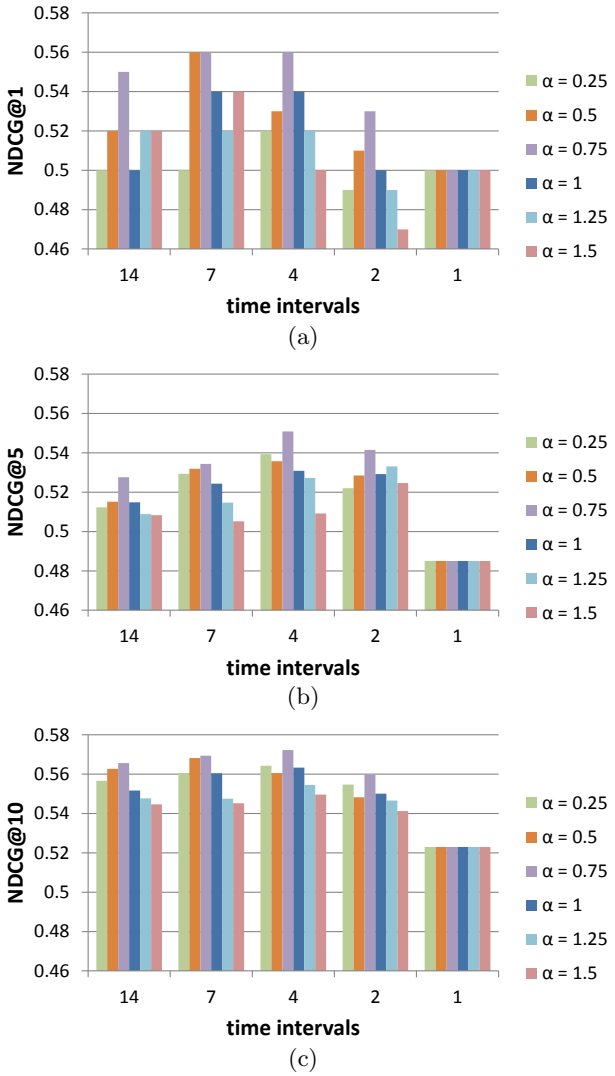

Figure 6: (a) NDCG@1, (b) NDCG@5 and (c) NDCG@10 results of the temporal-dependent ranking framework using different time intervals and α values of the temporal weight function. These models contain regular features.

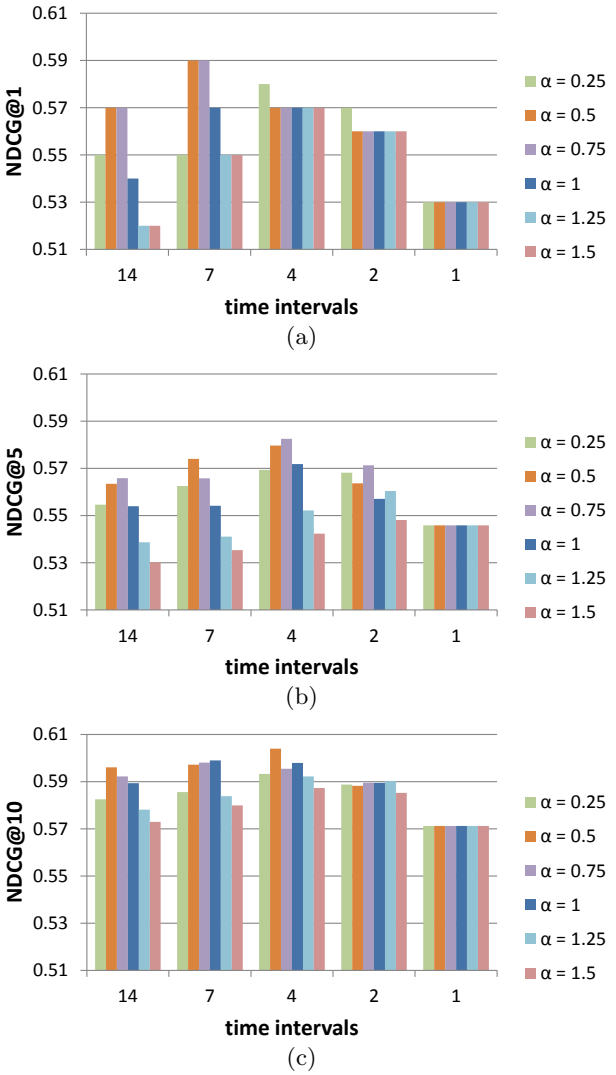

Figure 7: (a) NDCG@1, (b) NDCG@5 and (c) NDCG@10 results of the temporal-dependent ranking framework using different time intervals and α values of the temporal weight function. These models contain regular and temporal features.

web snapshots, we found that relevant documents tend to have a longer lifespan and more versions. Significant gains were achieved by modeling these persistence characteristics of web documents as novel ranking features. Additionally, since the characteristics of the web vary over time, both in structure and content, we proposed a temporal-dependent ranking framework that learns a different ranking model for each successive web period. Our experimental results show that the proposed multi-model framework outperforms a simpler approach based on a single ranking model, when both use the same L2R algorithms.

This work is focused in WAIR, but we believe that our approach could bring similar improvements to any digital libraries dealing with versioned content spanning long periods. As future work, we intend to study the evolution of URLs over time and their impact on search, since we detected changes in many URLs' top-level domains and subdomains. By tracking this evolution, we can better measure the long-term persistence of web documents. We also plan to investigate how to extend the temporal-dependent ranking framework to handle temporal diversity in search results.

8. ACKNOWLEDGMENTS

We thank FCT for the financial support of the Research Units of LaSIGE (PEst-OE/EEI/UI0408/2014) and INESC-ID (Pest-OE/EEI/LA0021/2013), and the DataStorm Research Line of Excellency (EXCL/EEI-ESS/0257/2012).

9. REFERENCES

[1] E. Adar, J. Teevan, S. Dumais, and J. Elsas. The web changes everything: understanding the dynamics of web content. In *Proc. of the 2nd ACM International Conference on Web Search and Data Mining*, pages 282–291, 2009.

[2] M. Banko and E. Brill. Scaling to very very large corpora for natural language disambiguation. In *Proc. of the 39th Annual Meeting on Association for Computational Linguistics*, pages 26–33, 2001.

[3] J. Bian, X. Li, F. Li, Z. Zheng, and H. Zha. Ranking specialization for web search: a divide-and-conquer approach by using topical RankSVM. In *Proc. of the 19th International Conference on World Wide Web*, pages 131–140, 2010.

[4] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] Y. Chung, M. Toyoda, and M. Kitsuregawa. A study of link farm distribution and evolution using a time series of web snapshots. In *Proc. of the 5th International Workshop on Adversarial Information Retrieval on the Web*, pages 9–16, 2009.

[6] M. Costa and M. J. Silva. Understanding the information needs of web archive users. In *Proc. of the 10th International Web Archiving Workshop*, pages 9–16, 2010.

[7] M. Costa and M. J. Silva. Characterizing search behavior in web archives. In *Proc. of the 1st International Temporal Web Analytics Workshop*, pages 33–40, 2011.

[8] M. Costa and M. J. Silva. Evaluating web archive search systems. In *Proc. of the 13th International Conference on Web Information Systems Engineering*, pages 440–454, 2012.

[9] N. Dai, M. Shokouhi, and B. Davison. Learning to rank for freshness and relevance. In *Proc. of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 95–104, 2011.

[10] J. Elsas and S. Dumais. Leveraging temporal dynamics of document content in relevance ranking. In *Proc. of the 3rd ACM International Conference on Web Search and Data Mining*, pages 1–10, 2010.

[11] D. Fetterly, M. Manasse, M. Najork, and J. L. Wiener. A large-scale study of the evolution of web pages. In *Proc. of the 12th International Conference on World Wide Web*, pages 669–678, 2003.

[12] X. Geng, T. Liu, T. Qin, A. Arnold, H. Li, and H. Shum. Query dependent ranking using k-nearest neighbor. In *Proc. of the 31st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 115–122, 2008.

[13] D. Gomes, J. Miranda, and M. Costa. A survey on web archiving initiatives. In *Proc. of the International Conference on Theory and Practice of Digital Libraries*, pages 408–420, 2011.

[14] D. Gomes and M. Silva. Modelling information persistence on the web. In *Proc. of the 6th International Conference on Web Engineering*, pages 193–200, 2006.

[15] T. Joachims. Optimizing search engines using clickthrough data. In *Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 133–142, 2002.

[16] R. Jones and F. Diaz. Temporal profiles of queries. *ACM Transactions on Information Systems*, 25(3), 2007.

[17] I. Kang and G. Kim. Query type classification for web document retrieval. In *Proc. of the 26th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 64–71, 2003.

[18] W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In *Proc. of the 25th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 27–34, 2002.

[19] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graph evolution: densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1(1):2, 2007.

[20] X. Li and W. B. Croft. Time-based language models. In *Proc. of the 12th International Conference on Information and Knowledge Management*, pages 469–475, 2003.

[21] T. Liu. *Learning to rank for information retrieval*, volume 3 of *Foundations and Trends in Information Retrieval*. Now Publishers Inc., 2009.

[22] J. Masanès. *Web Archiving*. Springer-Verlag New York Inc., 2006.

[23] C. Monz. Minimal span weighting retrieval for question answering. In *Proc. of the SIGIR 2004 Workshop on Information Retrieval for Question Answering*, pages 23–30, 2004.

[24] A. Ntoulas, J. Cho, and C. Olston. What's new on the web?: the evolution of the web from a search engine perspective. In *Proc. of the 13th International Conference on World Wide Web*, pages 1–12, 2004.

[25] S. Nunes, C. Ribeiro, and G. David. Using neighbors to date web documents. In *Proc. of the 9th Annual ACM International Workshop on Web Information and Data Management*, pages 129–136, 2007.

[26] K. Radinsky and E. Horvitz. Mining the web to predict future events. In *Proc. of the 6th ACM International Conference on Web Search and Data Mining*, pages 255–264, 2013.

[27] S. Robertson and H. Zaragoza. *The probabilistic relevance framework*, volume 3 of *Foundations and Trends in Information Retrieval*. 2009.

[28] T. Salles, L. Rocha, G. L. Pappa, F. Mourão, W. Meira Jr, and M. Gonçalves. Temporally-aware algorithms for document classification. In *Proc. of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 307–314, 2010.

[29] N. Tahmasebi, G. Gossen, and T. Risse. Which words do you remember? Temporal properties of language use in digital archives. In *Proc. of the 2nd International Conference on Theory and Practice of Digital Libraries*, pages 32–37, 2012.

[30] J. Xu and H. Li. Adarank: a boosting algorithm for information retrieval. In *Proc. of the 30th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 391–398, 2007.

[31] Y. Yamamoto, T. Tezuka, A. Jatowt, and K. Tanaka. Honto? Search: estimating trustworthiness of web information by search results aggregation and temporal analysis. *Advances in Data and Web Management*, pages 253–264, 2007.