Learning Effective Query Transformations for Enhanced Requirements Trace Retrieval

Timothy Dietrich, Jane Cleland-Huang, Yonghee Shin
Center of Excellence for Software Traceability (CoEST)
School of Computing, DePaul University, Chicago, IL, USA
tmdietrich@gmail.com, jhuang@cs.depaul.edu, syonghee@gmail.com

Abstract—In automated requirements traceability, significant improvements can be realized through incorporating user feedback into the trace retrieval process. However, existing feedback techniques are designed to improve results for individual queries. In this paper we present a novel technique designed to extend the benefits of user feedback across multiple trace queries. Our approach, named Trace Query Transformation (TQT), utilizes a novel form of Association Rule Mining to learn a set of query transformation rules which are used to improve the efficacy of future trace queries. We evaluate TQT using two different kinds of training sets. The first represents an initial set of queries directly modified by human analysts, while the second represents a set of queries generated by applying a query optimization process based on initial relevance feedback for trace links between a set of source and target documents. Both techniques are evaluated using requirements from the WorldVista Healthcare system, traced against certification requirements for the Commission for Healthcare Information Technology. Results show that the TQT technique returns significant improvements in the quality of generated trace links.

Index Terms—requirements traceability, query replacement, contractual requirements, text mining, machine learning, association rules

I. INTRODUCTION

Requirements traceability provides essential support for a wide variety of software engineering activities including impact analysis, compliance verification, and coverage analysis [11], [19]. Pre-specification trace links establish relationships between requirements and contribution structures such as rationales and regulatory codes, while post-specification links document how the requirement has been realized in the design, implemented in the code, and ultimately tested. Traceability is mandated by certification and approval bodies such as the USA Federal Aviation Authority and the Food and Drug Administration, and also recognized as a critical element of a sound engineering process [6]. In large and/or complex systems the number of required trace links can grow very large [18], and as a result, the manual effort required to create and maintain the needed links can be inhibitive.

Over the past decade, many research groups have proposed automated approaches based on information retrieval (IR) techniques. Commonly adopted approaches include the vector space model (VSM) [13], probabilistic approaches [7], or Latent Semantic Indexing (LSI) [4]. Results from a wide array of experiments, conducted across different domains and different types of artifacts, have demonstrated that although automated trace retrieval methods can significantly reduce the traceability effort by correctly rejecting approximately 90% of false links, they are imprecise, often exhibiting precision of only 20-50%. This means that an analyst must still spend significant time evaluating the results in order to find the correct set of links [9]. One promising solution involves the use of supervised learning techniques, in which user feedback is integrated into the traceability function either through use of machine learning [8], [10] or through adopting the well-known Rocchio approach which uses feedback to increase and/or decrease the weights of individual terms in a query [12], [15]. In both approaches, a modified query is created and used to regenerate a set of potentially improved trace links.

Unfortunately, in both cases, the user feedback is elicited for individual trace queries and its benefits are realized only for those same queries. This is particularly unfortunate in the traceability domain, where an analyst may need to create thousands or even hundreds of thousands of trace links. Leveraging the benefits of user feedback across queries would therefore be beneficial. In this paper we present a novel approach, which we refer to as Trace Query Transformation (TQT), designed to utilize feedback captured for an existing and validated set of trace queries and trace links in order to modify the remaining set of trace queries and improve the quality of trace links generated for the queries. TQT builds upon the concept of Direct Query Modification (DQM). In prior work [8], [10], [21] we introduced the DQM approach to allow a user to directly modify the trace query by filtering out unwanted terms and adding additional terms and synonyms. This manual manipulation of the query led to significant improvements in the quality of the generated trace links.

TQT takes a set of original queries and a corresponding set of modified queries as input, and uses them to learn transformation rules which can be applied to future queries. Our approach extends standard Association rule discovery techniques [3], which were initially developed in the E-Commerce domain to discover groups of products that buyers tend to purchase together. Association rule discovery algorithms analyze a set of transactions (i.e. items purchased at the same time), and identify recurring patterns such as the fact that people who purchase milk and diapers (nappies), are quite likely to also purchase beer. Trace query transformation differs from this type of “market-basket” analysis, which attempts to learn which items are typically found together in a single basket,
by learning the before-and-after associations. For example if terms authenticate and access occur together in the original query, then we frequently find the term role occurring in the correctly traced requirements, or a term such as patient that occurs frequently in the original queries, is found in so many target artifacts (both correct and incorrect ones), that more precise trace results can be achieved by removing it from the query.

We present two approaches for training TQT. The first uses a set of human modified queries from DQM as a training set for mining rules, while the second utilizes relevance feedback in which a human analyst has evaluated the correctness of an initial set of candidate links, and marked each link as correct (relevant) or not, creating a trace. These traces are used by our novel Automated Query Modification (AQM) to generate an optimized set of trace queries which then serve as the second training set for the TQT process.

In the remainder of the paper, Section II describes the the DQM versus the AQM approach for creating training sets of modified trace queries. Sections III and IV describe and evaluate our novel approach for learning query transformation rules from a training set of modified queries. Section ?? discusses the findings of our experiments. Finally, Section VI discusses threats to validity, Section VII discusses related work, and Section VIII summarizes our findings.

II. QUERY MODIFICATION

DQM allows the user to directly manipulate a trace query with the goal of improving the quality of the trace retrieval results. For example, given a trace query reading “The system shall support role based access for security controls” for which the automated stopword remover has eliminated terms such as the and system, the user may additionally manually remove terms such as support and controls, or add terms such as RBAC to produce a modified query. In practice we have observed that users intuitively modify queries in order to eliminate terms that reduce precision and also add terms that they deem to be useful in the retrieval process [21]. Support for DQM functionality is currently provided in trace retrieval tools such as Poirot [14].

A. Datasets and Their Preparation

In this paper we report the improvements achieved by using DQM and AQM in order to create a baseline for evaluating the TQT technique. All experiments are conducted using requirements for WorldVista, an electronic health care system developed by the USA Veterans Administration [2] traced against the Requirements for Ambulatory, Health Information Exchange, developed by the Certification Commission for Healthcare Information Technology (CCHIT) [1]. WorldVista contains 1064 requirements, of which 383 were traced to CCHIT requirements. The CCHIT EHR specification includes 454 requirements designed to evaluate health-care products for certification purposes. Trace links for this dataset were originally created by an undergraduate student who was hired in the summer of 2011 solely for this purpose. The created links were later evaluated and then confirmed and/or rejected by a member of our research team who had five years of experience working in the healthcare domain. These links were then used as a reference set for our study. We discuss threats to validity related to this dataset in Section VI.

![MAP for each Dataset](image)

**FIG. 1:** A comparison of MAP Scores achieved using the Baseline queries vs. DQM and AQM modified queries.

B. Evaluation Metrics

The metrics of Average Precision (AP) and Mean Average Precision (MAP) are used for evaluation purposes. MAP computes the extent to which correct links are placed at the top of the ranked list of generated trace links across an entire collection of documents. Because our implementation of MAP examines all correct links it also assumes recall (i.e. the ability to retrieve correct links) of 100%. The use of MAP as a traceability measure has been advocated in numerous papers [22], [23]. First the AP of each query is computed:

$$AP = \sum_{r=1}^{N}(\text{Precision}(r) \times \text{isRelevant}(r)) / \text{RelevantDocuments}$$  

where \( r \) is the rank of the target artifact in an ordered list of links, \( \text{isRelevant}(\cdot) \) is a binary function assigned 1 if the link is relevant and 0 otherwise, \( P(r) \) is the precision computed after truncating the list immediately below that ranked position, and \( N \) is the total number of documents. When multiple links are listed for a single similarity score (e.g. at similarity of zero) the links are evenly distributed across the space of that score, simulating their random distribution. MAP is then computed across all queries as follows:

$$\text{MAP} = \frac{\sum_{q=1}^{Q}AP}{Q}$$

where, \( q \) is a single query and \( Q \) is the total number of queries.

C. Creating a Baseline

To compute Baseline results, we generated trace links using the standard Vector Space Model as described in our prior work [21]. Baseline results therefore include no user feedback.
This detailed example shows how transformation rules are created using both AQM and DQM training sets.

**D. Modified Queries & Direct Query Modification (DQM)**

While the benefits of using DQM have been demonstrated in our previous work [21], we replicated this experiment against our new dataset in order to create a new DQM baseline for the novel work described in this paper. To accomplish this, one member of our team used the DQM feature in our tracing tool, Poirot, to improve trace results by manually modifying each trace query. The goal was to improve each individual query so that it successfully pulled correct links to the top of the ordered set of candidate links.

**E. Automated Query Modification (AQM)**

The modified queries serve as the input to the TQT approach described in the second half of this paper. However, because human modified queries are not always available, and not necessarily optimal, we developed a technique for generating “ideal” queries from an initial set of verified trace links.

AQM first eliminates terms in the query which are found in none of the relevant target documents. For example, if the term *subscribe* is found in the original query, but is not found in any of the traced target documents, then it is removed from the query.

In the second phase of the algorithm, all terms found in the combined set of relevant target documents are ranked according to IDF (inverse document frequency) values following standard formulas for computing IDF [21]. Each term in turn is then tentatively added to the candidate query, traces are generated for the new query against all documents using VSM, and the subsequent MAP score is computed. If the inclusion of the term improves the MAP score, then it is retained in the query, otherwise it is rejected. The result is a modified query that either improves its own MAP score or leaves it unchanged. To evaluate AQM, the set of AQM queries were used to generate trace links from WorldVista to CCHIT requirements using VSM.

**F. Results**

Figure 1 reports MAP scores achieved unmodified queries (baseline), human modified queries (DQM), and queries learned from the training set of verified links (AQM). The baseline returned MAP scores of 0.257 using the original unmodified trace queries, while the DQM queries returned MAP scores of 0.453. Interestingly, the AQM queries returned a MAP score of 0.512 and therefore outperformed DQM. However, this is not particularly surprising; and it is important to remember that DQM queries are created manually by users before the correct links are known, while AQM queries are created after the fact. AQM can therefore not be used in the same way as DQM i.e. to aid in creating the initial traces it requires to learn. Instead, given an initial set of verified trace links, we can use AQM to create a set of modified queries; thereby creating a training set for the TQT approach described in the remainder of this paper.

### III. Trace Query Transformations (TQT)

In the remainder of the paper, we assume an initial training set of original and modified pairs of trace queries. This training set is used to learn a set of transformation rules which can then be applied to future queries. Before presenting the modifications we made to the FP-Growth algorithm in order to generate such rules, we first introduce the RACE model.
to support query transformations, we first present the basic algorithm. Given a set of transactions $T$ and a set of items $I = \{I_1, I_2, \ldots, I_k\}$, and an item set $is \subseteq I$, let $T_{is} \subseteq T$ be the set of transactions that have all the items in $is$. The support of the item set $is$ is defined as $\sigma(is) = |T_{is}| / |T|$. Item sets that satisfy a predefined support threshold are referred to as frequent item sets. An association rule $r$ is expressed in the form $X \Rightarrow Y(\sigma_r, \alpha_r)$, where $X \subseteq T$ and $Y \subseteq T$ are item sets, $\sigma_r$ is the support of the item set $X \cup Y$, and $\alpha_r$ is the confidence for the rule $r$ given by $\sigma(X \cup Y)/\sigma(X)$.

The discovery of association rules from a transaction database involves two main parts: the discovery of frequent item sets (i.e. those item sets which satisfy a minimum support threshold) and the discovery of association rules from these frequent item sets which satisfy a minimum confidence threshold. Note that association rules are only generated from frequent item sets, so items that do not appear in at least one frequent item set are filtered out and will not appear in any rules. The strength of a discovered rule is measured according to the support for the underlying item set as well as the confidence of the rule.

A. Applying FP-Growth to Trace Query Transformation

In this paper we modified the FP-Growth algorithm to learn a set of query transformation rules. The query transformation problem differs in several primary respects from typical Association Rule Mining. First, whereas Association Rule Mining is concerned with discovering patterns of co-occurrence, the query transformation problem is concerned with discovering patterns of transformation. For example, instead of discovering that terms $a$ and $b$ co-occur in a single transaction, we are interested in discovering that when terms $a$ and $b$ co-occur in an original query, $c$ is found in the user-modified query (from now on referred to as the modified query). Similarly, when $d$ is found in an original query it rarely, if ever, is retained in the modified query. This means that while association rules can be modeled as a single state, the transformation problem must be modeled as two states representing the original query and the modified query. Furthermore, whereas the Association Rule Mining problem is interested in “constructive” rules such as $a, b \Rightarrow c$, the transformation problem is interested with both “constructive” and “destructive” rules, such that a “destructive” rule explicitly specifies items that should be removed from the original query, as seen in 2

B. Implementation

We start with an initial training set containing pairs of original and modified queries, and then datamine a set of transformation rules that determine how best to transform future trace queries. To facilitate the identification of transformation rules, the frequent item sets are stored in a directed acyclic graph, called a Frequent Itemset Graph (FIG)[16], [20]. A standard FIG is organized into levels from 0 to $k$, where $k$ is the maximum size among all discovered frequent item sets. Each node at depth $d$ in the graph corresponds to an item set $I$ of size $d$ and is linked to item sets of size $d + 1$ that contain $I$ at the next level. The root node at level 0 corresponds to the empty item set. Each node also stores the support value of the corresponding frequent item set. The FIG represents the aggregate model learned in the first phase of association rule discovery performed offline. The transformation rules are, however, generated at query time by performing depth-first searches of the FIG. For the query transformation process, the FIG is modified so that only non-annotated terms (i.e. terms from the original query) can appear in intermediate nodes, while leaf nodes can include both non-annotated terms and annotated terms. In this way, the rules that are generated by depth-first-search of the FIG are guaranteed to have valid forms.

Given a new trace query comprised of a term set $t$, the algorithm performs a depth-first search of the graph up to level $|t|$ to find each subset $t'$ of the term set $t$. Among the nodes that include a subset $t'$, only the nodes with annotated terms (leaf nodes) are used for transformation recommendation. Each transformation recommendation $r$ is a set of annotated terms with a frequent term set $t' \cup \{r\}$ at level $|t'| + 1$. For each such child node of $t'$, the annotated term set $r$ is added to the recommendation set if the support ratio $\sigma(t' \cup \{r\})/\sigma(t')$, which is the confidence of the association rule $t' \Rightarrow \{r\}$, is greater than or equal to a pre-specified minimum confidence threshold. Then the generated rules are applied to the trace query sequentially to transform it into a new query.

Because we apply TQT to two different training sets, we adopt the notation of referring to the queries transformed by TQT, using DQM as a training set, as DQM+ queries, and those obtained using the AQM training set as AQM+ queries.

An analysis of DQM, and subsequently DQM+ behavior, versus AQM and AQM+ behavior, shows that both approaches result in the removal of terms from documents, but that AQM adds a substantial number of additional terms. We explain this phenomenon by the fact that in DQM, human users are required to think creatively about which words to add. This takes more effort and often does not occur in practice [21].

IV. Evaluation of TQT

To evaluate TQT with training sets of DQM and AQM queries, we conducted a standard leave-one-out cross validation experiment. We selected this approach because of the relatively small size of the dataset, which had 383 queries with validated traces, and 681 queries with no traces. Each of the 383 traced queries was systematically set aside for testing, while the remaining 382 queries were used to construct a FIG as described in Section III, and generate transformation rules which were applied to the test query. As a result, each of the 383 queries in turn, was transformed into a new query.

The experiment was repeated at support levels of 1, 5, 10, 15, 20, and 25, and for each support level at confidence scores from 10% to 100% at intervals of 10%. For each experiment, the transformed WorldVista queries were traced against the CCHIT requirements, and the resulting MAP score was computed. In Fig 3, results show that both DQM+ and
AQM+ generated substantial improvements over the Baseline results.

V. RESULTS

As depicted in the topological graphs reported in Figure 4, both DQM+ and AQM+ showed increases in MAP as Support and Confidence increase. Given that MAP scores decrease at low confidence and support levels, it is important to apply TQT in a conservative manner. DQM+ performed best at confidence levels of 1.0, as seen in the convergence of support counts 5, 10, 15, 20, and 25. While this is a truly worthwhile improvement of 24.4% over the baseline, it falls short of the manually modified DQM level by -29.6%. AQM+, by contrast, showed continual MAP improvement as confidence and support levels increased, and returned a MAP score of 0.454 representing an improvement of 76.9% over the baseline, and failing short of the original AQM MAP score by only 11.2%.

It is interesting to note that Figure 4 shows that neither DQM+ nor AQM+ were consistently successful in improving a query’s AP and in some cases damaged it. In the lowest Confidence and Support count bands, DQM+ and AQM+ facilitated upwards of 17,000 and 84,000 applicable rules, respectively. This invites the application of spurious rules, none of which will add value in improving the queries. This is a noted issue within Association Rule Mining [3] and these false positives cause more damage than benefit. Within several segments of Support count and Confidence in both DQM+ and AQM+ generated trace links.

Fig. 4: Mean Avg Precision across various Support count and Confidence levels in DQM+ and AQM+

AQM+, full queries had their AP scores decimated to 0.00. The simplest approach is to balance Support count levels with Confidence, as implemented here. It is possible, however, that more information could be mined from lower Support and lower Confidence levels with a different algorithm to handle these “false positive” rules. As it stands, the Support count and Confidence did a satisfactory job of managing this risk.

Nevertheless, in our reported experiments, the majority of modifications were beneficial. DQM+ improved 45% of it’s queries, while failing only 16%. While AQM+ improved a modest, but critical, 8% of it’s queries and only failed 0.5%. These results are of particular interest, because they suggest that AQM+ could be applied across the entire set of queries, where it has the potential to very significantly improve AP of approximately 10% of the queries while having almost no ill effect on other queries. At least in this experiment, the increase in overall MAP score from a Baseline of 0.257 to 0.454 represents a non-trivial improvement in the quality of generated trace links.

VI. THREATS TO VALIDITY

There are several threats to validity for our work. First, the experiments were conducted against only one medium sized dataset, and we therefore cannot claim that the approach can be generalized to all traceability problems. In fact our results show that it worked more effectively on some trace queries than others, and therefore future work is needed to replicate the experiment with other datasets in order to better understand the constraints on the approach. Secondly, a threat to internal validity could emerge if bias were introduced during the creation
of the reference set used to evaluate MAP scores. We partially mitigated this threat through hiring an independent 3rd party to create the initial trace links. We also plan to release our dataset into the public domain via CoEST.org where it will be open to further scrutiny and possible revisions. The work described in this paper represents an initial study in trace query modification. With respect to internal and external validity of the study, numerous observations were made which cannot be fully substantiated or generalized without conducting a broader study that includes larger and more varied datasets. Nevertheless, we have been careful throughout the paper not to over claim results based on our observations.

VII. RELATED WORK

Several researchers have investigated the use of relevance feedback in the Traceability domain. In particular, Antoniol et al. [5] and Penta et al. [17] provided a subset of correct links as a training set and incorporate knowledge of these correct links into the trace process. Cleland-Huang et al. [8], [10], also incorporated the use of a training set, but used it to create a trace classifier that was trained to identify specific types of non-functional requirements or regulatory codes. Huffman Hayes et al. [12] investigated the use of Rocchio relevance feedback and allowed the user to iteratively classify candidate links either as true or false positives. This information was then used to modify the term weights in the query using the Rocchio algorithm. De Lucia et al. [15] proposed a technique for presenting trace links in an incremental Rocchio fashion, in which Rocchio iterations continued ad infinitum until the analyst tired of the process and determined no additional links were to be found. They showed that relevance feedback generally improves trace results when applied to either the Vector Space Model or to Latent Semantic Indexing (LSI).

VIII. CONCLUSION

This paper has presented a novel approach for learning a set of Trace Query Transformation rules. Current, state-of-the-art techniques that utilize user feedback as part of the tracing process are constrained to realizing their benefits within individual trace queries. In this paper we have presented techniques that are designed to extend the benefits of both capturing trace user’s relevance feedback and their subsequent modifications to a query onto a subset of queries not previously reviewed. While the approach was initially designed to mimic the way human analysts modified queries using DQM, we also presented AQM which enabled us to generate a training set of modified queries using only relevance feedback. While neither DQM+ nor AQM+ improved every query, both techniques produced substantial overall improvements in MAP scores, and both approaches resulted in significant improvements in some subset of queries.

ACKNOWLEDGMENT

The work described in this paper was partially funded by grant III:0916852 from the National Science Foundation.

REFERENCES

[23] E. Weyuker, R. Bell, and T.J.Ostrand. We’re finding most of the bugs, but what are we missing? In 3rd International Workshop on Testing, Verification and Validation, 2010.