Relevance Feedback for Association Rules using Fuzzy Score Aggregation

Georg Ruß
Institute for Knowledge and Language Engineering
University of Magdeburg, Germany
Email: russ@iws.cs.uni-magdeburg.de

Mirko Böttcher
Intelligent Systems Research Centre
BT Group plc, Ipswich, UK
Email: mirko.boettcher@bt.com

Rudolf Kruse
Institute for Knowledge and Language Engineering
University of Magdeburg, Germany
Email: kruse@iws.cs.uni-magdeburg.de

***Abstract*—We propose a novel and more flexible relevance feedback for association rules which is based on a fuzzy notion of relevance. Our approach transforms association rules into a vector-based representation using some inspiration from document vectors in information retrieval. These vectors are used as the basis for a relevance feedback approach which builds a knowledge base of rules previously rated as (un)interesting by a user. Given an association rule the vector representation is used to obtain a fuzzy score of how much this rule contradicts a rule in the knowledge base. This yields a set of relevance scores for each assessed rule which still need to be aggregated. Rather than relying on a certain aggregation measure we utilize OWA operators for score aggregation to gain a high degree of flexibility and understandability.**

I. Introduction

Association rule mining [1], [2] originally has been developed for market basket data analysis, where each basket, also referred to as a transaction, consists of a set of purchased items. The goal of association rule mining is to detect all those items which frequently occur together and to form rules which predict the co-occurrence of items. However, association rule mining is not just bound to this specific purpose. It can be applied, for example, to every relational database.

Nowadays, the discovery of association rules is a relatively mature and well-researched topic. Many algorithms have been proposed to ever faster discover and maintain association rules. However, one of the biggest problems of association rules still remains unresolved. Usually, the number of discovered associations will be immense, easily in the thousands or even tens of thousands. Clearly, the large numbers make rules difficult to examine by a human user. Moreover, many of the discovered rules will be obvious, already known, or not relevant to a user. For this reason several methods have been proposed to assist a user in detecting the most interesting or relevant ones. The vast majority of these approaches either calculate a relevance score or determine rules that contradict a user's prior knowledge based on Boolean logic.

In this paper we argue that such approaches only insufficiently reflect the way a user searches for relevant rules because a user's perception of relevance is not a static but rather a dynamic process due to several reasons: firstly, when a user starts to explore a set of discovered association rules he only has a very vague notion about which rules might be relevant to him. Secondly, while seeing more rules his knowledge about the domain of interest changes, some aspects might gain while others might lose importance. His notion of relevance depends on these changes and thus changes too, almost always becoming clearer.

The importance of user dynamics in assessing the relevance of data mining results only recently gained attention in the research community [19]. However, it is a rather well-researched topic in the field of information retrieval. In fact, the way a user builds up his internal notion of relevancy described above is very similar to the models of user behaviour used in information retrieval (cp. [3]). Based on these similarities we present a new approach to the problem of finding the most relevant rules out of a large set of association rules which is inspired by ideas from information retrieval. Our approach uses relevance feedback to acquire users' preferences and to build a knowledge base of what he considers to be relevant and non-relevant, respectively. By calculating the (dis-)similarity of each unexamined rule with the rules in the knowledge base and aggregating the scores we obtain a relevance score which—with each feedback provided—better reflects the user's notion of relevance.

The remainder of this paper is organised as follows: Section II gives the background on association rules, Section III shows the related work that is most relevant to our topic. Section IV will further elaborate the link between information retrieval and interestingness assessment of association rules. Section V introduces a novel notion of association rules based on features vectors which are inspired by document vectors from information retrieval. This representation is closely related to our notion of rule similarity explained in Section VI and Section VII. The relevance scoring metric will be derived in Section VIII before Section IX concludes the paper.

II. Association Rules

Formally, association rule mining is applied to a set $\mathcal{D}$ of *transactions* $\mathcal{T} \in \mathcal{D}$. Every transaction $\mathcal{T}$ is a subset of a set of items $\mathcal{L}$. A subset $\mathcal{X} \subseteq \mathcal{L}$ is called *itemset*. It is said that a transaction $\mathcal{T}$ *supports* an itemset $\mathcal{X}$ if $\mathcal{X} \subseteq \mathcal{T}$.

An association rule r is an expression $\mathcal{X} \rightarrow \mathcal{Y}$ where $\mathcal{X}$ and $\mathcal{Y}$ are itemsets, $|\mathcal{Y}| > 0$ and $X \cap Y = \varnothing$. Its meaning is quite intuitive: Given a database $\mathcal{D}$ of transactions the rule above expresses that whenever $\mathcal{X} \subseteq \mathcal{T}$ holds, $\mathcal{Y} \subseteq \mathcal{T}$ is likely to hold too. If for two rules $r : \mathcal{X} \rightarrow \mathcal{Y}$ and $r' : \mathcal{X}' \rightarrow \mathcal{Y}$, $\mathcal{X} \subset \mathcal{X}'$ holds, then it is said that r is a *generalization* of r'. This is denoted by $r' \prec r$.

As usual, the reliability of a rule $r : \mathcal{X} \rightarrow \mathcal{Y}$ is measured by its *confidence* $\mathrm{conf}(r)$, which estimates $P(\mathcal{Y} \subseteq \mathcal{T} \mid \mathcal{X} \subset \mathcal{T})$, or short $P(\mathcal{Y} \mid \mathcal{X})$. The statistical significance of r is measured by its *support* $\mathrm{supp}(r)$ which estimates $P(\mathcal{X} \cup \mathcal{Y} \subseteq \mathcal{T})$, or short $P(\mathcal{X}\mathcal{Y})$. We also use the support of an itemset $\mathcal{X}$ denoted by $\mathrm{supp}(\mathcal{X})$.

III. Related Work

The strength of an association rule learner to discover all patterns is likewise its weakness. Usually the number of discovered associations can be immense, easily in the thousands or even tens of thousands. Clearly, the large numbers make rules difficult to examine by a human user. Therefore significant research has been conducted into methods which assess the relevance, or interestingness, of a rule. Studies concerning interestingness assessment can roughly be divided into two classes. The first class are objective measures. These are usually derived from statistics, information theory or machine learning and assess numerical or structural properties of a rule and the data to produce a ranking [18]. Objective measures do not take any background information into account and are therefore suitable if an unbiased ranking is required, e.g. in off-the-shelf data mining tools. The second class are subjective measures which incorporate a user's background knowledge. In this class a rule is considered interesting if it is either *actionable* or *unexpected*.

Actionability of a rule means that the user "can act upon it to his advantage" [16]. Their focal point is on rules that are advantageous for the user's goals. The actionability approach needs detailed knowledge about the current goals and also about the cost and risks of possible actions. Systems that utilise it are hence very domain specific, like the *KEFIR* system described in [13].

A rule is unexpected if it contradicts the user's knowledge about the domain. Systems that build upon this approach require the user to express his domain knowledge – a sometimes difficult, long and tedious task. The methods are usually based on pairwise comparison of a discovered rule with rules representing the user knowledge. This comparison can be logic-based [10], [11], [12] or syntax-based [8]. In logic-based systems a contradiction is determined by means of a logical calculus, whereas in syntax-based systems a rule contradicts if it has a similar antecedent but a dissimilar consequence.

In [10], [11], [12] the authors connect belief models with association rules. In particular, they assume that a belief system has been provided by the user whereby beliefs are defined as association rules. Based on this definition they provide a set of conditions to verify whether a rule $\mathcal{X} \rightarrow y$ is *unexpected* with respect to the belief $\mathcal{X} \rightarrow z$ on the rule database D. They propose an algorithm *ZoomUR* which discovers the set of unexpected rules regarding a specified set of beliefs. The algorithm itself consists of two different discovery strategies: *ZoominUR* discovers all unexpected rules that are refinements (or specialisations). On the other hand, *ZoomoutUR* discovers all unexpected rules that are more general.

In [8] the authors address the insufficiency of objective interestingness measures by focusing on the unexpectedness of generalised association rules. They assume that taxonomies exist among association rules' attributes. In subsequent work [9], human knowledge is recognised to have different degrees of certainty or preciseness. Their system allows for three degrees, notably *general impressions*, *reasonably precise concepts* and *precise knowledge*. The interestingness Analysis System (IAS) they propose accounts for these degrees and uses the gathered knowledge to find rules which are unexpected in regard to the expressed knowledge. IAS works iteratively: first, the user specifies his knowledge or modifies previously specified knowledge, supported by the specification language; second, the system analyses the association rules according to conformity and unexpectedness; and third, the user inspects the analysis results (aided by visualisation), saves interesting rules and discards uninteresting rules. In the first step, based on the three preciseness categories, a user can express his knowledge, with constraints for each category's contents for syntax as well as confidence and support values. In the second step, the IAS uses this information by performing a syntax-based analysis to find unexpected rules, i.e., those which do not conform to the knowledge. Since each rule consists of an antecedent and a consequent with boolean conformity (*matches*, *does not match* the specified knowledge), the four resulting possibilities are exploited to determine unexpected rules by calculating a degree of match. Using those degrees of match, a re-ranking of the rules is calculated. Finally, a visualisation is used to present the results to the user.

IV. Using Concepts from Information Retrieval

Existing approaches to assess the relevance of association rules strongly require a user to explicitly specify his existing knowledge in advance. This leads to two major drawbacks. In the first place, when specifying their existing knowledge, domain experts often forget certain key aspects or may not remember others which come into play under rarer circumstances. This problem can be termed 'expert dilemma' and has already been observed by designers of expert systems in the 1980s [6]. Secondly, at the beginning of an analysis session a user can only very vaguely specify what he considers to be relevant. His notion of relevance only becomes clearer the more rules he examines. This problem, that a user is incapable of specifying his information need from scratch, is very well-known in the field of information retrieval [3] where it lead to the development of relevance feedback methods.

Relevance feedback is an intuitive technique that has been introduced to information retrieval in the mid-1960s [14]. In information retrieval it is a controlled, semi-automatic, iterative process for query reformulation, that can greatly improve the usability of an information retrieval system [7]. Relevance feedback allows a user to express what he considers to be relevant by marking rules as relevant and non-relevant, respectively. Whenever a rule has been marked as relevant, it is added to the set of relevant rules $R_{\mathtt{r}}$. Whenever a rule is marked as non-relevant, it is added to the set of non-relevant rules $R_{\mathtt{n}}$. For simplicity, we will assume that in each feedback cycle exactly one rule is marked.

After each feedback cycle the remaining rules are compared with the set of annotated rules and a new relevance score is calculated. The set of annotated rules, in turn, can be seen as a representation of the user's notion of relevance. Hence it also provides a solution to the first of the above-mentioned drawbacks by supporting an iterative, easy way for a user to specify his knowledge about a domain. For example, he may annotate rules that are already known as non-relevant and some novel rules as relevant.

In order to develop a feedback system for association rules the following questions need to be answered:

- How do we represent association rules for the purpose of relevance feedback?
- How do we score the likely relevance of a rule in relation to a rule already marked as (non-)relevant?
- How do we aggregate those scores to an overall relevance score?

We will provide answers to these questions in the subsequent sections. In particular we are aiming at adapting established methods from information retrieval.

V. Rule Representation

To be the core building block of a relevance feedback approach it is necessary to transform the rules into an equivalent representation. In particular, such a representation should have a couple of properties. Firstly, rather than relying on generalisation and specialisation relationships among rules as a key to rule similarity it should support a less crisp and thus more flexible definition. For example, rules that have the same consequent and share items in their antecedent should be regarded as similar to a certain degree. Secondly, items have a different importance to a user. For example, an item that is contained in almost every rule does not contribute much towards a user's understanding of the domain, whereas an item that is only contained in a few rules can contribute considerably. This importance should be reflected in the rule representation. Thirdly, it should be easy to extend the rule representation by further numeric properties of a rule. For example, recently there has been an increasing interest into the change of a rule's support and confidence values (e.g. [4]). In this scenario the rule representation should incorporate the timeseries of support or confidence in order to enable similarity calculations based on rule change. To illustrate the usage of further information about rules for relevance feedback we will use the example of rule change throughout this paper.

A representation that fulfills all of the above requirements is a *feature vector* $\vec{r}$ *of an association rule* r whose elements are numerical values and which consists of three components: a representation of the rule's body, a representation of the rule's head and a rule's time series. The latter component can easily be replaced by other numeric features of a rule or completely omitted. Formally, a feature vector thus is defined as

$$\vec{r} = (\underbrace{\overbrace{r_1, \ldots, r_b}^{\mathtt{body}}, \overbrace{r_{b+1}, \ldots, r_{b+h}}^{\mathtt{head}}}_{\mathtt{symbolic}}, \underbrace{r_{b+h+1}, \ldots, r_{b+h+t}}_{\mathtt{timeseries}}) \quad (1)$$

The different components can be seen as a projection of $\vec{r}$ and will be referred to as follows:

$$\vec{r}_{\mathtt{body}} = (r_1, \ldots, r_b) \quad (2)$$
$$\vec{r}_{\mathtt{head}} = (r_{b+1}, \ldots, r_{b+h}) \quad (3)$$
$$\vec{r}_{\mathtt{sym}} = (r_1, \ldots, r_{b+h}) \quad (4)$$
$$\vec{r}_{\mathtt{time}} = (r_{b+h+1}, \ldots, r_{b+h+t}) \quad (5)$$

To calculate the *item weights* r_i we adapted the well-known TF-IDF approach [15] from information retrieval. The TF-IDF approach weights terms according to their appearance in a document and in the overall document collection. A high term weight, which is correlated with a high importance of that particular term, is achieved if the term appears frequently in the document (term frequency, TF) but much less frequently in the document collection (inverse document frequency, IDF). This approach filters out commonly used terms and tries to capture the perceived relevance of certain terms.

This method, carried over to association rules, means that items that appear in the vast majority of rules will get a very low weight whereas items that are rather infrequent will get a rather high weight. Since item appearance in rules is linked to item appearance in a data set this also means that infrequent attribute values in the data set will receive a high weight.

The term frequency tf of an item x in an association rule r is calculated as follows:

$$tf(x, r) = \begin{cases} 1 & \text{if } x \in r, \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

The inverse document frequency idf of an item x in an association rule r and in regard to a rule set R is calculated as follows:

$$idf(x, R) = 1 - \frac{\ln |r : r \in R \wedge x \in r|}{\ln |R|} \quad (7)$$

To generate feature vectors, a series of steps has to be performed. For body and head separately, a set of items is generated: $I_{body} = \{x_1, \ldots, x_b\}$ and $I_{head} = \{x_1, \ldots, x_h\}$ where the x_i are the items that occur in body or head of the association rules in R, respectively. Each item of these sets is assigned exactly one vector dimension in $\vec{r}_{\mathtt{body}}$ or $\vec{r}_{\mathtt{head}}$. Hence, the values for b and h in (1) are the cardinalities of the respective itemsets: $b = |I_{body}|$ and $h = |I_{head}|$

The symbolic part of the feature vector of an association rule r will contain TF-IDF values. Let x_i the i-th item of the alphabetically ordered set I_{body}. Then, the part for the rule's body in the feature vector is filled as follows:

$$r_i = tf(x_i, r) \cdot idf(x_i, R), \quad i = 1, \ldots, b \quad (8)$$

$\vec{r}_{\texttt{head}}$ is treated in the same way, except that x_j is the j-th item of the alphabetically ordered set I_{head}

$$r_{b+j} = tf(x_j, r) \cdot idf(x_j, R), \quad j = 1, \ldots, h \tag{9}$$

VI. Pairwise Similarity

Our association rule feedback approach builds upon a notion of similarity among rules, respectively rule consequences (antecedences). For this reason a similarity measure needs to be selected. As such we have chosen the cosine similarity. It calculates the cosine of the angle between two n-dimensional vectors r and s as follows:

$$sim(\vec{r}, \vec{s}) = \frac{\sum_{i=1}^{n} r_i s_i}{\sqrt{r_i^2}\sqrt{s_i^2}} \tag{10}$$

The cosine similarity compared to other similarity measures, like ones based on the euclidian distance, has the advantage that it does not take missing items in a rule into account. For example, when measuring the similarity between a rule $\mathcal{X}y \rightarrow z$ and its more general rule $\mathcal{X} \rightarrow z$ only the item weights contained in both rules (i.e. $\mathcal{X}$ and z) contribute towards the similarity measure. This property of the cosine measure is also the reason why it is frequently used in information retrieval systems. When comparing, for example, a query with a document it is desirable only to take the actual words contained in the query into account and not each of the many words the user did not specify.

The cosine measure is also suitable as a measure of time series similarity which we use in this paper as an example of further information about rules embedded into the rule vector. For time series the cosine measure has the advantage only to reflect the magnitude of the angle between two vectors but—compared with other distance measures (e.g. Euclidean distance)—to ignore the magnitude difference between the two vectors. This means, it is robust w.r.t. different variation ranges of the time series. It is, however, not robust w.r.t. shifts of the time series mean value. Nevertheless, robustness can be achieved by subtracting from both time series their respective mean value prior to similarity calculation.

Since the cosine measure yields values in $[0, 1]$, we will express the dissimilarity of two vectors as

$$dissim(\vec{r}, \vec{s}) = 1 - sim(\vec{r}, \vec{s}) \tag{11}$$

VII. Similarity Aggregation

So far we have discussed how to calculate pairwise similarities between rules. Nevertheless, for the purpose of relevance feedback it is necessary to measure the similarity of a (unrated) rule r relative to a rule set R which may represent relevant and non-relevant rules.

Generally, we define the similarity of a rule r relative to a rule set $R = \{s_1, \ldots, s_m\}$ as

$$sim_{rs}(\vec{r}, R) = \Omega(\{sim(\vec{r}, \vec{s}_1), \ldots, sim(\vec{r}, \vec{s}_m)\}) \tag{12}$$

whereby Ω denotes a suitable aggregation operator which we will describe in the next section. This score aggregation approach is similar to the ones proposed in other publications, like [5] and [17]. As in Section VI, the dissimilarity of a rule relative to a rule set is defined as

$$dissim_{rs}(\vec{r}, R) = 1 - sim_{rs}(\vec{r}, R) \tag{13}$$

A. The OWA Operator

Our choice of the aggregation operator Ω is guided by two requirements: firstly, the user should be able to influence the aggregation operator, either implicitly or explicitly. Secondly, to obtain comparable results, our aggregation operator should be able to emulate simple aggregation operators like min, max or median.

These two requirements led us to the OWA family of operators, which originate in the Fuzzy Domain and have been introduced by [20]. They are strongly related to the concepts of linguistic quantifiers, such as *many, a few, most*. Nevertheless, [20] presented the connection to linguistic quantifiers, by explaining how the weights that appeared in the OWA expression could be obtained by using the membership function of any quantifier. In subsequent work [21], OWA operators are presented as a way to compute the accomplishment of linguistic quantifiers when used in conjunction with imprecise properties.

An OWA operator Ω is a mapping $\Omega : S \rightarrow \mathbf{R}$, where S is a set of numerical values s_i with $S \neq \varnothing$ and $|S| = n$. The OWA operator Ω has an associated weighting vector $W = (w_1, w_2, \ldots, w_n)^T$ with $w_j \in [0, 1]$ and $\sum_{j=1}^{n} w_j = 1$. It is defined as

$$\Omega(\{s_1, s_2, \ldots, s_n\}) = \sum_{j=1}^{n} w_j b_j \quad , \tag{14}$$

with b_j being the j-th largest of the s_i.

The most important feature of this operator is the ordering of the arguments by value. The OWA operator is in a way very general in that it allows different conventional aggregation operators. This is achieved by appropriately setting the weights in W – different arguments can be emphasised based upon their position in the ordering.

Min, *max*, *mean*, and *median* are special cases for the OWA operator and were described by [22]. They illustrate the generality and flexibility of the OWA operator.

By setting the weights accordingly, the user can influence the relevance score to suit the needs of his particular application scenario.

It should be noted that, as the sets of relevant and non-relevant rules grow, the weight vector of the OWA operator has to grow accordingly. If none of the above special cases of the OWA operator is used, an appropriate weight distribution should be computed. This could be done using a concept similar to a probability density function where mean and variance are specified by the user, according to which of the similarities he would like to emphasise.

B. Relative Importance of Recent Relevance Choices

The retrieval of relevant association rules is a consecutive, iterative process. The user's knowledge, his beliefs and assumptions change during the relevance feedback cycle as he sees new data. Therefore, the user's latest choices should be considered as having a higher priority over the first, relatively uninformed, relevance choices. This concept can be captured as the *decay of a relevant or non-relevant* rule's importance over time. The similarity aggregation should account for this and thus should weight recently selected rules higher than older ones.

Let $t(r)$ be the *age* of a relevant or non-relevant association rule r. This means, $t(r)$ is the number of feedback cycles that have been performed since the rule r was marked as being (non-)relevant, thereby a newly selected rule receives $t = 0$. Let $\delta \in [0, 1]$ a decay constant that controls the speed of decay. Then two possibilities for the *time-weighted importance* τ are as follows:

$$\tau_{exp}(r) = (1-\delta)^{t(r)} \qquad (15)$$

$$\tau_{lin}(r) = max(1 - t(r) \cdot \delta, 0) \qquad (16)$$

with Equation 15 for an exponential type of decay and Equation 16 for a linear decay down to a minimum of zero. This concept can also be described as a kind of *memory* of the relevance feedback engine. The higher the decay factor δ, the faster the system forgets what has been chosen in an earlier step. If we set $\delta = 1$ then our approach would only consider the user's latest relevance decision in its relevance score calculation. The value of $\delta = 0$ would deactivate the decay completely. Values of δ in between those bounds activate a gradual decay. Using the time weighted importance we refine our definition of a rule r its similarity relative to a rule set R and yield

$$sim_{rs}(\vec{r}, R) = \Omega(\{\tau(\vec{s}_1)sim(\vec{r}, \vec{s}_1), \ldots, \tau(\vec{s}_m)sim(\vec{r}, \vec{s}_m)\}) \qquad (17)$$

VIII. Relevance Scoring

Based on the similarity measure we defined in the last section we can develop a notion of a rule's pairwise score, i.e. its relevance score with respect to a certain rule that was marked as relevant. While in information retrieval it is mostly assumed that those documents which are similar to (non-)relevant ones are (non-)relevant too, we use a slightly different approach.

For rules marked as relevant we assume that once a user has seen such a rule rather in being interested in similar ones his attention is attracted by those which are similar in certain features but dissimilar in others. This means, a user aims for rules which have an element of surprise. For example, a rule could have a very similar antecendent, but a rather dissimilar consequent when compared to a relevant one. It would therefore be surprising to a user because it is an exception to his previous knowledge. This approach also captures the case of rule contradiction employed by other authors [8], [12], albeit in a fuzzy, less crisp way.

Table I shows six of such interesting combinations of rule features. The example discussed above is named ω_1 in this table. Another example is ω_2. It assigns a high score to those rules that are very different in their symbolic representation,

similar \ *dissimilar*	head	body	time series	symbolic
head	-	ω_4	ω_5	-
body	ω_1	-	ω_6	-
time series	-	-	-	ω_2
symbolic	-	-	ω_3	-

TABLE I
Interestingness Matrix

but exhibit a similar time series. Such a combination can hint at a unknown hidden cause for the observed changes, which in turn are of interest to a user who typically will assume that only similar rules change similarly. The remaining four entries in Table I can be motivated in a similar way.

For rules marked as non-relevant we use an approach alike the one used in information retrieval, i.e. rules that are similar to non-relevant ones are also considered non-relevant.

Based on these considerations our calculation of the overall relevance score is split into two parts: one each for the relevant and non-relevant rules, respectively.

Our definition of the relevance of a rule with regard to the set of relevant rules is rather straightforward and shown in (18). To allow a user greater flexibility in using one of the pairwise relevance scores shown in Table I we incorporate them all. Their individual contribution to the final relevance score is determined by the weights ω. To pick up on our examples from the previous section, using ω_1 a rule receives a high relevance score if its body is similar to the rule bodies in $R_{\mathtt{r}}$ and its head dissimilar to the rule heads in $R_{\mathtt{r}}$. Likewise, the score for ω_2 is calculated by multiplying the similarity of the rule/rule set combination for the time series with the dissimilarity of the rule/rule set combination for the symbolic representation. For example, by choosing $\omega_1 = 0.5$, $\omega_2 = 0.5$ and $\omega_i = 0$ for $i = 3, \ldots, 6$ we can obtain a score which accounts for both cases of relevance. In most cases, however, it is very likely that only one of the six alternatives for pairwise relevance scores will be employed.

$$\begin{aligned}\Phi(\vec{r}, R_{\mathtt{r}}) = {} & \omega_1 sim_{rs}(\vec{r}_{\mathtt{body}}, R_{\mathtt{r}}) dissim_{rs}(\vec{r}_{\mathtt{head}}, R_{\mathtt{r}}) \\ & + \omega_2 sim_{rs}(\vec{r}_{\mathtt{time}}, R_{\mathtt{r}}) dissim_{rs}(\vec{r}_{\mathtt{sym}}, R_{\mathtt{r}}) \\ & + \omega_3 sim_{rs}(\vec{r}_{\mathtt{sym}}, R_{\mathtt{r}}) dissim_{rs}(\vec{r}_{\mathtt{time}}, R_{\mathtt{r}}) \\ & + \omega_4 sim_{rs}(\vec{r}_{\mathtt{head}}, R_{\mathtt{r}}) dissim_{rs}(\vec{r}_{\mathtt{body}}, R_{\mathtt{r}}) \\ & + \omega_5 sim_{rs}(\vec{r}_{\mathtt{head}}, R_{\mathtt{r}}) dissim_{rs}(\vec{r}_{\mathtt{time}}, R_{\mathtt{r}}) \\ & + \omega_6 sim_{rs}(\vec{r}_{\mathtt{body}}, R_{\mathtt{r}}) dissim_{rs}(\vec{r}_{\mathtt{time}}, R_{\mathtt{r}}) \end{aligned} \qquad (18)$$

For the non-relevant rules we assume that rules in $R_{\mathtt{n}}$ specify a subspace of the rule space where more non-relevant rules are located. To direct the user away from this subspace, rules that are far away from it will receive a higher score, whereas those in the vicinity will receive a low score. An unrated rule r should therefore receive a high interestingness score the more dissimilar it is from the set of non-relevant rules, i.e.

$$\Psi(\vec{r}, R_{\mathtt{n}}) = dissim(\vec{r}, R_{\mathtt{n}}) \tag{19}$$

Our final relevance score of an unrated rule r under consideration of the set of relevant and (non-)relevant rules consists of two parts, $\Phi(\vec{r}, R_{\mathtt{r}})$ and $\Psi(\vec{r}, R_{\mathtt{n}})$, which are both weighted to give the user more influence on the scoring.

$$F(\vec{r}, R_{\mathtt{r}}, R_{\mathtt{n}}) = w_{\mathtt{rel}}\Phi(\vec{r}, R_{\mathtt{r}}) + w_{\mathtt{nrel}}\Psi(\vec{r}, R_{\mathtt{n}}) \tag{20}$$

After every feedback cycle, i.e. after every update of $R_{\mathtt{r}}$ or $R_{\mathtt{n}}$, each unrated rule r is being reevaluated whereby a new score $F(\vec{r}, R_{\mathtt{r}}, R_{\mathtt{n}})$ is assigned. Rules which previously have been ranked as rather non-relevant can now receive a higher score whereas others may lose their relevance.

IX. Conclusion

In this paper, we have dealt with the well-known issue of finding interesting association rules out of a large set of rules. Starting from the finding that interestingness and therefore relevance requires subjectivity, we have tailored our relevance assessment approach to incorporate relevance feedback from the user. Thereby, our findings where guided and inspired by similar problems in the field of information retrieval.

A user's relevance decisions ultimately contain the user's knowledge and assumptions about the domain under consideration. Using rule vectors as numerical representations of association rules we derived a similarity-based notion of relevance which we aggregated to a final relevance score using an OWA operator. The use of the OWA operator provides a user with a high level of flexibility in finding a relevance scoring that suits his application area best. Likewise it is interpretable and easy to understand.

Our relevance scoring approach can be used in a wide range of application scenarios where association rules are involved. In effect, we have created a relevance feedback engine that adapts to each user as he explores the set of association rules.

References

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *Proc. ACM SIGMOD 1993*, pages 207–216, Washington, DC, 1993.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.

[3] R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[4] M. Boettcher, D. Nauck, D. Ruta, and M. Spott. Towards a framework for change detection in datasets. In *Proceedings of the 26th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 115–128. Springer, 2006.

[5] R. Fagin and E. L. Wimmers. A formula for incorporating weights into scoring rules. *Theoretical Computer Science*, 239(2):309–338, 2000.

[6] D. B. Fogel. The advantages of evolutionary computation. In D. Lundh, B. Olsson, and A. Narayanan, editors, *Bio-Computing and Emergent Computation*. World Scientific Press, Singapore, 1997.

[7] T. Jaakkola and H. Siegelmann. Active information retrieval. In *Advances in Neural Information Processing Systems 14*, pages 777–784. MIT Press, 2001.

[8] B. Liu, W. Hsu, and S. Chen. Using general impressions to analyze discovered classification rules. In *Proceedings of the 3rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 31–36, 1997.

[9] B. Liu, W. Hsu, S. Chen, and Y. Ma. Analyzing the subjective interestingness of association rules. *IEEE Intelligent Systems*, 15(5):47–55, 2000.

[10] B. Padmanabhan and A. Tuzhilin. Unexpectedness as a measure of interestingness in knowledge discovery. *Decision Support Systems*, 27, 1999.

[11] B. Padmanabhan and A. Tuzhilin. Small is beautiful: discovering the minimal set of unexpected patterns. In *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 54–63, 2000.

[12] B. Padmanabhan and A. Tuzhilin. Knowledge refinement based on the discovery of unexpected patterns in data mining. *Decision Support Systems*, 33(3):309–321, 2002.

[13] G. Piatesky-Shapiro and C. J. Matheus. The interestingness of deviations. In *Proceedings AAAI workshop on Knowledge Discovery in Databases*, pages 25–36, 1994.

[14] G. Salton. *The SMART Information Retrieval System*. Prentice Hall, Englewood Cliffs, NJ, 1971.

[15] G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 5(24):513– 523, 1987.

[16] A. Silberschatz and A. Tuzhilin. What makes patterns interesting in knowledge discovery systems. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):970–974, 1996.

[17] P. K. C. Singitham, M. S. Mahabhashyam, and P. Raghavan. Efficiency-quality tradeoffs for vector score aggregation. In *VLDB*, pages 624–635, 2004.

[18] P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the right objective measure for association analysis. *Information Systems*, 29(4):293–313, 2004.

[19] K. Wang, Y. Jiang, and L. V. S. Lakshmanan. Mining unexpected rules by pushing user dynamics. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 246–255, 2003.

[20] R. R. Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Trans. Syst. Man Cybern.*, 18(1):183–190, 1988.

[21] R. R. Yager. Fuzzy quotient operator. In *Proceedings of the Fourth International Conference on Information Processing and Management of Uncertainty*, pages 317–322, Palma de Majorca, Spain, 1992.

[22] R. R. Yager. On the inclusion of importances in owa aggregations. In *The ordered weighted averaging operators: theory and applications*, pages 41–59, Norwell, MA, USA, 1997. Kluwer Academic Publishers.