Empirical Study of a Probabilistic Network Model for Retrieving Traceability Links

Xuchang Zou, Raffaella Settimi, Jane Cleland-Huang, Chuan Duan
Depaul University
School of Computer Science,
Telecommunication & Information System
Chicago, IL 60604
{xzou, rsettimi, jhuang, cduan}@cs.depaul.edu

Abstract

Requirements traceability provides critical support in helping manage the evolution of software systems. Establishing and maintaining trace links is an arduous problem requiring intensive human effort when traces need to be established manually. Automatic retrieval tools can help maintain traceability links by dynamically identifying traces between artifacts. In order to effectively reduce the effort involved in evaluating the correctness of the retrieved links such automatic tools must achieve high retrieval performance. This paper presents results of experimental studies to analyze the performance of a dynamic trace retrieval approach implementing a probabilistic information retrieval network. The experiments are conducted on three different datasets. An implementation of the retrieval approach described in the paper involves the definition of a small training set for which correct traces are previously defined. This can be accomplished by either using past knowledge or by manually evaluating the traces in the generated training set. A study explores the effect of different sized training sets on the retrieval performance of the automatic tool. A second study analyzes methods for defining confidence values, which are attached to each (un)retrieved trace to indicate how confident we are that the dynamically retrieved trace represents a true link, or vice versa that the link that are not retrieved by the tool are in fact false links. The results of this research are beneficial for enhancing the utility and performance of dynamic tracing tools.

1. Introduction

Requirement traceability has been widely recognized as an important factor in effectively managing the development and evolution of software systems. Researchers have investigated the application of information retrieval techniques to the modeling of traceability retrieval process and found these approaches promising [1,2,3,4]. The process of these automatic retrieval approaches consists of document parsing based on grammatical structure of the terms, link generation according to the term occurrence in both the requirement and the searchable documents, and finally presenting the candidate links to the user for their evaluation.

Although these dynamic retrieval methods have demonstrated the feasibility of replacing traditional trace tools, their performance is not entirely satisfactory. The performance of the retrieval tools is mainly evaluated by two metrics: recall and precision.

Recall is used to measure the number of correctly retrieved documents out of the set of all the documents that should be retrieved, while precision is used to measure the number of correctly retrieved documents out of the total set of retrieved documents. To effectively support the system impact analysis, tracing tools are expected to have high recall to retrieve as many relevant artifacts as possible. Research in this area shows that when recall of a traceability tracing tool reaches 90%, precision drops below 40%, sometimes even less than 10% [1,3,4,5]. When this occurs, the user is forced to conduct a time-consuming, error-prone search through the remaining document collection to find missed traces.

Our study aims to reduce the human effort involved in manual link discrimination by improving the performance of automatic traceability tools. To this end, a series of experiments has been conducted against three different datasets to analyze the performance of a dynamic trace retrieval approach implementing a probabilistic information retrieval network. Section 2 presents the dynamic trace retrieval approach utilized in our previous work. Section 3 describes the results of experiments to investigate the effect of different training sets on the retrieval algorithm performance. Section 4 proposes a method for computing confidence levels for both retrieved and rejected links. The confidence values express the degrees of belief that both the automatically retrieved links and the rejected ones are correct. Section 5 concludes with an analysis of the results of the empirical study and a discussion on future work.

2. Probabilistic Information Retrieval Model

The basic retrieval algorithm in our dynamic trace retrieval approach was implemented using a probabilistic network model based on Wong and Yao's work [7,8]. The model is represented by a directed acyclic graph where the nodes are random variables and arcs indicate relationships between variables. Three sets of variables are defined in this retrieval model: documents set ($d_1,d_2,\ldots,d_n$), queries set ($q_1,q_2,\ldots,q_m$) and term set ($t_1,t_2,\ldots,t_n$). The term set is derived from the document collection at a pre-processing stage by eliminating stop words and stemming those remaining words to their common grammatical roots.

Documents and queries are assumed to be conditionally independent given the index terms. This is represented in the graphical model by arcs only between document and terms nodes or query and terms nodes, while no direct arcs between query and document nodes are present. Thus we assume that there is no direct association between document and query nodes, but the lexical association between the two nodes can be completely explained by the term node. Each document or query can be interpreted as a proposition of the whole concept space defined by index terms ($t_1,t_2,\ldots,{}_n$). Then the probability pr(d|q) of a document d being relevant to a query d can be explained as the degree of coverage of d given q.

Based on this model, the probability of relevance of a document d_j with respect to a query q is defined as [6]:

$$pr(d_j \mid q) = \left[\sum_i pr(d_j \mid t_i) \mid pr(q,t_i) \right] / \, pr(q) \qquad (1)$$

where $pr(d_j \mid t_i) = \dfrac{freq(d_j, t_i)}{\sum_k freq(d_j, t_k)}$, $pr(q, t_i) = \dfrac{freq(q, t_i)}{n_i}$ and $pr(q) = \sum_i pr(q, t_i)$.

Documents d_j are ranked by their degree of relevance to the query q $pr(d_j|q)$. A threshold value is established prior to the retrieval so that all documents with probability score higher than the threshold will be retrieved.

3. Training Set Identification

The trace retrieval tool ranks the relevance of documents with respect to a given query according to a probability value computed using expression (1). A candidate link between query q and document d is established whenever the probability score pr(d|q) is above a certain threshold.

Such a threshold value can be identified using a training set that contains a subset of documents. The traces linking documents in the training set to queries are assumed to be known: these known links are either manually established or can be derived from previous knowledge. The size of the training set is typically small, since creating and maintaining the traceability matrix for the training set may become difficult if the number of traces is very large. A discussion on these issues is presented in section 3.2 below.

The following algorithm selects the threshold value for the probability scores computed by the retrieval algorithm. The threshold value is selected by maximizing a selection criterion that depends on two standard performance measures in information retrieval. Such measures are recall and precision. As previously stated, recall is the percentage of correctly retrieved links out of all correct links, and precision is the percentage of correct links retrieved out of all retrieved links.

First for each document d in the training set and each query q, the probability values pr(d|q) are computed using the expressions in section 2. A target recall Rt is selected by the user. The target recall is typically high, since the objective is to retrieve as many links as possible. The threshold value is evaluated by optimizing the target function "*approaching target recall Rt while maximizing recall + precision, where recall > precision*". Thus, the optimal threshold will be selected as the value T that yields a recall value close to a target recall Rt, while maximizing the retrieval precision.

The algorithm used to select thresholds from the training set is given in [9] and repeated here for convenience:

3.1 Optimal_Thresholding(training set, Rt)

```
Input: Training set, which contains tuples of query artifacts, document artifacts, and
       associated probability scores.
Input: Rt, denotes target recall, which is chosen by the user.
       For each category, do until termination condition is met:
       1. Initialize T to a value $T_0$, where $T_0 \in [0,1]$. We choose $T_0=0.1$ according to
          previous experiments.
       2. Calculate the output recall R and precision P over all queries given
          threshold T.
          Store T if R+P is maximized compared to all previous output pairs of
          recall and precision, and R > P.
```

3. T←T+ΔT, where $\Delta T = \alpha * (R - Rt)$. α is a relatively small positive number to constrain the threshold adjustment. For the purposes of these experiments $\alpha=0.01$.

Output: Optimal threshold T.

Termination condition is:

1. Threshold adjustment ΔT is small enough (for instance, $|\Delta T| \leq 0.0001$, which occurs when output recall R is closely achieved towards target recall Rt), or
2. T exceeds the upper bound 1 and lower bound 0 (which occurs when target recall Rt was set too high).

An experimental study was conducted to analyze the effect of size of the training sets on the retrieval performance of the automatic tracing algorithm. The experiments applied the automatic retrieval algorithm to generate links between requirement artifacts and class diagrams for three software applications which are described below.

IBS (Ice-Breaker) system is an application to deploy de-icing services on roads in a specific district. By using information gathered from weather stations and road sensors, the system forecasts road freezing conditions and manages the distribution of de-icing materials. Tasks of the system include maintaining district maps, managing de-icing material, arranging real-time truck work orders, planning de-icing and other issues. The Ice Breaker system consists of 172 functional requirements and 75 UML class diagrams.

EBT (Event-Based Traceability) system was originally developed at the International Center for Software Engineering at the University of Illinois at Chicago [6] to provide support for software system maintenance over a long term. It utilized a publish-subscribe mechanism to dynamically maintain artifacts in the system. The system is composed of 54 requirements and 60 UML classes.

The third database, Light Control (LC) system was developed for the University of Kaiserslautern [2] to manage the lighting tasks in a building. According to the space occupation of the building and the current illumination, the LC system controls lights to fit the lighting schemes pre-defined by the user. There are a total of 36 requirements and 25 class diagrams in this system.

Our experiments focused on finding traces from requirements to UML class diagrams, as tracing to UML diagrams can be easily converted into traces to code by many commercial UML case tools [6].

3.2 Experiment Setup

The performance of the threshold selection algorithm is affected by the quality of the training set. Two issues that should be considered when selecting a training set are:

1) *The training set selection strategy*: if the training set contains non-representative data, the obtained information is at risk of "overfitting".
2) *The size of the training set*: when the size of the training set is too small, it might not be representative enough of the true characteristics of the whole data set. However, manually building the requirement traceability matrix for training set evaluation is extremely time consuming. An "optimal" size of the training set needs to be identified to find a balance that minimizes human effort yet optimizes performance.

The focus of the experimental study was to select an appropriate size for the training set, so that the threshold value selected from the training set could be effectively used to retrieve links from the entire document collection. The underlying hypothesis is that if the training set is well defined, then the percentage of correctly retrieved links from the entire document collection should be close to the target recall value Rt specified by the user during the threshold selection step.

In our experiments, the documents in the training set are randomly selected from the collection of searchable documents. Traceability matrices between requirements and class diagrams are available for all three examples. The matrices were created after long and tedious work by manually tracing requirements to class diagrams and were used to evaluate the performance of our approach.

In IBS system, there are 75 functional requirements and 172 UML class diagrams, resulting in12900 potential links. A training set is established by randomly choosing a subset of the class diagrams.

The threshold value on the training set is discovered using the thresholding strategy described above in chapter 3. The selected threshold value is then applied to the entire document and query collection to examine its overall performance. Recall and precision of the retrieval algorithm over the entire dataset are analyzed, and compared to the recall and precision values for the training set.

We conducted a Monte Carlo study to evaluate the performance of the threshold strategy if different sized training sets were selected. The study compared the performance of the threshold algorithm for training sets of sizes equal to 20%, 25%, 30%, 35%, 40% or 45% of the entire data set. For IBS dataset, the algorithms were run for three target recalls: (90%, 85%, 80%). Details about the results of 85% and 80% are reported in the appendix. The Monte Carlo study consisted of running 1,000 simulations. In each simulation, a training set of a given size is generated from the entire document collection and the threshold algorithm is run to find the optimal threshold for the given target recall. Each simulation is analyzed using the following metrics:

D_r: Difference between training set recall and overall recall value.

D_p: Difference between training set precision and overall precision value.

Both metrics measure how well the retrieval algorithm performs on the entire data set using the threshold obtained from the training set. As the training set size gets larger, we expect D_r and D_p to get smaller and closer to zero.

The performance of the threshold strategy can be analyzed by computing min, max, average and standard deviation for both metrics in each simulation study.

3.3 Results Analysis

3.3.1 Results for IBS database

Table 1 displays the statistical analysis results on IBS data with 1000 iterations and target recall of 90%. The size of the training set varies from 20% to 45% of the entire dataset. The results in table 1 show that the difference of performance between the training set and the entire dataset decreases as the size of the training set increases. When 20% of the data were selected as the training set, the average difference between recall between the two sets was 0.016502, with standard deviation equal to 0.044078; while for the training set containing 40% of the documents, the average recall difference decreased to 0.007115 and the standard deviation to 0.027537. Difference in precision between the training set and the entire dataset followed a similar trend.

Table 1: **Statistical analysis on simulation of IBS with target recall 90%**

	Training set size					
Measure	20%	25%	30%	35%	40%	45%
$Max(D_r)$	0.179225	0.157986	0.174107	0.148238	0.115276	0.099864
$Min(D_r)$	-0.06673	-0.06999	-0.06338	-0.05582	-0.05233	-0.04303
$\overline{D_r}$	0.016502	0.012281	0.009271	0.007822	0.007115	0.005258
$s(\overline{D_r})$	0.044078	0.039069	0.032235	0.0305	0.027537	0.023939
$Max(D_p)$	0.099003	0.110759	0.086076	0.090283	0.059268	0.049224
$Min(D_p)$	-0.09132	-0.07552	-0.06307	-0.05202	-0.05189	-0.04558
$\overline{D_p}$	0.002983	0.002661	0.002434	0.001846	0.002293	0.001073
$s(\overline{D_p})$	0.029561	0.025859	0.021855	0.019671	0.0175	0.015175

If the training set is well defined, the sample averages $\overline{D_r}$ and $\overline{D_p}$ should be close to 0 with standard deviations $s(\overline{D_r})$ and $s(\overline{D_p})$ as small as possible.

By examining the results on table 1, we find that for training sets containing about 30% of the documents, the sample average for the difference in recall is $\overline{D_r}$ =0.009271 with standard deviation equal to $s(\overline{D_r})$ =0.032235. This means that in about 95% of the simulated training sets, the values computed for D_r are in the interval (-0.051, 0.073). This is also displayed by the histogram in Figure 1. Our results show that in most simulated training sets, the overall recall on the entire data set is close to the target recall, and that in more extreme cases, overall recall is either 5% higher or 7% lower than target recall in the training set. For instance, if target recall is equal to 90%, the selected threshold value would allow retrieval of a percentage of true links varying between 83% and 95% of the entire collection. In a very few cases, a larger difference in retrieval performance was observed between training set and entire collection. At a 90% target recall, the worst result returned overall recall values 15% lower than the target.

The 95% of the simulation results have precision difference between (-0.038, 0.048). This indicates that on average the precisions in the training set and in the entire collection are very similar.

From a practical standpoint, we cannot accept a decrease in overall recall and precision larger than 15% of the corresponding values in the training set. For instance, with 95% of recall retrieved in the training set, the acceptable overall recall should be within the interval of [80%, 100%], with very few cases exhibiting recall less than 80%. Given this criteria, we determined that a training set of 30% of the entire dataset provides a reasonable ratio for IBS.

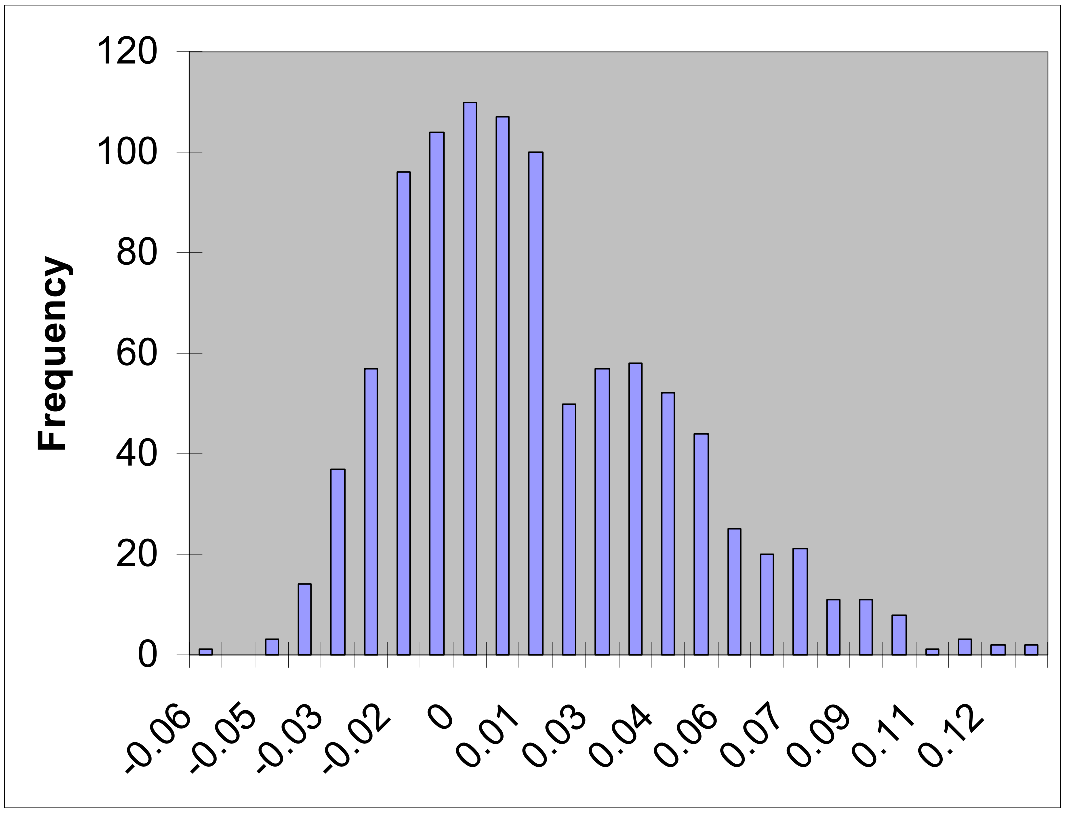

Figure 1: Distribution of recall difference from training set of size 30% to entire IBS dataset in 1000 iterations, with target recall 90%.

Given the target recall 90% and the optimal training set size of 30%, Figure 1 depicts the distribution of *Dr* computed in 1000 iterations. The histogram is right skewed. There are a few extreme cases where recall on the training set was more than 10% higher than the overall recall on the whole dataset. The histogram shows that in the worst case scenario, the threshold value would likely yield 78% overall recall.

Figure 2 shows the distribution of the difference in precision values computed from 1000 iterations. The distribution is symmetric and centered around 0. Given the target recall of 90%, the actual precision achieved on the entire dataset ranges from 16% to 38%, with an average of 21%.

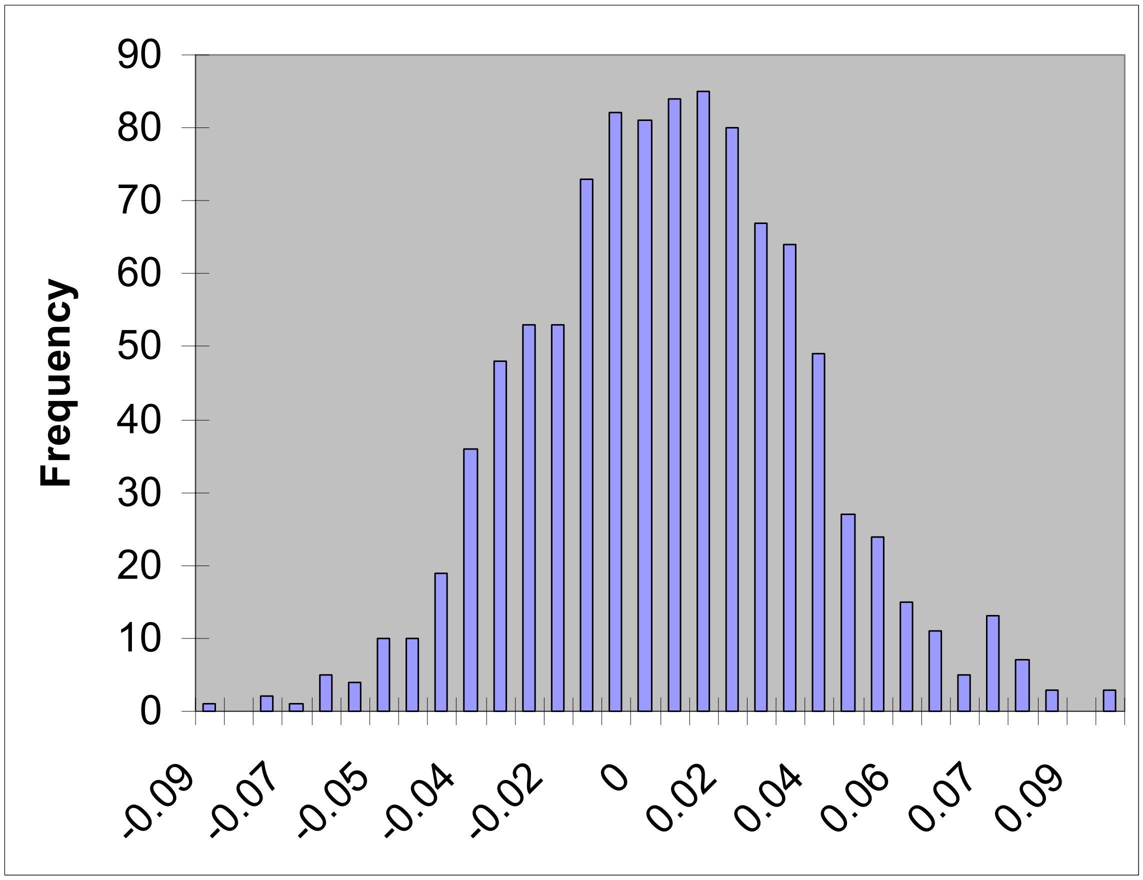

Figure 2: Distribution of precision difference from training set of size 30% to entire IBS dataset in 1000 iterations, with target recall 90%.

The results on IBS database indicate that if the threshold is learned from a training set containing about 30% of the documents, the automatic tracing algorithm performance is very good and high values of recall and precision can be achieved.

3.3.2 Results and Analysis on EBT and LC databases

The same simulation procedure was conducted on EBT and LC databases. Table 2 and 3 display the simulation results of 1000 iterations on EBT and LC data individually, with target recall of 90%.

Table 2: Statistical analysis on simulation of EBT with target recall 90%

	Training set size					
Measure	20%	25%	30%	35%	40%	45%
$Max(D_r)$	0.350556	0.32186	0.287138	0.241959	0.227797	0.205556
$Min(D_r)$	-0.15972	-0.375	-0.10859	-0.07054	-0.06143	-0.05665
$\overline{D_r}$	0.04661	0.03384	0.028322	0.018686	0.021698	0.016127
$s(\overline{D_r})$	0.080644	0.070283	0.059799	0.05252	0.046541	0.039303
$Max(D_p)$	0.306558	0.223684	0.149049	0.124844	0.127473	0.095776
$Min(D_p)$	-0.11173	-0.1177	-0.10523	-0.08141	-0.09072	-0.08773
$\overline{D_p}$	0.009469	0.007056	0.004632	0.002033	0.004104	0.0034
$s(\overline{D_p})$	0.05461	0.046492	0.038621	0.03256	0.030581	0.027698

From the results on table 2, we found that when the size of the training set is 30% of the entire data, $\overline{D_r}$ =0.028322, $s(\overline{D_r})$ =0.059799. This indicates that in 95% of the simulated runs the difference in recall values between the training set and the entire document collection was in (-0.091, 0.148). For the difference in precision, the 95% interval is (-0.072, 0.081).

Given the same acceptable departure of 15%, the results in EBT indicate 30% could also be a suitable training set size.

Table 3: Statistical analysis on simulation of LC with target recall 90%

	Training set size					
Measure	20%	25%	30%	35%	40%	45%
$Max(D_r)$	0.41685	0.34777	0.264692	0.348571	0.238761	0.224407
$Min(D_r)$	-0.2674	-0.18482	-0.1453	-0.13936	-0.09032	-0.09196
$\overline{D_r}$	0.052201	0.036743	0.025078	0.024504	0.016245	0.012874
$s(\overline{D_r})$	0.092938	0.073873	0.058479	0.057908	0.043873	0.03913
$Max(D_p)$	0.396825	0.379683	0.321895	0.314099	0.239146	0.207299
$Min(D_p)$	-0.21522	-0.19164	-0.16488	-0.16278	-0.13233	-0.11009
$\overline{D_p}$	0.029133	0.021228	0.020395	0.017038	0.011278	0.008241
$s(\overline{D_p})$	0.110705	0.090345	0.077402	0.072696	0.059367	0.050239

The LC database is significantly smaller with only 850 possible pairs (document, query) to be linked. By analyzing table 3 we observed that for a training set containing 30% of the documents the difference in overall recall and training set recall is small on overage. In about 95% of the simulations, the overall recall decreases at most by 15%, while the overall precision decreases at most by 17%.

The analysis above shows that a training set that contains about 30% of the document collection can be used to effectively identify a threshold value for the entire document collection. When the threshold value for 90% recall is used in the trace retrieval algorithm on the entire document collection, the retrieval algorithm performance is close to 90% recall, in the worst cases 83% correct links are retrieved in IBS, 76% in EBT and LC.

4. Confidence Scores

Confidence scores have been defined to measure our degree of belief on the correctness of the results returned by the automatic trace retrieval algorithm. Confidence scores range from 0 to 100%, and their purpose is to provide a meaningful reference to the user when presenting the retrieval results. For retrieved links, confidence values suggest how much confidence we have that the links are indeed relevant; while for rejected links, they suggest how confident we are that they are truly not related.

In the requirements traceability domain, tracing tools should favor high recall rather than precision in order to retrieve as many true links as possible. This is accomplished through setting the threshold at a relatively low value. Previous research in this area shows that with approximately 90% recall, precision rages from 10% to 40% [1,2,3,4]. So for retrieved links, especially those just slightly above the threshold, we will have lower confidence that they are true links. On the other hand, we will have much stronger confidence that rejected links are truly irrelevant.

Candidate links with large probability scores will have higher confidence values. This observation is based on our experimental studies that indicate that candidate links with large probability scores are more likely to be true links. However, probability scores should not be the only criterion for defining confidence levels. During the experiments on the three databases, we noticed that the distribution of correct links should be taken into account when we compute confidence levels. Figure 3 shows the distribution of the probability scores for true-links vs. non-links for the IBS dataset. The graph indicates that the majority of non-links are found in the very low probability region, while the majority of true Rt links are found in higher probability value regions [6]. Therefore, high confidence can be assigned to links with high probability values. Similarly, we can assume with high confidence that document-query pairs with very low probability values should not be linked.

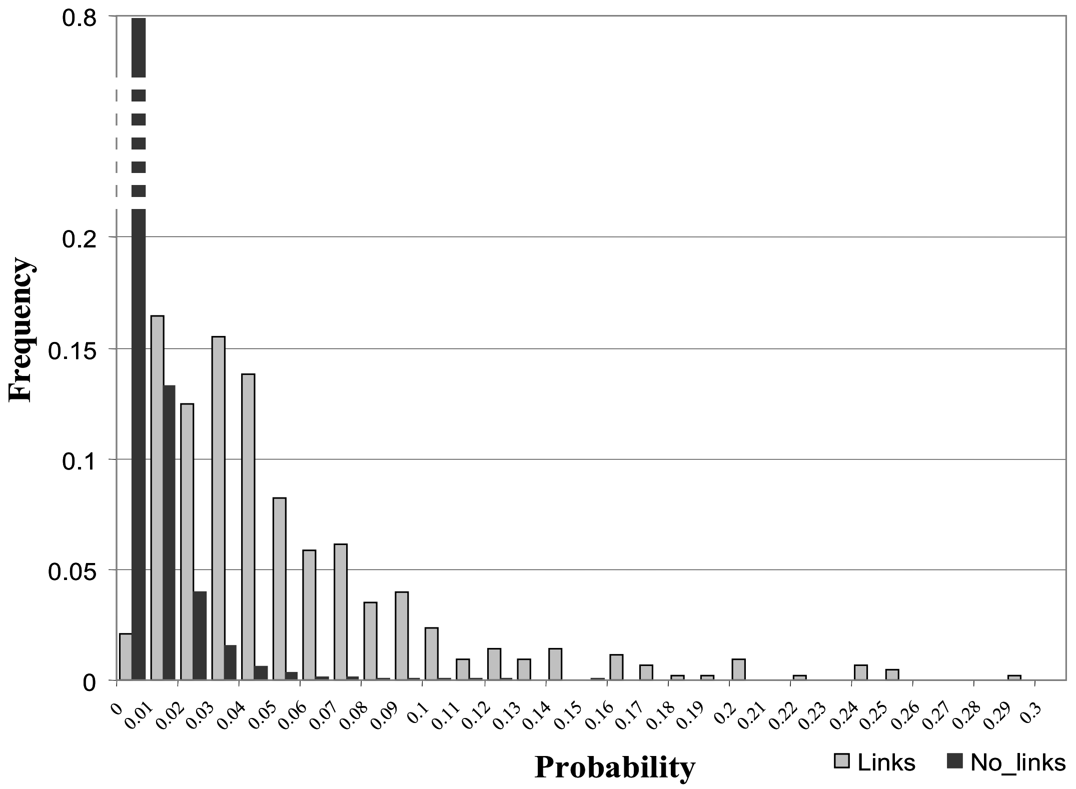

Figure 3: Probability distribution of links and non-links

In the region in between, true-links and non-links are mostly cross-distributed, making it difficult to discern true and non-true links. While the increase of probability value strengthens confidence level, the link distribution should also be taken into account to indicate the degree of change on confidence level in a certain region. Candidate links in the region with higher true-links distribution should have confidence values that increase at a faster rate than candidate links with higher probability scores but in a region with scarce true links. In other words, in a region where most candidate links are true-links, the confidence levels for candidate links in this region will increase very fast.

4.1 Segmented Confidence Scores

We propose a segmentation method for defining confidence scores based on the links distribution on training set.

Let Pi ($0 \leq P_i \leq P_{max}$, $i=0,1,\ldots,m$) be the probability value for a candidate link in the training set. Suppose the training set consists of m documents. The value P_{max} represents the candidate link that has the highest probability value in the training set. $C(P_i)$ be the confidence score attached to the link with probability value P_i.

Let X_e be the point with less than 5% confidence level, where $C(X_e)<0.05$. The lowest confidence point X_e needs to be discovered prior to the segmentation on the training set. X_e should have very low probability score P_e, so that we have almost very high confidence that the number of true links in the area around P_e is very scarce. In our experiments, we define $C(X_e)=0$.

The 100% confidence score is assigned to the candidate link that has the highest probability value P_{max}.

The region between the P_e and P_{max} is then divided into K segments (1<K<m) according to the probability score distribution of the true links.

In a segment k (1≤k≤K), the density of true links μ(k) is defined as:

$$\mu(k) = \frac{\text{number of true links in segment}}{\text{number of candidate links in segment}}$$

The measure μ(k) can be regarded as the local precision metric relative to the segment.

Given a segment k where the average probability value is $\overline{P}_k$, we define the confidence score: C($\overline{P}_k$) = μ(k).

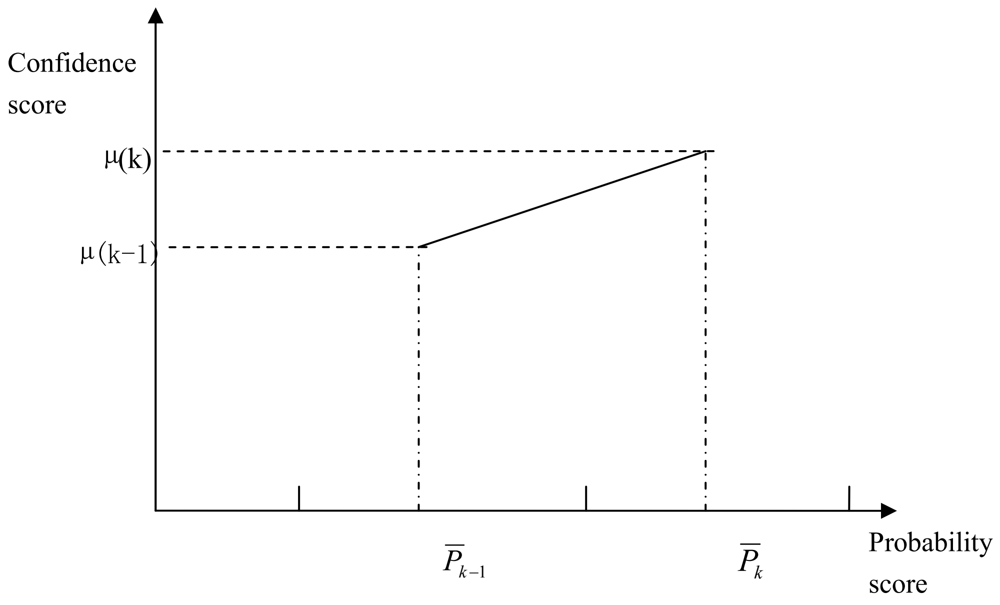

Figure 4: Confidence definition in two neighboring segments

As showed in figure 4, in two neighboring segments k-1 and k for 1<k<K where their average probability values are $\overline{P}_{k-1}$ and $\overline{P}_k$ respectively, a candidate link with probability value P ($\overline{P}_{k-1}$≤P≤$\overline{P}_k$) has confidence score computed as follows:

$$C(P) = \left[\mu(k-1) + \frac{\mu(k) - \mu(k-1)}{\overline{P}_k - \overline{P}_{k-1}} \times (P - \overline{P}_{k-1}) \right] \times 100\%$$

In the first segment, the confidence score of a candidate link with the probability value P (P_e≤P≤$\overline{P}_1$) is defined as:

$$C(P) = \left[\frac{\mu(1)}{\overline{P}_1} \times (P - P_e) \right] \times 100\%$$

Accordingly, in the last segment K, the confidence score of a candidate link with the probability value P ($\overline{P}_k$ ≤P≤P_{max}) is defined as:

$$C(P) = \left[1 - \frac{1-\mu(k)}{P_{\max} - \overline{P}_k} \times (P - \overline{P}_k)\right] \times 100\%$$

For pairs with 0 probability value, the confidence level for the rejected links is set equal to 100%, meaning we have 100% confidence they are non-links. In the area where probability values are lower than X_e, a pair with the probability value P ($0 \le P < X_e$) has confidence score given by:

$$C(P) = \left[1 - \frac{1-\mu}{\overline{P}} \times (P - \overline{P})\right] \times 100\%$$

where μ, the density of non-links in the region, is defined as

$\mu = \dfrac{\text{number of non - links in the region}}{\text{number of all pairs in the region}}$, and $\overline{P}$ denotes the average probability value in this region.

4.2 Experiments on segmentation methods

4.2.1 Results on IBS databases

The segmentation method was applied to each database to analyze its effectiveness in computing meaningful confidence values. Based on the conclusion drawn from the study on training sets, 30% of the entire data set was randomly chosen as the training set.

In one randomly generated training set of IBS with 4200 pairs of candidate links, we found that P_{max} = 0.25, so C(0.25) = 100%. The value of 0.003 was chosen as X_e, and C(0.003) = 0%. This was based on the observation that almost no true links have probability scores lower than 0.003 in the training set. For any candidate links with probability values lower than 0.003, we have strong confidence that they are non-links.

All candidate links were divided into 6 segments according to the links distribution. Table 4 displays the segments on this training set, the corresponding true links density μ (non-links density in the case of non-link segment 6) and the average probability $\overline{P}$.

Table 4: Segmentation on IBS training set

Segment		μ	$\overline{P}$
1	>0.08	0.761	0.116
2	0.05-0.08	0.571	0.063
3	0.03-0.05	0.453	0.038
4	0.01-0.03	0.163	0.018
5	0.003-0.01	0.060	0.006
6	0-0.003	0.968	0.002

As we defined earlier, C($\overline{P}_k$)=μ(k). So in segment 1, C(0.116) = 0.761; in segment 2, C(0.063) = 0.571, and so on. Table 5 displays the definition of confidence level $C(P_i)$ in each segment.

Table 5: Confidence scores definition in each segment of IBS training set.

P_i	$C(P_i)$
⩾0.116	[0.761+1.78×(Pi-0.116)] ×100%
[0.063, 0.116)	[0.571+3.58×(Pi-0.063)] ×100%
[0.038, 0.063)	[0.453+4.72 ×(Pi-0.038)] ×100%
[0.018, 0.138)	[0.162+14.5 ×(Pi-0.018)] ×100%
[0.006, 0.018)	[0.060+8.58×(Pi-0.006)] ×100%
[0.003, 0.006)	[20×(Pi-0.003)]×100%
<0.003	[1-16×Pi] ×100%

Figure 5 illustrates the curve displaying the relationship between the probability value of a candidate link and its confidence score.

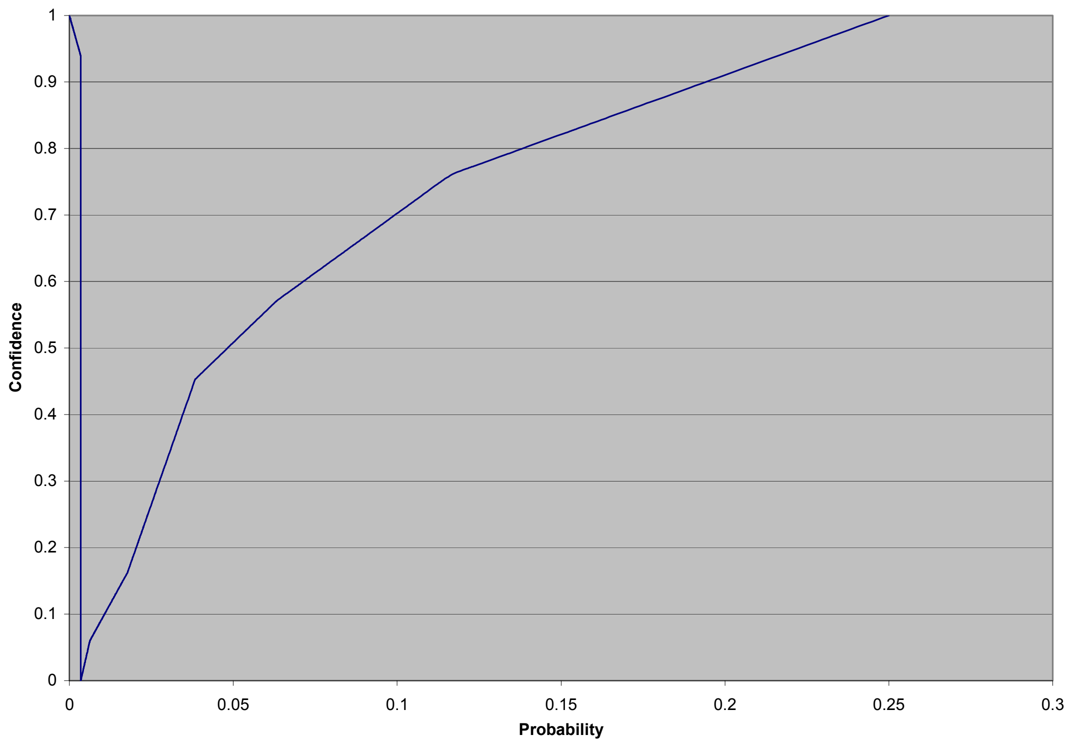

Figure 5: Confidence scores on IBS training set

4.2.2 Results and Analysis on EBT and LC databases

The same method was also applied to randomly selected training sets of size equal to 30% for the EBT and LC databases.

Table 6 illustrates the segmentation on one EBT training set with P_{max}=0.054. The probability value of 0.0006 was chosen as the 0 confidence point X_e and then all candidate links were divided into 5 segments.

Table 6: Segmentation on EBT training set

Segment		μ	$\overline{P}$
1	>0.04	0.75	0.047
2	0.025-0.04	0.4	0.031
3	0.01-0.025	0.304	0.016
4	0.0006-0.01	0.112	0.004
5	0-0.0006	0.996	0.0003

The confidence definition $C(P_i)$ for a candidate link with probability score of P_i was given in table 7.

Table 7: Confidence scores definition in each segment of EBT training set

Pi	C(Pi)
⩾0.047	[0.75+35.7(Pi-0.047)] ×100%
[0.031, 0.047)	[0.4+23.3×(Pi-0.031)] ×100%
[0.016, 0.031)	[0.304+6.4×(Pi-0.016)] ×100%
[0.004, 0.016)	[0.112+16×(Pi-0.004)] ×100%
[0.0006, 0.004)	[32.9×(Pi-0.0006)] ×100%
<0.0006	[1-13.3×Pi] ×100%

Figure 6 depicts how confidence levels change with probability values.

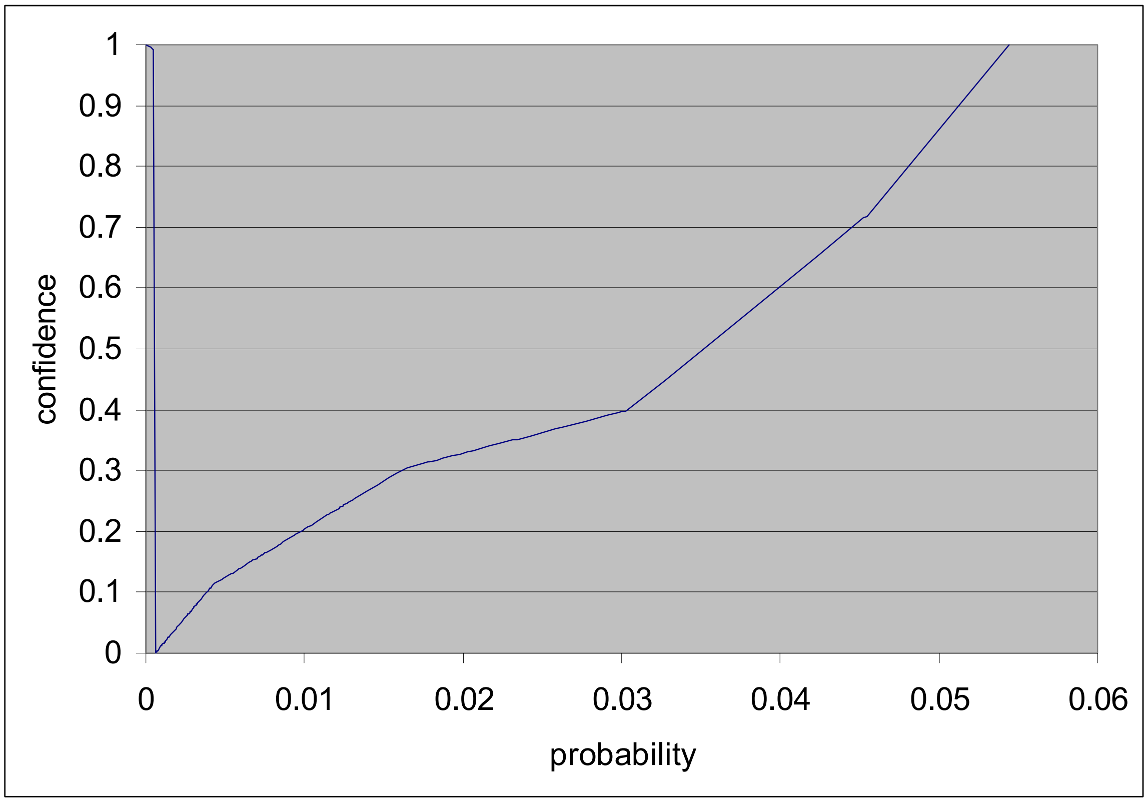

Figure 6: Confidence scores on EBT training set

Table 8, table 9 and figure 7 display the results of segmentation method on one LC training set with $P_{max} = 0.25$ and probability value of 0.0067 chosen as X_e.

Table 8: Segmentation on LC training set

Segment		μ	$\overline{P}$
1	>0.07	0.75	0.162
2	0.045-0.07	0.615	0.055
3	0.0067-0.045	0.328	0.018
4	0-0.0067	0.963	0.0026

Table 9: Confidence scores definition in each segment of LC training set.

Pi	C(Pi)
⩾0.162	[0.75+2.84(Pi-0.162)] ×100%
[0.055, 0.162)	[0.615+1.26×(Pi-0.055)] ×100%
[0.018, 0.055)	[0.328+7.76×(Pi-0.018)] ×100%
[0.0067, 0.018)	[29.0×(Pi-0.0067)] ×100%
<0.0067	[1-14.2×Pi] ×100%

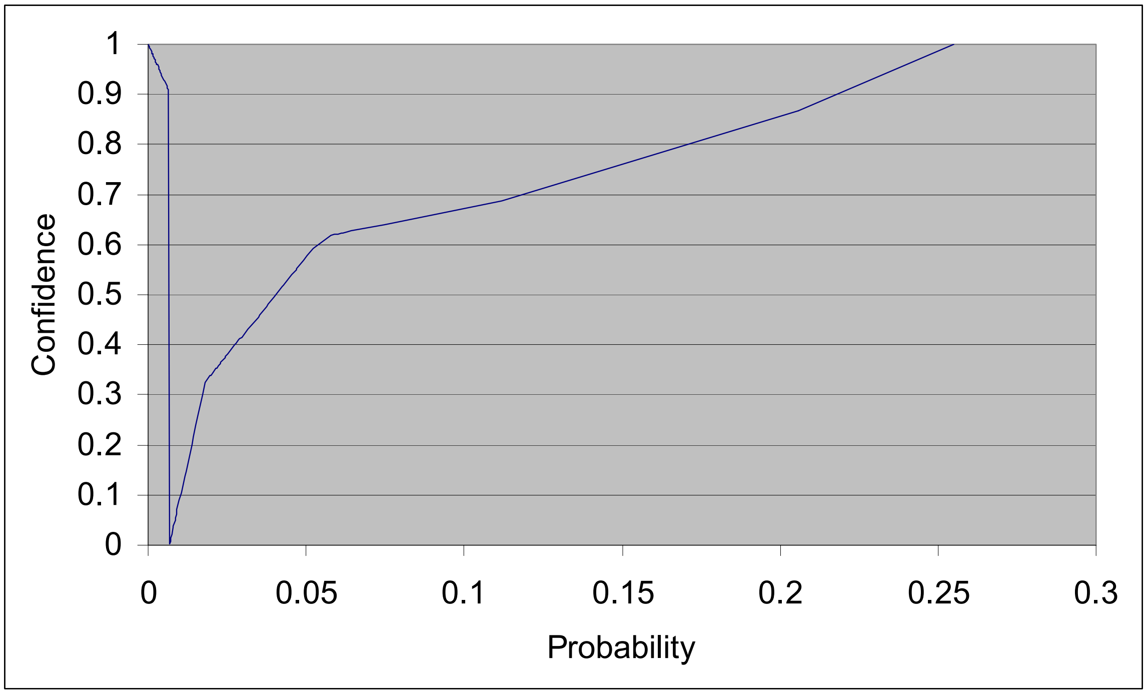

Figure 7: Confidence scores on LC training set

5. Discussion

The experiments on training set identification demonstrate the effectiveness of using an appropriate sized training set to identify optimal threshold values. The results of our analysis show that 30% of the document collection can be considered a suitable size for the training set, as in most cases, the training set is able to capture the characteristic of the data in the whole set. For smaller databases, the training set needs to contain more data in order to include representative characteristics of the whole set.

Based on the knowledge learned from a training set, the method for defining confidence scores provides the user with a measure of the degree of confidence that an automatically retrieved trace is correct or not. This method considers both probability scores and link distribution as contributing factors for defining confidence levels. We believe this is beneficial for improving the utility of the tracing tools.

We are planning on conducting more experiments on other databases to verify the methods we proposed and the conclusion drawn from our previous study.

Acknowledgments

The work described in this paper was partially funded by NSF grant CCR-0306303.

Reference

[1] G.. Antoniol, G.. Canfora, G.. Casazza, A. De Lucia and E. Merlo. “Recovering Traceability Links between Code and Documentation”, *IEEE Transactions on Software Engineering*. Vol. 28(10), 2002, pp. 970-983.

[2] J. Huffman Hayes, A. Dekhtyar, and J. Osborne, “Improving Requirements Tracing via Information Retrieval”, IEEE International Requirements Engineering Conference, Monterey, CA, Sept. 2003, pp.138-150.

[3] J. Huffman-Hayes, A. Dekhtyar, S. Karthikeyan Sundaram, S. Howard, “Helping Analysts Trace Requirements: An Objective Look”, International Requirements Engineering Conference, Kyoto, Japan, pp.249-259

[4] R. Settimi, J. Cleland-Huang, O. BenKhadra, J. Mody, W. Lukasik, and C. DePalma, C., “Supporting Change in Evolving Software Systems through Dynamic Traces to UML”, IEEE International Workshop on Principles of Software Evolution, Kyoto, Japan, Sept. 2004, pp.49-54.

[5] J. Cleland-Huang, R. Settimi, O. BenKhadra, E. Berezhanskaya, S. Christina, ”Goal-Centric Traceability for Managing Non-Functional Requirements”, International Conference on Software Engineering, St. Louis, USA, May 2005.

[6] J. Cleland-Huang, R. Settimi, C. Duan, X. Zou, “Utilizing Supporting Evidence to Improve Dynamic Requirements Traceability”, to be appear in the 13th IEEE International Requirements Engineering Conference, Paris, France, Aug 2005.

[7] S.K.M. Wong, and Y.Y. Yao, “A probabilistic inference model for information retrieval” *Information Systems*, Vol. 16(3), pp.301-321.

[8] S.K.M. Wong, and Y.Y. Yao, “On Modeling Information Retrieval with Probabilistic Inference”, *ACM Transactions on Information Sys,* Vol. 3(1), pp.38-68.

[9] X. Zou, R. Settimi, J. Cleland-Huang,, C. Duan, "Thresholding Strategy in Requirements Trace Retrieval", CTI Research Symposium, Chicago, 2004, pp.100-103.

[10] W.B. Frakes., and R. Baeza-Yates., *Information retrieval: Data structures and*

Algorithms, Prentice-Hall, Englewood Cliffs, NJ, 1992.

Appendix

The following are the results of in IBS database with target recall of 85% and 80% respectively.

Table 10 analyzes the simulation results for training sets of size equal to 20%, 25%, 30%, 35%, 40% and 45% of the entire IBS dataset with target recall of 85%. In table 11, the analysis with target recall of 85% is presented. The results are similar with that of target recall of 90% presented in section 3.2.1.

Figure 8, 9, 10 and 11 illustrate the distribution of the recall/precision difference for training set sized at 30% of the entire dataset with two different target recalls.

Table 10: **Statistical analysis on simulation of IBS with target recall 85%**

	Training set size					
Measure	20%	25%	30%	35%	40%	45%
$Max(D_r)$	0.236607	0.188502	0.144818	0.152232	0.15776	0.14301
$Min(D_r)$	-0.10069	-0.09539	-0.08246	-0.07718	-0.07712	-0.06133
$\overline{D_r}$	0.016731	0.011002	0.0099	0.091769	0.006189	0.005478
$s(\overline{D_r})$	0.057853	0.050236	0.0447	-0.06113	0.036909	0.031758
$Max(D_p)$	0.170368	0.1471	0.094527	0.008123	0.077103	0.079783
$Min(D_p)$	-0.08849	-0.08374	-0.07248	0.039936	-0.05483	-0.05671
$\overline{D_p}$	0.00422	0.003006	0.002148	0.00238	0.000824	0.000938
$s(\overline{D_p})$	0.035931	0.03	0.026959	0.02217	0.020125	0.018338

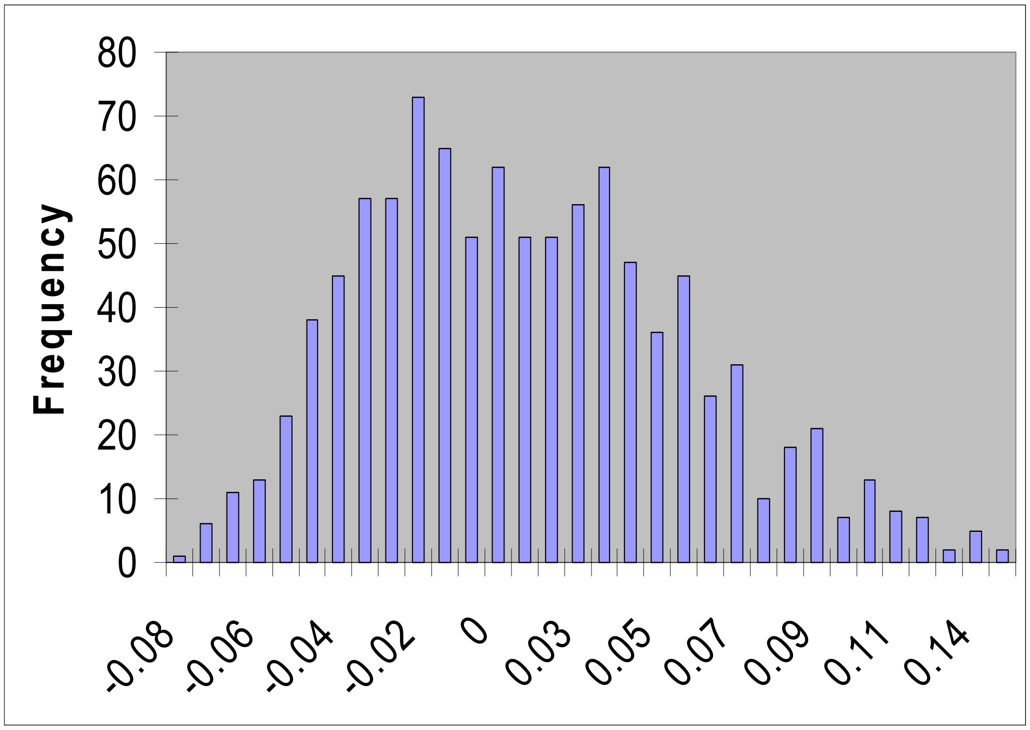

Figure 8: Distribution of recall difference from training set of size 30% to entire IBS dataset in 1000 iterations, with target recall 85%.

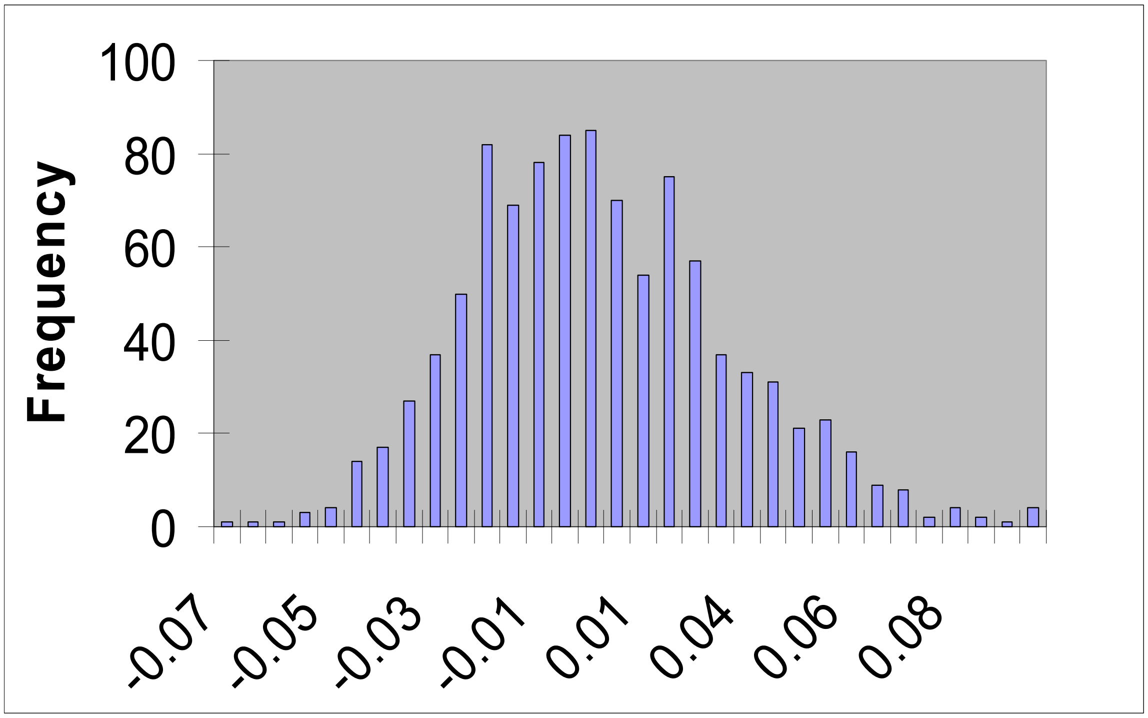

Figure 9: Distribution of precision difference from training set of size 30% to entire IBS dataset in 1000 iterations, with target recall 85%.

Table 11: Statistical analysis on simulation of IBS with target recall 80%

	Training set size					
Measure	20%	25%	30%	35%	40%	45%
$Max(D_r)$	0.245387	0.222305	0.15997	0.163089	0.166454	0.11932
$Min(D_r)$	-0.14356	-0.11674	-0.11346	-0.10466	-0.10057	-0.09966
$\overline{D_r}$	0.01626	0.008754	0.00857	0.007093	0.006118	0.003212
$s(\overline{D_r})$	0.067186	0.055435	0.049096	0.04572	0.042041	0.035816
$Max(D_p)$	0.14619	0.153549	0.105772	0.121714	0.120469	0.067764
$Min(D_p)$	-0.12753	-0.09327	-0.08417	-0.08513	-0.05742	-0.06293
$\overline{D_p}$	0.005583	0.003623	0.003817	0.002173	0.002411	0.002361
$s(\overline{D_p})$	0.044081	0.036363	0.030739	0.027535	0.024761	0.020943

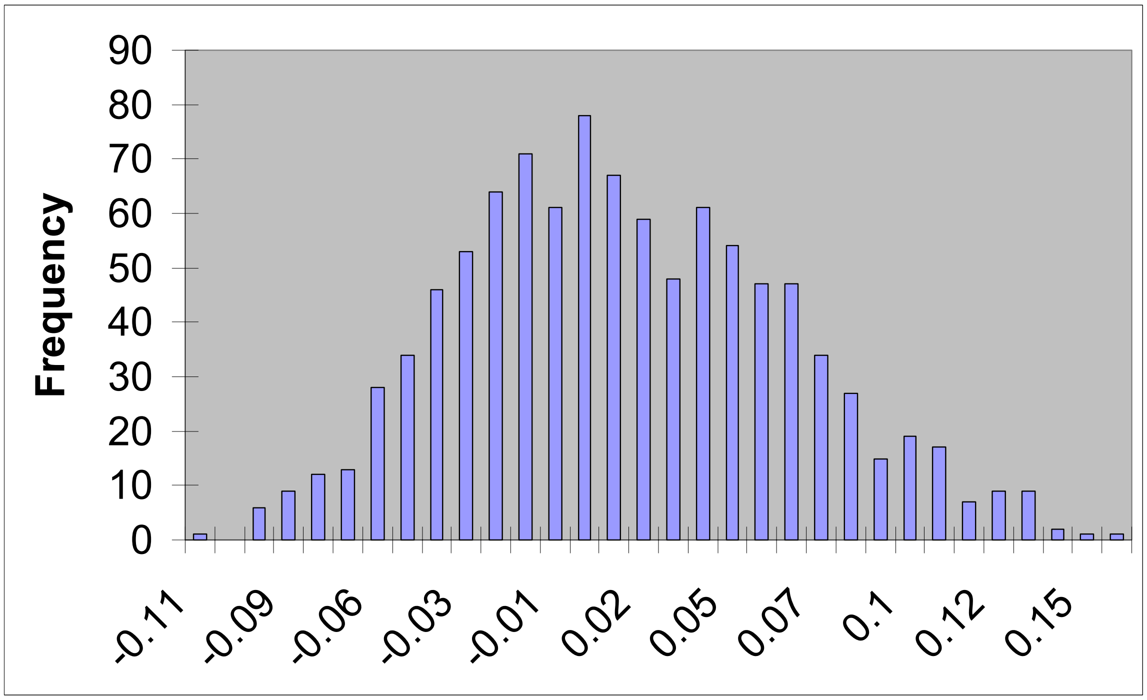

Figure 10: Distribution of recall difference from training set of size 30% to entire IBS dataset in 1000 iterations, with target recall 80%.

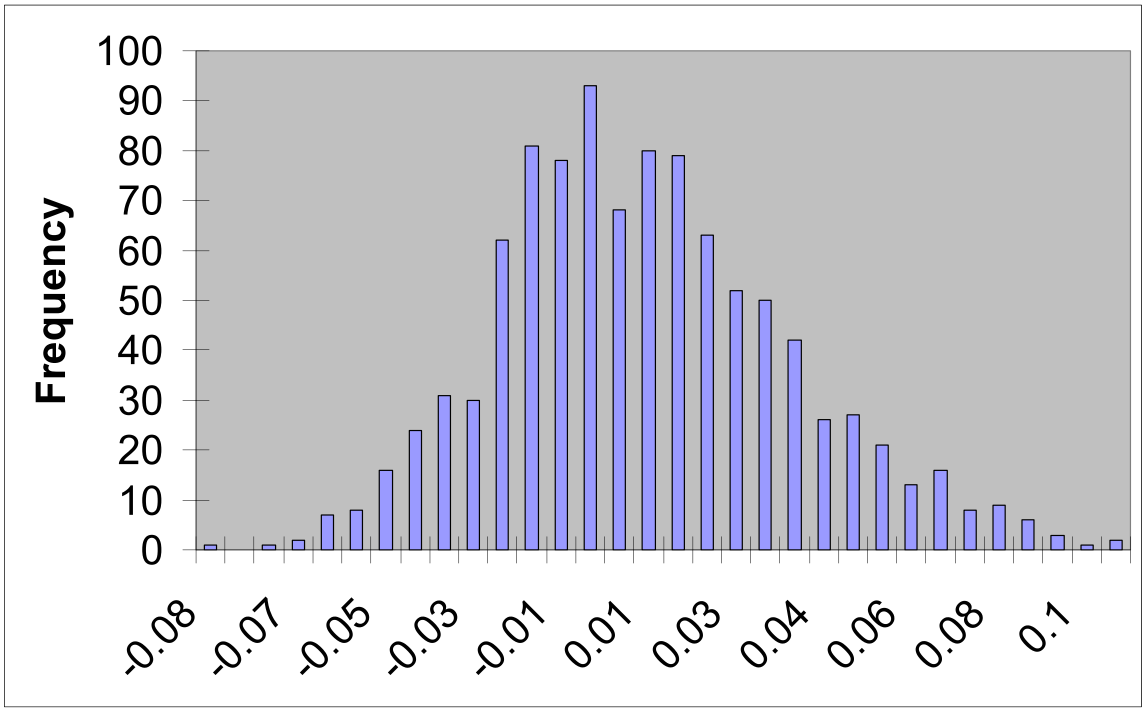

Figure 11: Distribution of precision difference from training set of size 30% to entire IBS dataset in 1000 iterations, with target recall 80%