Text Mining for Software Engineering: How Analyst Feedback Impacts Final Results

Jane Huffman Hayes and Alex Dekhtyar and Senthil Sundaram
Department of Computer Science
University of Kentucky
{hayes,dekhtyar}@cs.uky.edu, skart2@uky.edu

Abstract

The mining of textual artifacts is requisite for many important activities in software engineering: tracing of requirements; retrieval of components from a repository; location of manpage text for an area of question, etc. Many such activities leave the "final word" to the analyst – have the relevant items been retrieved? are there other items that should have been retrieved? When analysts become a part of the text mining process, their decisions on the relevance of retrieved elements impact the final outcome of the activity. In this paper, we undertook a pilot study to examine the impact of analyst decisions on the final outcome of a task.

1 Introduction

One of the distinguishing features of data mining versus, for example, similar database tasks, is the fact that knowledge acquired from mining need not be exact. In fact, it may, in part, be inaccurate. Methods for typical data mining tasks, such as classification, discovery of association rules, and retrieval of relevant information, do their best to produce the most accurate results. However, the accuracy is subject to the internal properties of the method, as well as the quality and complexity of the artifacts (data) under consideration.

In the field of Software Engineering, we can see two distinct and well-defined ways in which data mining, information retrieval, and machine learning methods are applied. The first direction is the exploratory study of existing artifacts of software development: document hierarchies, code repositories, bug report databases, etc., for the purpose of learning new, "interesting" information about the underlying patterns. Research of this sort is tolerant to the varying accuracy of data mining methods: while certain subtleties of some datasets might be missed, the most general patterns will, most likely, be discovered in analysis.

The second direction is the application of data mining[1] techniques to different processes in the software lifecycle with the purpose of automating and improving performance on the tasks involved. Potential benefits of such automation are significant. Data mining techniques are typically applicable to some of the most labor-intensive tasks, and are capable of speeding up the performance on them by orders of magnitude. At the same time, such applications of data mining methods **are not very error-tolerant**: undetected inaccuracies that creep into the results of data mining procedures may beget new inaccuracies in the later stages of development, thus producing a snowball effect.

To be able to obtain the benefits of applying data mining methods to specific tasks (good accuracy, *fast*), without the drawbacks (inaccuracies are very costly), a *human analyst must always assess and possibly correct the results of the automated methods*. The process of involving data mining methods in task execution during the software development lifecycle is described in Figure 1. A specific task is assigned to an analyst. The analyst has software to help execute the task. The analyst consults the software, obtains preliminary results, and provides the software with feedback. At some point, the analyst decides that the obtained answer is correct and outputs the final results of the task.

As the goal of introduction of data mining methods is improvement of the process, we are naturally concerned with the results produced by the automated tool. However, we observe that **the only result that is seen by others is generated by the analyst**! Therefore, we can only succeed if the final result, prepared by a human analyst, is good. In general, this is *not* equivalent to producing good results automatically.

We view this process from the point of view of the developers of the automated tool. Traditionally, the success of a data mining tool is measured by the accuracy of its re-

[1] Here and elsewhere in the paper we use the term "data mining" in its broadest sense, including certain related activities and methodologies from machine learning, natural language processing, and information retrieval in its scope.

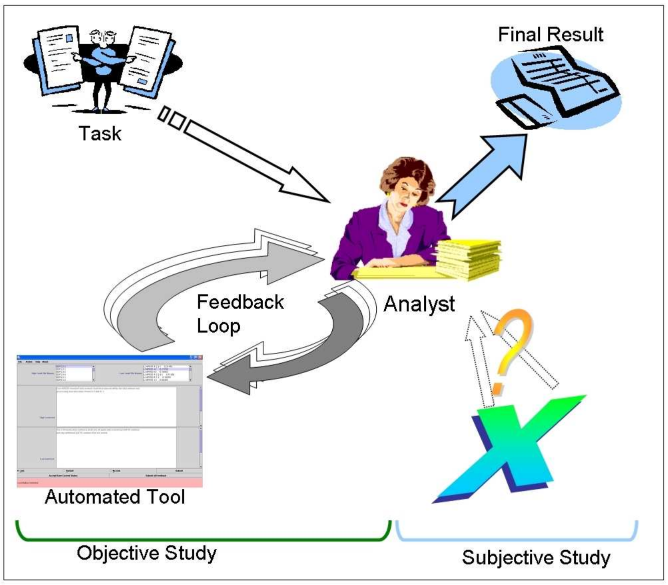

Figure 1. Human analyst will always stand between computer-generated results and the final result.

sults. However, in the process described in Figure 1, the ultimate concern lies with the accuracy of the final, analyst-generated output. This output is affected by a number of factors, including the accuracy of the automated tool. But is better accuracy of the tool equivalent to better accuracy of the analyst? And are there any other factors that play a role in analyst decision-making? Level of expertise? Trust of the software?

In order to claim success of the software, we must study not only the quality of the software output, but also *what the analysts do with it.* The ultimate success of the software is then determined by the quality of the final output.

What we have done. We have performed a pilot study on how human analysts work with machine-generated data. Our study was conducted using the task of IV&V requirements tracing [2, 3, 1] on a relatively small dataset. While the study was too small in size (only three analysts participated) to draw any far-reaching conclusions, its results (in all cases, the quality of the results decreased) suggest that we are looking at the right problem. The pilot study is discussed in Section 4 of the paper.

What we are planning to do. In Section 3 we outline the framework for a large scale experiment measuring the work of analysts with computer-generated data. Our goal is to determine the "regions" of precision-recall (see Section 2) space representing the quality computer-generated answer sets that allow human analysts to produce final results of high quality. We are also interested in studying what external factors affect analyst interaction with computer-generated artifacts.

	SuperTracePlus	Analyst
Correct links (total)	41	41
Correct links found	26	18
Total number of candidate links	67	39
Missed requirements	3	6
Recall	63.41%	43.9%
Precision	38.8%	46.15%

Table 1. SuperTracePlus and analyst performance on the MODIS dataset.

We begin by briefly outlining our research on the use of Information Retrieval (IR) methods for candidate link generation in requirements tracing tasks, and by describing how we came across the problem discussed in this paper.

2 Motivation: Humans Matter!

We first came across the issue of the quality of analyst evaluation of computer-generated results during our preliminary experiments with the application of information retrieval (IR) to requirements tracing [2]. At its core, requirements tracing boils down to comparing the content of pairs of high- and low-level requirements and determining whether they are similar/relevant to each other. We hypothesized that IR methods, that basically do the same thing, can be applied successfully to tracing.

We had implemented two IR methods and wanted to compare the results these methods produced with the results obtained by a senior analyst working with a familiar advanced requirements tracing tool (SuperTrace Plus). The complete results of that experiment can be found in [2]. The analyst received a tracing task (19 high-level and 50 low-level requirements, 41 true links from the MODIS [5],[4] documentation) and performed it in two steps. First, he used SuperTracePlus (STP) [6], a requirements tracing tool developed by Science Applications International Corporation (SAIC), to obtain a list of candidate links. The analyst then used the candidate trace generated by STP as a starting point and examined each link in detail. He removed from the trace links that he deemed unnecessary and also introduced some new links (wherever he felt that a link was missing). In Table 1, we have summarized this part of the experiment (all data comes from [2]).

As can be seen from the table, the analyst improved the precision of the final trace. However, the analyst significantly lowered the recall[2]. Using a recall-vs.-precision plot,

[2]Precision is defined as the number of correct answers returned divided

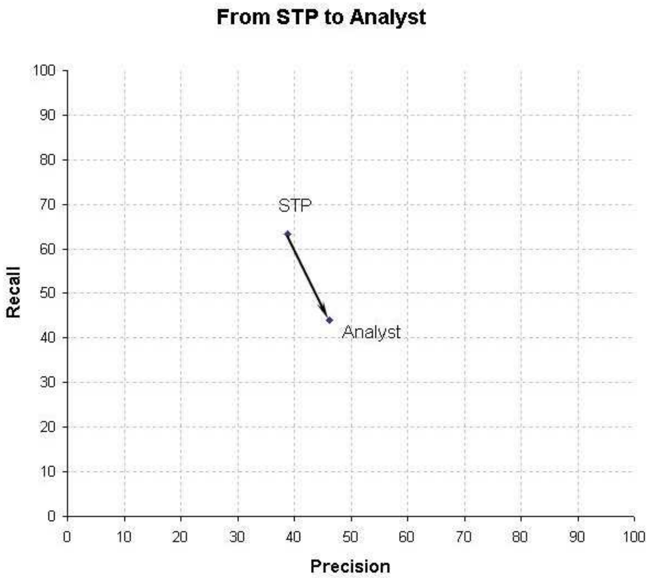

Figure 2. From SuperTracePlus trace to Analyst's trace.

we can illustrate the shift in these key measures of the quality of the trace from STP to the analyst (see Figure 2). In this experiment, the senior analyst had a high level of familiarity with SuperTracePlus, however, he was not very familiar with the MODIS project (beyond the dataset that we provided).

While a single point of anecdotal evidence is insufficient for any conclusions, it prompted us to consider the implications of the analyst's work with the software on the final results.

3 Large Scale Study

As mentioned in Section 1, when data mining tools are used directly in the software lifecycle process, rather than for after-the-fact analysis, high accuracy of the outcome must be ensured. Human analysts play the role of inspectors, examining the output of the tools and correcting it where necessary. The result of their work is passed along to the next tasks.

We ask ourselves a question: in the presence of mining tools, what factors affect the result produced by the analyst?

Right now, we only have a partial answer. Clearly, there are some overall factors that affect the quality of the analyst work, with or without software: analyst expertise with the task, level of domain expertise, and even such mundane and hard-to-control factors such as boredom with the task. However, in the presence of mining software designed to provide good approximations fast, there are other factors. The accuracy of the tool must play a role. Also, the degree of the analyst's familiarity with the tool and the degree of analyst trust in the results of the tool play a role.

However, simply stating that there is a dependence is not enough - as the exact character of such dependence is not obvious. For example, we would like to hypothesize that the quality of the tool results (measured in terms of precision and recall) affect the quality of analyst results in a monotonic way: better recall-precision of the tool yields better recall-precision of the final result. However, we note that if the precision and recall of the tool are very low (e.g., around 10% each), the analyst has "nowhere to go but up. " At the same time, when the precision and recall of the tool are very high (e.g., around 95%), the analyst has almost "nowhere to go but down. " Should we observe such results, how do we interpret them and what conclusions do we draw for the tool? Should we be "watering down" the results of an accurate tool, to ensure that an analyst will not decrease the precision and recall?

The goal of the large scale study we plan to undertake is to discover the patterns of analyst behavior when working with the results of data mining tools during the software lifecycle and to establish the factors that affect it and the nature of their effects.

The key factor we are looking at in the first stage is software accuracy, that we represent via the precision-recall pair of measures. The space of all possible accuracy results is thus a two-dimensional unit square as shown in Figure 3. For both precision and recall, we establish the regions where the values are *low*, *medium*, and *high*[3]. The nine regions are shown in Figure 3.

Our experiment consists of offering senior analysts, who have experience with requirements tracing, a computer-generated candidate trace with a preset accuracy from one of the nine regions. The analyst would then be asked to check the candidate trace and submit, in its place, the final trace. We will measure the accuracy of the final trace and note the shift from the original (such as in Figure 2).

Our goal is to establish under which levels/conditions of the software, analyst expertise, and analyst attitude towards software, the resulting output improves (significantly) upon the computer-generated candidate trace. Such discovery has two implications on the software and the process. We must ensure that the mining tool delivers results in the accuracy range that allows the analysts to improve it. We must also strive to create the right conditions for the analysts to work

by the total number of answers returned. Recall is defined as the number of correct answers returned divided by the total number of answers.

[3]There is a certain assymetry between recall and precision in this respect. We assume that precision is high if it is above 60%, and is low when it is under 33%. However, the recall is high when its value is above 70-75%, and is low when it is below 45%.

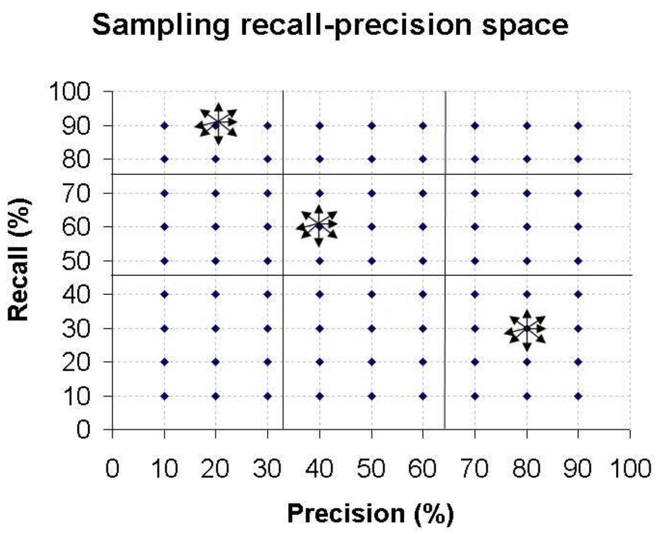

Figure 3. Sampling the space of possible outputs: what will the analysts do?

```
You may perform this task on your own schedule, at
your convenience, in the location that you prefer
(home, work, etc.).  The work need not be completed
all in one sitting.
We have provided you with the following items:
1 - A textual listing of the high level requirements;
2 - A textual listing of the low level requirements;
3 - A trace report indicating potential low-level
requirements for a given high-level requirement;
These items may be used in softcopy form (i.e., you
may feel free to use a word processing tool to perform
interactive searches, etc.)  or may be printed out and
used in hardcopy.  We will discuss each below.
The trace report contains the list of candidate links
for each high-level requirement.  The candidate links
for a single high-level requirement are shown in the
order of decreasing relevance.
Your task is to produce the final traceability report
from the given trace report and ensure that all high
level requirements have been traced correctly and that
any low level requirements for them have been identi-
fied.  The end product that you should provide to us
is:
- A marked up trace report (cross through low level
requirements that you do not agree with, write in low
level requirements that you feel are missing);
- A brief description of how you performed the task
(did you use Word to search through the softcopy?, did
you read everything from the hardcopy?);
- A log of how much time you spent on this task.
```

Figure 4. Instructions for the pilot study participants (abridged).

with the data.

4 First Steps

In our preliminary study, our goal is to investigate the feasibility of our hypothesis, that the accuracy of computer-generated candidate traces affects the accuracy of traces produced by the analysts. We also want to understand if a more detailed study is needed.

For the pilot study, we used the final version of the MODIS dataset described in [2, 3]. It contains 19 high-level requirements, 49 low-level requirements, and 41 true links between them. Using the output of one of our IR methods as the base, we generated candidate traces for a number of sampled points from the precision-recall space described in Section 3, including the three candidate traces (also shown in Figure 3) with the following parameters[4]:

T1: Precision=60%; Recall=40%;
T3: Precision=20%; Recall=90%;
T4: Precision=80%; Recall=30%[5].

The candidate traces were distributed to experienced analysts with tracing experience (manually or with a tool). The abridged version of the instructions provided to the analysts is shown in Figure 4.

Each analyst was provided with one of the trace sets described above. They were given a one-week period to perform the work, but were not given any time constraints (i.e., they could spend as many hours on the task as they desired). The analysts were asked to return their answer sets (all chose to return softcopies in various formats), a brief description of the process employed during the experiment (to determine conformance), and a brief log of activities. From each answer set we have collected the following information: **Or-Pr, Or-Rec**, original recall and precision; **Pr, Rec**, precision and recall achieved by the analyst; **Rmv, Rmv-Tr**, total number of links and number of true links removed and **Add, Add-Tr**, total number of links and number of true links added by the analyst; **Delta-Pr, Delta-Rec**, the change in precision and recall, and **Time**, the time spent on the task. Table 2 and Figure 5 summarize the results of the pilot study.

5 Conclusions and Future Work

As stated in Section 1, we are aware of some shortcomings of IR and text mining methods, such as that they admit inaccurate results. This is why, when used in tasks within the software lifecycle, an analyst needs to inspect computer-generated results to prevent the snowball effect.

[4]Altogether, we have generated six different candidate traces and distributed them to six analysts. However, only three analysts have completed their work at this time.

[5]Because we had 41 true links in the dataset, for some values of recall and precision, we had to take the nearest achievable point (e.g., 12 true links in the trace, equal to 29.2% recall, was used for the 30% recall point).

	T1	T3	T4
Or-Pr:	39.6%	20%	80%
Or-Rec:	60.9%	90.2%	29.2%
Rmv:	38	155	10
Rmv-Tr:	6	11	5
Add:	26	16	43
Add-Tr:	4	2	6
Pr:	45.1%	58.7%	22.9%
Rec:	56.1%	65.8%	26.8%
Delta-Pr:	+ 3.1%	+ 38.7%	- 57.1%
Delta-Rec:	- 5.8%	- 24.4%	- 2.4%
Time:	2.5 hrs	2 hrs	3 hrs

Table 2. Summarized results of the pilot study.

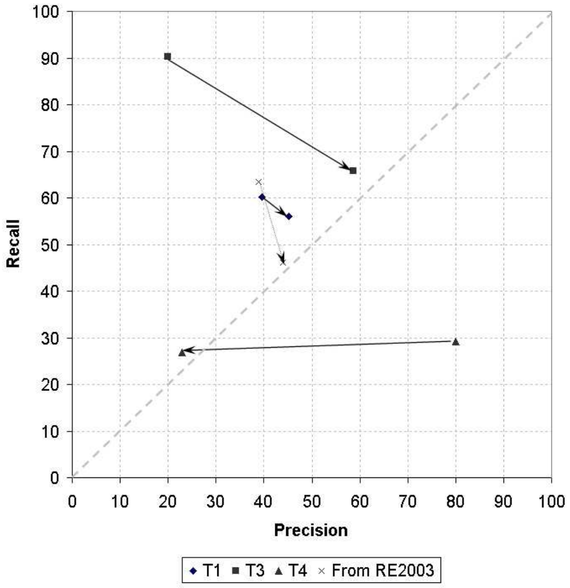

Figure 5. Pilot study: what the analysts did.

It is clear from our anecdotal study that there are factors at work influencing analyst decision making, and, hence, the final results. For example, examining Table 2, we can see that analysts who were given trace sets with low recall took longer to complete the task (25 - 50% longer). They did not necessarily produce any "worse" final results than the analyst with a high recall trace set (note that the analyst who had the high recall trace set ended with recall that was 24.4% lower). This observation is particularly interesting because the analyst with that trace set, T3 (recall of 90% and precision of 20%), had a large amount of false positives to go through. That means many more candidate links had to be examined. One would think that such an arduous process would result in worse results than an analyst who did not have to 'wade through' many false positives. But in this very small study, that was not the case.

In the pilot study, the analysts did not exhibit a pattern of improving the results of the candidate traces "no matter what." That would have rendered our questions moot. On the contrary, analyst behavior consistently shifted the answers towards the vicinity of the $precision = recall$ line (see Figure 5). This was evident if recall was higher than the $precision = recall$ line to begin with, or if it was lower than the $precision = recall$ line to begin with.

It is clear, then, that we must undertake a larger, controlled experiment, as described in Section 3. This must be done to ensure that we account for important factors that may influence analyst decisions, such as expertise, familiarity with/trust in the software, domain knowledge, etc... At the same time, we must factor out some extraneous variables, such as environmental issues (e.g., ambient noise), etc.

Acknowledgements Our work is funded by NASA under grant NAG5-11732. We thank Stephanie Ferguson, Ken McGill, and Tim Menzies. We thank the analysts who assisted with the pilot study.

References

[1] Dekhtyar, A., Hayes, J. Huffman, and Menzies, T., Text is Software Too, Proceedings of the International Workshop on Mining of Software Repositories (MSR) 2004, Edinborough, Scotland, May 2004, pp. 22 - 27.

[2] Hayes, J. Huffman, Dekhtyar, A., Osbourne, J. Improving Requirements Tracing via Information Retrieval, in *Proceedings of the International Conference on Requirements Engineering (RE)*, Monterey, California, September 2003, pp. 151 - 161.

[3] Hayes, J. Huffman, Dekhtyar, A., Sundaram K.S., Howard S., Helping Analysts Trace Requirements: An Objective Look, in *Proceedings of the International Conference on Requirements Engineering (RE)*, Kyoto, Japan, September 2004.

[4] Level 1A (L1A) and Geolocation Processing Software Requirements Specification, *SDST-059A, GSFC SBRS*, September 11, 1997.

[5] MODIS Science Data Processing Software Requirements Specification Version 2, *SDST-089, GSFC SBRS*, November 10, 1997.

[6] Mundie, T. and Hallsworth, F. Requirements analysis using SuperTrace PC. In *Proceedings of the American Society of Mechanical Engineers (ASME) for the Computers in Engineering Symposium at the Energy & Environmental Expo*, 1995, Houston, Texas.