RJ 10234 (A0203-001) March 1, 2002
Computer Science

IBM Research Report

Evaluating Knowledge Base Relevancy

Scott Spangler
IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099

Evaluating Knowledge Base Relevancy

Scott Spangler
IBM Almaden Research Center
650 Harry Road, San Jose, CA 95120-6099
408-927-2887
email: spangles@us.ibm.com

ABSTRACT

As the web and e-business have proliferated, the practice of using customer facing knowledge bases to augment customer service and support operations has increased. This can be a very efficient, scalable and cost effective way to share knowledge. Because of the costs involved in developing knowledge bases, we would prefer to reuse knowledge base content rather than redevelop it. The problem is, how to determine the relevancy of an existing knowledge base to a new problem domain, or to determine which of a set of existing knowledge bases is most relevant.

In this paper, we describe an implementation of an algorithm and methodology for comparing the relevancy of different knowledge bases to a set of problem tickets. Using distance metrics that have been applied in the field of Text Clustering, we systematically compare the text of the problem domain to the text of each knowledge base. We then allow an expert in the domain area to check the validity of the problem/solution matches and determine where to draw the line between the sets of solved and unsolved problem tickets. Our claim is that this approach makes it possible to objectively compare thousands of problems against thousands of solutions in a matter of hours, to determine an approximate percent coverage of the solutions when applied to the problem domain. We have implemented our approach, and we present the results of performing knowledge base relevancy evaluation on a computer helpdesk problem ticket set compared to different knowledge base solution sets.

1. INTRODUCTION

In this paper, we focus on analyzing text produced by service and support helpdesks and corresponding knowledge bases. Human helpdesk operation is very labor intensive and therefore expensive. Developing knowledge bases for these helpdesks to automate the helpdesk process is also very time consuming. Consequently, wherever possible we would like to reuse generic helpdesk knowledge or apply the knowledge developed at one helpdesk to another. The thesis of this paper is that there is much value derived from using our algorithms and methodology to help determine which of a set of candidate knowledge bases is best suited to address the problems most often found at a particular helpdesk. Our approach uses text mining distance metrics to summarize the content of problem tickets and compare this content to candidate solutions. Our algorithms then help the expert rapidly make a determination as to what percentage of the problem space is addressed by the candidate knowledge base.

In this paper, we apply our algorithm specifically to the domain of the computer helpdesk support center. A typical text document (known as a *problem ticket*) from such a data set consists of a brief description by the helpdesk agent of the exchanges between an end user and an expert helpdesk advisor, for example:

```
1836853 User calling in
with WORD BASIC error when
opening files in word. Had
user delete NORMAL.DOT and
had her reenter Word, she
was fine at that point.
00:04:17     ducar     May
2:07:05:656PM
```

Problem tickets may be comprised only of a single symptom and resolution pair as in the above example, or they may span multiple questions, symptoms, answers, attempted fixes, and resolutions--all pertaining to the same basic issue. Problem tickets are *opened* when the user makes the first call to the helpdesk and *closed* when all user problems documented in the first call are finally resolved in some way. Helpdesk operators enter problem tickets directly into the database. Spelling, grammar and punctuation are inconsistent. The style is terse and the vocabulary very specialized.

Knowledge bases are essentially a combination of content and inference [6]. For the purposes of this paper we ignore the underlying inferencing mechanism of the knowledge base and focus solely on the knowledge base content, in the form of unstructured text. Our solution web site at Almaden contains over 2000 solution documents. A typical example of the relevant text taken from a solution on this website is the following:

```
To request an Agora account,
fill in the form on this page.
You will receive e-mail
addressed to your Lotus Notes
userid with Agora account and
further instructions. If you
would like to arrange for
AIX/Agora to be installed on
your RS/6000 workstation, fill
in the installation request form
on this page.
```

Given this format for problems and solutions, our task is to compare them objectively to determine approximately what percentage of the problems are addressed by the solutions. We describe our approach to this problem in the remaining sections of this paper. In Section 2, we describe how we use text clustering techniques to represent and compare text content and also to narrow the scope of the problem tickets to only those that are suitable for automated resolution. In Section 3, we describe how we systematically compare representative problem tickets to a set of solution documents. In Section 4, we describe an interactive tool that allows a domain expert to determine the percent coverage of a given knowledge base solution set. In Section 5 we validate our approach on an artificial problem ticket set where the exact percent coverage of each solution set is known. Finally, in Section 6, we discuss how our application has been used to compare two real world knowledge bases and draw some conclusions about the overall usefulness of this approach.

2. TEXT ANALYSIS & CLUSTERING

In general, not all problem tickets in a given domain are amenable to automated knowledge base solutions. For example, at a computer help desk a broken disk drive may need to be replaced, requiring an on-site visit by a repair person. The first step, therefore, in automation of this task is to understand it, and one of the first steps in understanding is to segment examples into meaningful categories [1]. In general, this is not a simple problem, because the straightforward solution of having an expert read each problem ticket and decide if it is solvable is far too time consuming.

Instead we employed standard text clustering techniques [5]. We used tabulation of word occurrences in each problem ticket as the basis for or feature space. After eliminating common stop words and (high- and low-frequency) non-content-bearing words, we represented the text data set as a vector space model, that is, we represented each problem ticket as a vector of certain weighted frequencies of the remaining words [7]. A stemming algorithm [3] is used to produce a "synonym table" which allows us to treat various similar word forms as a single term (e.g. print, printer, printing). To create a document vector, we used the **txn** weighting scheme [8]. This scheme emphasizes words with high frequency in a document, and normalizes each document vector to have unit Euclidean norm. For example, if a document were the sentence, "We have no bananas, we have no bananas today," and the dictionary consisted of only two terms, *bananas* and *today*, then the unnormalized document vector would be [2 1] (to indicate two *bananas* and one $\left[{}^{2}\!/\!\sqrt{5}, {}^{1}\!/\!\sqrt{5}\right]$ normalized version would be: .

Once we represent each document as such a vector of unit Euclidean norm, we use the k-means text clustering algorithm [2] [4] to obtain an initial clustering of the data. The distance metric employed is the cosine similarity metric. Thus two points are considered identical if the cosine of the angle between them is 1.0 and considered most dissimilar if the cosine of this angle is 0.0.

The k-means approach starts by finding the nearest of the k problem ticket seeds to each problem ticket example (using the cosine distance metric). The expert chooses the value of k (the number of clusters). A typical value is usually somewhere around 30, a number of clusters that can be easily examined by the expert in less than an hour. Each problem ticket is then said to "belong" to the cluster corresponding to that nearest seed. For each cluster, a centroid is then calculated, which is the vector space mean of the examples contained in that cluster. In the next iteration, each problem ticket is compared to every centroid and each problem ticket is moved to the cluster of the nearest centroid. The centroids are then recalculated and the process continues until an iteration does not result in any cluster membership changes from the previous iteration. The result of the clustering is the set of problem ticket classes. Each class is given a name based on the dictionary term or terms that dominate the class (i.e. are contained by most of the class examples).

Next a domain expert selects the set of problem classes that are amenable to knowledge based solutions. In most cases the name of the problem class is enough to identify to the expert whether or not the class of problems in amenable to solution or not. In a few cases, typical examples from a class may need to be examined to identify the common theme of the class. In any case, once the expert selects the classes that are solvable, all the members of the selected problem classes are placed in a pool of candidate examples.

3. COMPARING PROBLEMS AND SOLUTIONS

This section discusses our method for objectively comparing problems and solutions. The goal in this comparison is to find out what percentage of problems is addressed by a given solution set. The first step is to find the closest solution to each problem and the second step is to determine whether or not that solution is relevant enough to address the problem in question. All of this needs to be done with minimal expert intervention, though we recognize that some expert involvement is necessary since there is no practical way for a machine to reason about whether or not a given problem ticket is actually solved by a given solution.

We begin by measuring the cosine distance between every solution document and a representative sample of the problem tickets. Sampling of problem tickets is often necessary to make the problem tractable, since many helpdesks have hundreds of thousands of problem tickets in their repositories. However, knowledge base solution sets are not sampled in our approach. Note that in most cases the solution documents will be far less numerous than the problem tickets.

To directly compare problem and solution documents, we first reduce each solution document to a vector of word occurrence counts, using the same dictionary and synonym table that was used in clustering the problem tickets. We then convert these to unit vectors as we did with the problem tickets. The *most similar* document in the solution set to a given problem is defined to be the document whose word occurrence vector is nearest to the centroid of the given cluster (using the cosine distance metric).

$$\cos(X,Y) = \frac{X \bullet Y}{\| X \| \cdot \| Y \|}$$

Equation 1: Where *X* is the problem set document vector, and *Y* is the solution set document vector.

Note that in the equation for cosine distance the denominator simplifies to 1.0 whenever the input vectors are unit Euclidean norms, as is the case here. For each problem document we then calculate a *g score* which is the largest (i.e. nearest) cosine distance between that problem document and the set of solutions.

$$g(X) = \underset{y \in solutions}{Max} (\cos(X,y))$$

Equation 2: The g-score is the distance of the problem set document to nearest solution document

Note that we use the maximum cosine distance to select the "most similar" document because cosine distance returns a value of 1.0 for identical documents and 0.0 for completely distinct documents. As the g-score increases, therefore, we expect the likelihood of a matching solution document to the problem to also increase. A low g-score indicates that no matching solution document is present (thus a low g-score indicates a large "knowledge gap" [9]).

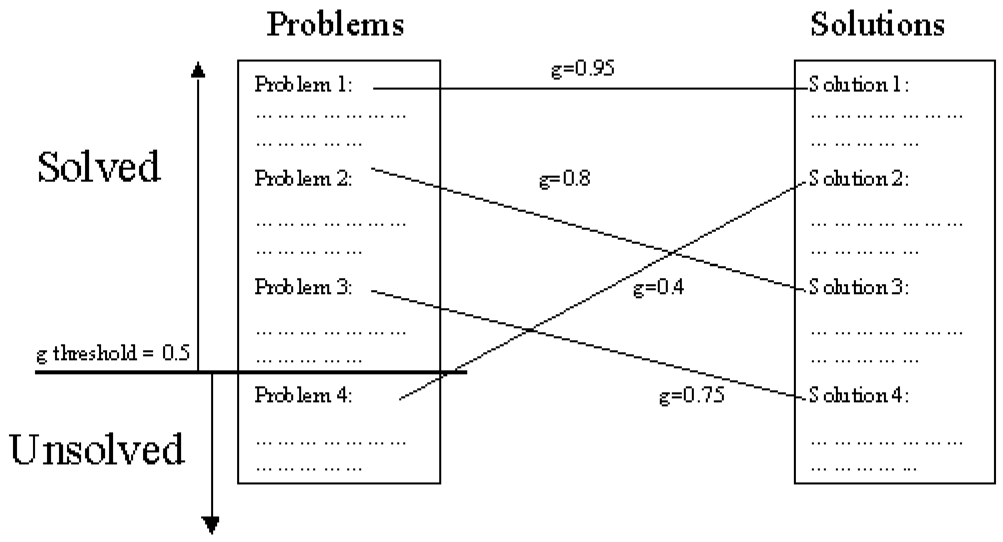
There are some inherent assumptions underlying this approach. The principle assumption is that solution documents contain many of the same terms as the problems that they intend to solve. While there are applications and domains where this may not be the case, we have observed across a broad set of helpdesks and knowledge bases that this assumption holds true. We have found that in practice most helpdesk solution documents contain at least a brief summarization of the problem that they propose to solve, and this summary invariably will contain words that are similar to those used in the problem ticket descriptions. In addition, the problem tickets themselves will often contain brief descriptions of what was done to resolve the issue. In many cases, the terms in this solution description match those used in the solution documentation. Therefore, we believe this assumption is valid and will hold for many other domains as well.

Once a g-score has been calculated for each problem in the problem set, the problems are sorted in order of decreasing g-score and displayed to the expert in that order, along with a set of solutions for each problem which most nearly match the problems word content. These solutions are also sorted in order of g-score. The problems and their corresponding solutions are then presented to the expert for further evaluation, as described in the next section.

4. KNOWLEDGE BASE EVALUATION

The sorted problem tickets and g-scores calculated above serve as the basis for our knowledge base evaluation. Our goal in this step is to approximately determine the percentage of problem tickets that are addressed by a solution. Our assumption in determining this percentage is that those problems that are solved occur before those that are not solved in g-score sorted order. Our strategy then, is to determine just where in the sorted order we can best draw the line between solved and unsolved problems. Once we draw this line, we can approximate the percent solved by counting all the problem tickets above the line and dividing by the total number of problem tickets.

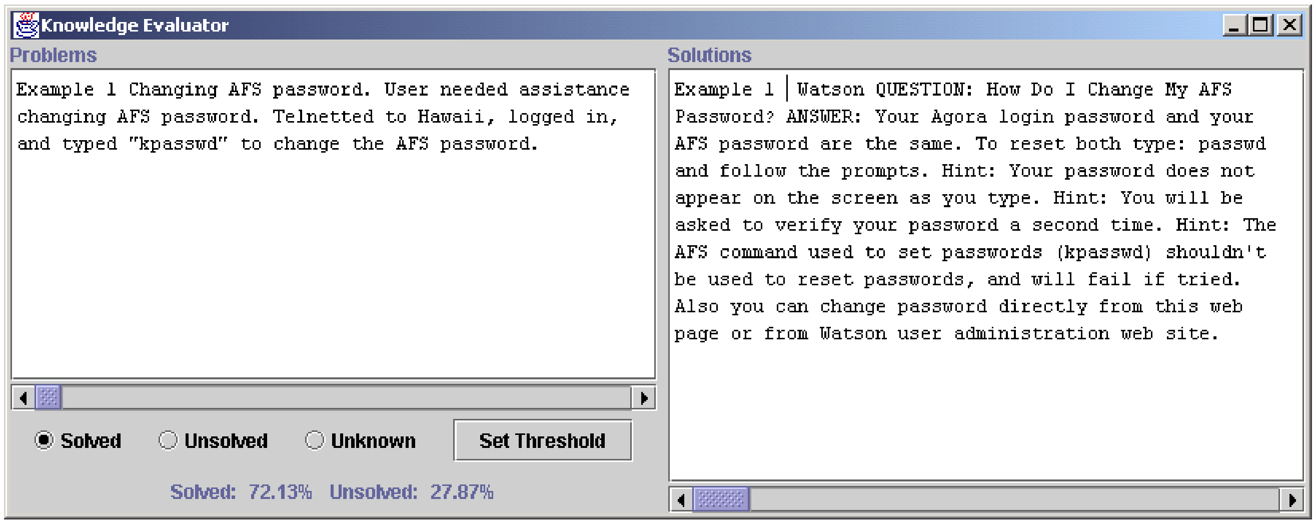
Figure 1: Knowledge Base Evaluation



Obviously, an expert's judgement is critical in drawing this line in the correct place. Only by actually reading and understanding the problem and solution pair can the expert determine if the knowledge base actually addresses the problem in question, which is something no computer program can do. What our system does is allow the expert to make an educated guess as to the percent solved, while putting forth the minimum effort required to make an informed choice. We claim that our approach leverages whatever time the expert can spend comparing problems and solutions to the utmost, by extrapolating information about the individual problems that are solved or not solved to evaluate the knowledge base as a whole.

Once the expert has drawn the line between solved and unsolved in the ordering of problem tickets, further refinement in the knowledge base evaluation may still be done if time permits. Individual problem tickets that occur above the line may be marked as unsolved by the expert and similarly individual problem tickets occurring below the line may be marked as solved. This is done in each case by compare the problem ticket in question to the corresponding most similar solutions in terms of g-score. The system will then adjust the percentage-solved score accordingly, based on this additional expert input. Thus the system allows the expert to utilize whatever time they are willing or able to spend in fine tuning their evaluation to improve its accuracy.

Figure 2: Knowledge Base Evaluator GUI



Example 1 Changing AFS password. User needed assistance changing AFS password. Telnetted to Hawaii, logged in, and typed "kpasswd" to change the AFS password.

Example 1 | Watson QUESTION: How Do I Change My AFS Password? ANSWER: Your Agora login password and your AFS password are the same. To reset both type: passwd and follow the prompts. Hint: Your password does not appear on the screen as you type. Hint: You will be asked to verify your password a second time. Hint: The AFS command used to set passwords (kpasswd) shouldn't be used to reset passwords, and will fail if tried. Also you can change password directly from this web page or from Watson user administration web site.

[✓ Solved] Unsolved Unknown

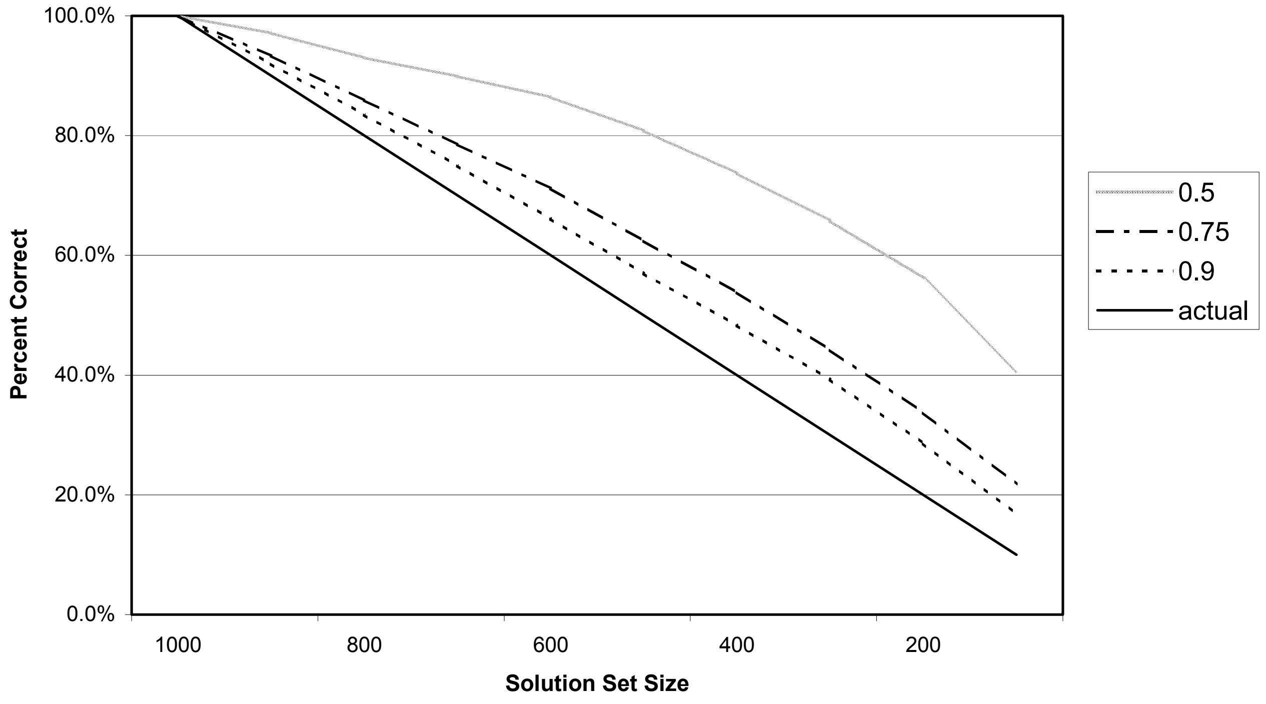
Solved: 72.13% Unsolved: 27.87%

In addition to marking problem tickets as "solved" or "unsolved" the user may choose to mark a ticket as "unknown" meaning neither or not sure. This allows us to put some "error bars" on our estimate of knowledge base coverage. Also note that each time the expert scrolls to a new problem, the GUI automatically updates the right hand side solution display window to include the 10 most applicable solutions from the knowledge base, sorted in order of decreasing g-score.

5. VALIDATION

To validate our approach we applied our knowledge base evaluator to an artificial situation where the problem space and solution space were identical. The text of the solution/problems were 1000 randomly selected solution documents from the Almaden helpdesk website. We then applied the automated method described in this paper to the identical problem and solution sets. Next we reduced the solution set size by systematically removing solutions to see the effect on solution set coverage. We tried this evaluation strategy with three different solved/unsolved g-score thresholds to see the effect (0.5, 0.75, and 0.9). The chart in figure 3 shows the results, along with the actual percentage solved by each solution set.

Figure 3: **Knowledge Relevancy Estimates of Artificial Solution Sets using Different Thresholds**



Clearly our algorithm is able to detect the missing solutions as they are removed from the solution set and reflect this in the percent correct score (i.e. percent solved) in every case. The fact that the algorithm overestimates the actual percentage correct for every threshold is not surprising, because some of the solution examples may use similar words and thus look "almost" the same to the system. The selection of the threshold at 0.9 produces the most accurate results for this artificial problem domain. Against a real problem ticket data set, this high a threshold might be impractical since problem tickets do not usually match so exactly with the corresponding solution text. In practice we have found that the 0.5 threshold works fairly well as a starting point, though clearly this might vary from domain to domain. Ultimately the expert must judge the best point at which to set this threshold in each situation.

6. APPLICATIONS AND CONCLUSIONS

We applied our knowledge base evaluator to two different knowledge solution sets to see which best addressed the solvable problems found at a typical IBM Managed Helpdesk. The purpose of the evaluation was to determine if the purchase of an off-the-shelf computer helpdesk knowledge base would be cost effective in terms of additional customer problems solved. Following the procedure described in the previous sections and using a g-score solved/unsolved threshold of 0.5, we arrived at the following percent-solved score for each KB.

Figure 4: Percentage Solved for Two Knowledge Base Solution Sets on IBM Managed Helpdesk data.

Knowledge Base	*Solution Set Size*	*Percent Solved*
IBM KB	5547	15.3%
Competitive KB	21728	33.4%

We found that the coverage of the competitive knowledge base was substantially better than the one we had developed in house. Based partly on this information it was decided that purchase of the competitive KB would be cost effective.

This initial application of our automated knowledge base evaluator has shown that it has the potential to greatly reduce the cost of evaluating knowledge bases, thus allowing for greater knowledge base reuse. Further study is needed to determine the overall accuracy of this approach when compared to direct expert evaluation of knowledge. One area of future work will focus on providing better "error bars" on the percentage-solved score. This might be accomplished by having an expert evaluate a sample of the "solved" problem tickets to determine how many were actually addressed by the solution set.

In conclusion, our approach shows significant promise as a method for automatically evaluating problem tickets against solution sets. Ultimately this method can be used as one ingredient in a determination of knowledge base relevancy. The ability to determine knowledge base relevancy in a cost effective manner should allow a greater degree of reuse of existing knowledge base resources within and across organizations.

ACKNOWLEDGEMENTS

The author would like to thank Mike Kreklow of IBM Global Services for first suggesting the problem of Knowledge Base Evaluation and for being the first user of the technology. His feedback was an invaluable ingredient in shaping this application.

7. REFERENCES

[1] Brachman, R. and Anand T. (1996). The Process of Knowledge Discovery in Databases. In Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., and Uthurusamy, R., editors, *Advances in Knowledge Discovery and Data Mining,* Chapter 2, pages 37-58. AAAI/MIT press.

[2] Duda, R. O. and Hart, P. E. (1973). *Pattern Classification and Scene Analysis.* Wiley.

[3] Frakes, W. (1992). Stemming algorithms. In Frakes, W. B. and Baeza-Yates, R., editors, *Information Retrieval: Data Structures and Algorithms,* pages 131-160. Prentice Hall, Englewood Cliffs, New Jersey.

[4] Hartigan, J. A. (1975) *Clustering Algorithms.* Wiley.

[5] Rasmussen, E. (1992). Clustering algorithms. In Frakes, W. B. and Baeza-Yates, R., editors, *Information Retrieval: Data Structures and Algorithms,* pages 419-442. Prentice Hall, Englewood Cliffs, New Jersey.

[6] Rich E., and Knight K. (1991). *Artificial Intelligence 2nd Edition*, McGraw-Hill Book Company, pp. 547-558.

[7] Salton, G. and McGill, M. J. (1983). *Introduction to Modern Retrieval.* McGraw-Hill Book Company.

[8] Salton, G. And Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management,* 4(5):512:523.

[9] Spangler, S. and Kreulen, J. (2001). Knowledge Base Maintenance Using Knowledge Gap Analysis. *Proceedings of the Seventh ACM SIGKDD-2001.* Pages 462-466. ACM, New York, NY, 2001.