Precision and Recall of *GlOSS* Estimators for Database Discovery

Luis Gravano   Héctor García-Molina   Anthony Tomasic

Computer Science Department
Stanford University
Stanford, CA 94305-2140

{gravano,hector,tomasic}@cs.stanford.edu

1 Overview

On-line information vendors offer access to multiple databases. In addition, the advent of a variety of INTERNET tools [1, 2] has provided easy, distributed access to many more databases. The result is thousands of text databases from which a user may choose for a given information need (a user query). This paper, an abridged version of [3], presents a framework for (and analyzes a solution to) this problem, which we call the *text-database discovery problem* (see [3] for a survey of related work).

Our solution to the text-database discovery problem is to build a service that can suggest potentially good databases to search. A user's query will go through two steps: first, the query is presented to our server (dubbed *GlOSS*, for **Gl**ossary-**O**f-**S**ervers **S**erver) to select a set of promising databases to search. During the second step, the query is actually evaluated at the chosen databases. *GlOSS* gives a hint of what databases might be useful for the user's query, based on word-frequency information for each database. This information indicates, for each database and each keyword in the database vocabulary, how many documents at that database actually contain the keyword, for each field designator (Sections 2 and 3). For example, a Computer-Science library could report that *"Knuth"* (*keyword*) occurs as an *author* (*field designator*) in 180 documents, the keyword *"computer,"* in the *title* of 25,548 documents, and so on. This information is orders of magnitude smaller than a full index (see [4]) since for each keyword field-designation pair we only need to keep its frequency, not the identities of the documents that contain it.

To evaluate the set of databases that *GlOSS* returns for a given query, Section 4 presents a framework based on the precision and recall metrics of information-retrieval theory. In that theory, for a given query q and a given set S of relevant documents for q, *precision* is the fraction of documents in the answer to q that are in S, and *recall* is the fraction of S in the answer to q. We borrow these notions to define metrics for the text-database discovery problem: for a given query q and a given set of "relevant databases" S, P is the fraction of databases in the answer to q that are in S, and R is the fraction of S in the answer to q. We further extend our framework by offering different definitions for a "relevant database" (Section 4). We have performed experiments using query traces from the FOLIO library information-retrieval system at Stanford University, and involving six databases available through FOLIO. As we will see, the results obtained for different variants of *GlOSS* are very promising (Section 5). Even though *GlOSS* keeps a small amount of information about the contents of the available databases, this information proved to be sufficient to produce very useful hints on where to search.

2 *GlOSS*: Glossary-Of-Servers Server

Consider a *boolean "and"* query q [1] that we want to evaluate over a set of databases DB. *GlOSS* selects a subset of DB consisting of "good candidate" databases for actually submitting q. To make this selection, *GlOSS* has available the following information:

- $DBSize(db)$, the total number of documents in database db, $\forall\ db \in DB$, and
- $freq(t, db)$, the number of documents in db that contain t, $\forall\ db \in DB$, and for all keyword field-designation pairs t. Note that *GlOSS* does not have available the actual "inverted lists" corresponding to each keyword-field pair and each database, but just the length of these inverted lists.

[1] We can generalize this paper's approach to *"or"* queries (see [4]), and to the vector-space retrieval model [5].

This information is extracted from each database by a *collector* program (not discussed further here) that forwards it periodically to *GlOSS*.

To assess how good each database is for a given query, *GlOSS* uses an *estimator*: an estimator EST_ϵ (for some fixed $0 \leq \epsilon \leq 1$) consists of a function ESize_{EST} that predicts the result size of a query in a database, and a "matching" function that uses these estimates to select the "good" databases ($Chosen_{EST_\epsilon}$). Once $\mathsf{ESize}_{EST}(q, db)$ has been defined, we can determine $Chosen_{EST_\epsilon}(q, DB)$ in the following way:

$$Chosen_{EST_\epsilon}(q, DB) = \{db \in DB | \quad (1)$$
$$\mathsf{ESize}_{EST}(q, db) > 0 \wedge \left| \frac{\mathsf{ESize}_{EST}(q, db) - hest}{hest} \right| \leq \epsilon\}$$

where $hest = \max_{db' \in DB} \mathsf{ESize}_{EST}(q, db')$.

Users can set the value for ϵ according to the query semantics they are interested in: in general, higher values for ϵ make the $Chosen_{EST_\epsilon}$ set "larger:" if $\epsilon = 0$, only those databases containing the strictly highest non-zero estimates will belong to $Chosen_{EST_\epsilon}$, whereas if $\epsilon = 1$, all databases with a non-zero estimate will belong to $Chosen_{EST_\epsilon}$.

3 The Ind_ϵ estimators

The Ind_ϵ (for "independence") estimators [4] are built upon the (unrealistic) assumption that keywords appear in the different documents of a database following independent and uniform probability distributions [2]. Under this assumption, given a database db, any n keyword field-designation pairs $t_1, \ldots, t_n$, and any document $d \in db$, the probability that d contains all of $t_1, \ldots, t_n$ is:

$$\frac{freq(t_1, db)}{DBSize(db)} \times \ldots \times \frac{freq(t_n, db)}{DBSize(db)}$$

So, according to Ind_ϵ, the estimated number of documents in db that will satisfy the query "$find\, t_1 \wedge \ldots \wedge t_n$" is [6]:

$$\mathsf{ESize}_{Ind}(find\ t_1 \wedge \ldots \wedge t_n, db) = \frac{\prod_{i=1}^{n} freq(t_i, db)}{DBSize(db)^{n-1}}$$

The $Chosen_{Ind_\epsilon}$ set is then computed with Equation 1, for any value of ϵ.

4 Evaluation metrics

Let DB be a set of databases and q a query. In order to evaluate an estimator EST (e.g., $EST{=}Ind_\epsilon$), we need to compare its prediction against what actually is $Right(q, DB)$, the "right subset" of DB to query. There are several notions of what the right subset means, depending on the semantics the query submitter has in mind. In this paper, we will consider two of these notions, for which the goodness of a database db with respect to a query q will be determined by the number of documents that db returns when presented with q [3].

Our first definition for $Right(q, DB)$ is $Matching(q, DB)$, the set of all databases in DB containing at least one document that matches query q. More formally,

$$\begin{aligned} Right(q, DB) &= Matching(q, DB) \\ &= \{db \in DB | \mathsf{RSize}(q, db) > 0\} \end{aligned}$$

where $\mathsf{RSize}(q, db)$ is the actual result size of query q in database db. There are (at least) two types of users that may specify $Matching(q, DB)$ as their right set of databases. One is users that want an exhaustive answer to their query. They are not willing to miss any of the matching documents. We will refer to these users as "recall-oriented" users. On the other hand, "precision-oriented" users may be in "sampling" mode: they simply want to obtain *some* matching documents without searching useless databases.

Our second definition for $Right(q, DB)$ is, for a fixed $0 \leq \delta \leq 1$, $Best_\delta(q, DB)$, the set of those databases containing the most matching documents. More formally,

$$\begin{aligned} Right(q, DB) &= Best_\delta(q, DB) \\ &= \{db \in DB | \mathsf{RSize}(q, db) > 0 \wedge \\ &\quad \left| \frac{\mathsf{RSize}(q, db) - hreal}{hreal} \right| \leq \delta\} \end{aligned}$$

where $hreal = \max_{db' \in DB} \mathsf{RSize}(q, db')$. Parameter δ is not a parameter of our estimators, but of our evaluation metrics: the submitter of a query does not give a δ value to *GlOSS*. Higher values for δ yield more comprehensive $Best_\delta$ sets. Therefore, parameter δ should be fixed according to the desired "meaning" for $Best_\delta$. For example, suppose that we are evaluating Ind_ϵ for a user that wants to locate *Best* databases, but is willing to search at sites that have 90% or more of the number of matching documents than the overall *Best* sites have. Then, the experimental results that are relevant to this user are those obtained for $\delta = 0.1$.

Again, users that define $Best_\delta(q, DB)$ as their right set of databases for query q might be classified as being "recall oriented" or "precision oriented." "Recall-oriented" users are willing to miss some databases,

[2] Even though this assumption is unrealistic, we will see that the Ind_ϵ estimators work surprisingly well.

as long as they are not the best ones. These users want to ensure that at least those databases having the highest payoff (i.e., the largest number of documents) are searched. On the other hand, "precision-oriented" users want to examine (some) best databases. Due to limited resources (e.g., time, money) the users only want to submit their query at databases that will yield the highest payoff.

Once we have defined the *Right* set for a query q and a database set DB, we evaluate how well $Chosen_{EST}(q, DB)$ approximates $Right(q, DB)$ by adapting the well-known *precision* and *recall* parameters from information-retrieval theory [5] to the text-database discovery framework. If we regard *Right* as the set of "items" (databases in this context) that are relevant to a given query q, and $Chosen_{EST}$ as the set of items that is actually retrieved, we can define the following functions P_{Right} and R_{Right}, based upon the precision and recall parameters:

$$P_{Right}(q, DB) = \begin{cases} \frac{|Chosen \cap Right|}{|Chosen|} & \text{if } |Chosen| > 0 \\ 1 & \text{otherwise} \end{cases}$$

$$R_{Right}(q, DB) = \begin{cases} \frac{|Chosen \cap Right|}{|Right|} & \text{if } |Right| > 0 \\ 1 & \text{otherwise} \end{cases}$$

where $Chosen = Chosen_{EST}(q, DB)$ (EST is a fixed estimator for *GlOSS*) and $Right = Right(q, DB)$.

Intuitively, P is the fraction of selected databases that are *Right* ones, and R is the fraction of the *Right* databases that are selected. "Precision oriented" users will be interested in high values of P, while "recall oriented" users will be interested in high values of R.

Section 5 evaluates different estimators in terms of the average value, over a set of user queries, of the P and R parameters defined above, for different *Right* sets of databases.

5 Ind_ϵ results

In order to evaluate the performance of the Ind_ϵ estimators according to the P and R parameters of Section 4, we performed experiments using 6897 queries and six databases available through the FOLIO library information-retrieval system at Stanford University. Real users issued the queries to the INSPEC database through the FOLIO system. In [3], we describe the query trace and the experiments. We also study how well ESize_{Ind} approximates RSize.

Figure 1 shows the average values of the P and R parameters for Ind_0, for growing values of δ. Our estimator remains fixed (since $\epsilon = 0$) for the different values of δ, and so does *Matching*. This is why

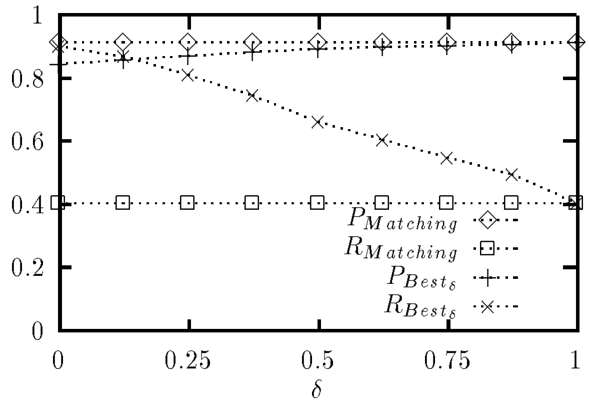

Figure 1: The average P and R parameters as a function of δ, for the Ind_0 estimator ($\epsilon = 0$).

the curves corresponding to $P_{Matching}$ and $R_{Matching}$ are flat. In particular, $P_{Matching}$=0.9126: this means that, for the average query, 91.26% of the databases in $Chosen_{Ind_0}$ have matching documents. In contrast, $R_{Matching}$=0.4044, meaning that, for the average query, $Chosen_{Ind_0}$ includes only 40.44% of the *Matching* databases: Ind_0 chooses only the most promising databases, not all of the ones that might contain matching documents. Higher values of ϵ address this problem (see below).

On the other hand, the set of best databases, $Best_\delta$, varies as δ does. In Figure 1 we see that parameter R_{Best_δ} worsens as δ grows, since $Best_\delta$ tends to contain more databases, while $Chosen_{Ind_0}$ remains fixed. This is exactly why P_{Best_δ} improves with higher values of δ. Note that for $\delta = 1$, $Best_1 = Matching$, and so, $P_{Matching}$ and $R_{Matching}$ coincide with P_{Best_δ} and R_{Best_δ}, respectively.

Figure 2 shows the average values of the P and R parameters for Ind_ϵ, for growing values of ϵ. For all these results, $\delta = 0$ (i.e., the "best" set of databases is fixed to $Best_0$). Since $Chosen_{Ind_\epsilon}$ tends to cover more databases as ϵ grows, $R_{Matching}$ and R_{Best_0} improve for higher values of ϵ. For $\epsilon = 1$, $R_{Matching}$= R_{Best_0}= 1, since $Chosen_{Ind_1}$ contains all of the potentially matching databases. This is also why P_{Best_0} worsens as ϵ grows. Parameter $P_{Matching}$ remains basically unchanged for higher values of ϵ, but worsens for ϵ close to one, for the same reasons P_{Best_0} gets lower.

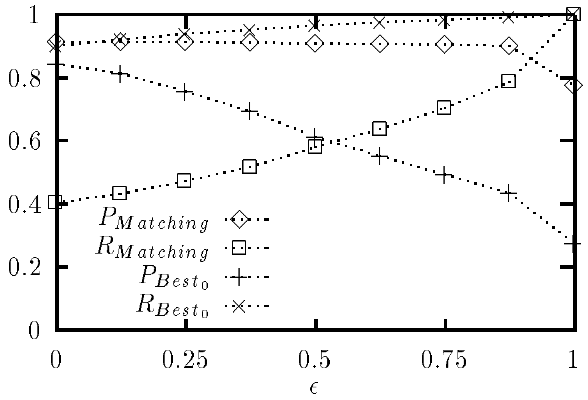

Figure 2: The average P and R parameters as a function of ϵ, for the Ind_ϵ estimator ($\delta = 0$).

Right	P_{Right} (Min_0)	R_{Right} (Min_0)	P_{Right} (Ind_0)	R_{Right} (Ind_0)
Matching	0.9077	0.4031	0.9126	0.4044
$Best_0$	0.8356	0.8938	0.8438	0.9010

Figure 3: The average P and R parameters for the Min_0 estimator. The last two columns show the corresponding values for the Ind_0 estimator.

6 Other results

The Ind_ϵ estimators are based upon the assumption that the occurrence of query keywords in documents follows independent and uniform probability distributions. We can build alternative estimators by departing from this assumption. For example, we can adopt the "opposite" assumption, and assume that the keywords that appear together in a user query are strongly correlated. So, we define another family of estimators for *GlOSS*, Min_ϵ (for "minimum"), by letting:

$$\mathsf{ESize}_{Min}(find\ t_1 \wedge \ldots \wedge t_n, db) = \min_{i=1}^{n} freq(t_i, db)$$

$\mathsf{ESize}_{Min}(q, db)$ is an upper bound of the actual result size of query q: $\mathsf{RSize}(q, db) \leq \mathsf{ESize}_{Min}(q, db)$. $Chosen_{Min_\epsilon}$ follows from the definition of ESize_{Min}, using Equation 1. As Figure 3 shows, the results we obtained for the Min_0 estimator, using the INSPEC queries, are very similar to those we obtained for the Ind_0 estimator.

To analyze how dependent the results are on the trace used, we ran our experiments using a different query trace, consisting of 2404 real-user queries.

Right	P_{Right} (ERIC)	R_{Right} (ERIC)	P_{Right} (INSPEC)	R_{Right} (INSPEC)
Matching	0.8960	0.4621	0.9126	0.4044
$Best_0$	0.8498	0.9384	0.8438	0.9010

Figure 4: The average P and R parameters for the Ind_0 estimator, using the ERIC queries. The last two columns show the corresponding values for the INSPEC queries.

Real users issued these queries to the ERIC database through Stanford's FOLIO system. Figure 4 shows the results corresponding to these queries, for the different instances of the P and R parameters. The results obtained differ only slightly from the ones for the INSPEC queries, which suggests that our results are not sensitive to the type of trace used.

Acknowledgments

This research was sponsored by the Advanced Research Projects Agency (ARPA) of the Department of Defense under Grant No. MDA972-92-J-1029 with the Corporation for National Research Initiatives (CNRI). This work was supported by an equipment grant from Digital Equipment Corporation. We would also like to thank Jim Davis and Daniela Rus for their helpful comments on an earlier version of this paper.

References

[1] Michael F. Schwartz, Alan Emtage, Brewster Kahle, and B. Clifford Neuman. A comparison of INTERNET resource discovery approaches. *Computer Systems*, 5(4), 1992.

[2] Katia Obraczka, Peter B. Danzig, and Shih-Hao Li. INTERNET resource discovery services. *IEEE Computer*, September 1993.

[3] Luis Gravano, Héctor García-Molina, and Anthony Tomasic. Precision and recall of *GlOSS* estimators for database discovery. Technical Report STAN-CS-TN-94-010, Stanford University, July 1994. Available by anonymous `ftp` from `db.stanford.edu` in `/pub/gravano/1994/stan.cs.tn.94.010.ps`.

[4] Luis Gravano, Héctor García-Molina, and Anthony Tomasic. The effectiveness of *GlOSS* for the text-database discovery problem. In *Proceedings of the 1994 ACM SIGMOD Conference*, May 1994. Also available by anonymous `ftp` from `db.stanford.edu` in `/pub/gravano/1994/stan.cs.tn.93.002.sigmod94.ps`.

[5] Gerard Salton and Michael J. McGill. *Introduction to modern information retrieval.* McGraw-Hill, 1983.

[6] G. Salton, E. A. Fox, and E. Voorhees. A comparison of two methods for boolean query relevance feedback. TR 83-564, Cornell University, July 1983.