An Apriori-like algorithm for Extracting Fuzzy Association Rules between Keyphrases in Text Documents

Abstract

In this paper we present an algorithm for extracting fuzzy association rules between weighted keyphrases in collections of text documents. First, we discuss some classical approaches to association rule extraction and then we show the fuzzy association rules algorithm. The proposed method integrates the fuzzy set concept and the apriori algorithm. The algorithm emphasizes the distinction between three important parameters: the support of a rule, its strength, and its confidence. It searches for rules containing different number of phrases and having confidence level and strength level above certain thresholds. The algorithm makes the distinction between a small number of occurrences with high support intersections and large number of occurrences with low support intersections. Finally we present results of initial experiments on real-world data that illustrate the usefulness of the proposed approach.

Keywords: Association Rules, Fuzzy Logic, Text Mining.

1 Introduction

The goal of fuzzy association rule mining process in a given collection (corpus) of text documents is to discover interesting rules of the type $A \rightarrow B$, where $A$ and $B$ are sets of phrases associated with some membership functions. The intuitive meaning of such a rule is that documents in the corpus that contain $A$ also tend to contain $B$. Agrawal et al. [1] designed some algorithms for generating association rules among items in a transaction database. The significance of every rule is determined by its support and confidence. The support is the proportion of documents in the database where both $A$ and $B$ occur together. The confidence is the proportion of documents in the database where both $A$ and $B$ occur together. The confidence is the proportion of documents in the database having $B$ given that they also have $A$.

The data mining process consists of two phases. In the first phase, all candidate itemsets (combinations of some items) are found, and support is calculated for each of them. Those whose support is above a certain threshold (minimum support) are called "frequent itemsets", and used to find larger itemsets. In the second phase, association rules are formed from the frequent itemsets: for each frequent itemset, the confidence value is computed for all the combinations of the prefix and postfix of the
rule (A and B respectively, which both are distinct subsets of the large itemset). Those with confidence level above a certain threshold (called minimum confidence) are displayed as interesting association rules.

In this paper we present a new method for extracting fuzzy association rules between weighted key phrases in text documents. Each keyphrase is associated with five membership grades computed from applying different membership functions to normalized term frequencies multiplied by the weights (numerical scores) calculated with the Extractor Content Summarization Package [2] [6]. The Extractor score is the number of times the phrase appears in the document, multiplied by a factor and adjusted by the number of words in a phrase [6].

Our algorithm extracts rules for itemsets containing variable number of items. The rules are scored by three parameters: support, strength and confidence. Thus we make the distinction between a small number of occurrences with high support intersections and a large number of occurrences with low support intersections.

This paper is organized as follows: In Section 2 we describe the method for determining fuzzy values using five different membership functions. The Fuzzy Rule Extraction Algorithm is presented in Section 3. A detailed example is shown in Section 4. Section 5 presents the results of a real-world case study. Finally, conclusions are given in Section 6.

2 Determining Fuzzy Values

The input to the algorithm is a collection of text documents. First we use the Extractor Software Development Kit [2] to extract keyphrases and their respective weights from each document, then we calculate the raw in-document frequencies of extracted keyphrases, and finally apply five different membership functions to each keyphrase. In order to have a smoother distribution of weights and fuzzy values, we normalize the frequencies by dividing the raw frequency of each phrase by the maximum phrase frequency in the same document. Then we multiply the normalized frequency by the Extractor score. Figures 1-2 compare the distribution of normalized weights to distribution of the Extractor weights (scores) in a collection of 5,555 distinct documents downloaded from WWW (Corpus A). Figures 3-4 compare the distributions in a different collection of 579 distinct documents (Corpus B). The overlap between the vocabularies of the two collections is minimal.

It can be seen in both cases that normalizing the weights makes the distribution smoother and therefore the values of the three parameters characterizing each extracted rule (confidence, support, and strength) will have higher variance in a given collection of documents.

![Figure 1: Distribution of Extractor weights in Corpus A](image1)

![Figure 2: Distribution of normalized weights in Corpus A](image2)
Fuzzy data have uncertain values associated with fuzzy (linguistic) terms (such as "high" and "low") and a membership function, which maps the design parameter to the range between 0 and 1. In this paper, we consider membership functions of a trapezoidal shape [3], and denote them by $MF(a, b, c, d)$, where the parameters mark the endpoints of the shape. A triangular function is a special case, where the parameters $b$ and $c$ are equal: $MF(a, b, b, d)$. In this paper, we apply five different membership functions to occurrence of each keyphrase in a given document: very low, low, medium, high, and very high. The medium function is of a triangular shape, while the four others are of a trapezoidal shape. Each two adjacent functions have a common range of normalized weights, where both of them return a non-zero value as the membership grade of the weight. The five membership functions are shown schematically in Figure 5.

### Table 1: Notation

<table>
<thead>
<tr>
<th>NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_n$</td>
<td>Number of documents in the collection</td>
</tr>
<tr>
<td>$a_n$</td>
<td>Number of distinct keyphrases in the collection</td>
</tr>
<tr>
<td>$m_n$</td>
<td>Number of membership functions</td>
</tr>
<tr>
<td>$D_i$</td>
<td>The $i^{th}$ transaction (document)</td>
</tr>
<tr>
<td>$P_j$</td>
<td>The $j^{th}$ attribute (phrase)</td>
</tr>
<tr>
<td>$M_k$</td>
<td>The $k^{th}$ membership function</td>
</tr>
<tr>
<td>$F_{i,j}$</td>
<td>Frequency of phrase $P_j$ in document $D_i$</td>
</tr>
<tr>
<td>$F_i$</td>
<td>Maximum frequency of phrase in document $D_i$</td>
</tr>
<tr>
<td>$NF_{i,j}$</td>
<td>Normalized frequency of phrase $P_j$ in document $D_i$</td>
</tr>
<tr>
<td>$E_{i,j}$</td>
<td>Extractor numerical score of phrase $P_j$ in document $D_i$</td>
</tr>
<tr>
<td>$I$</td>
<td>Set of items (itemset). Each itemset is characterized by two fields: i) keyphrase, ii) membership function (e.g., &quot;web graphics&quot;: low)</td>
</tr>
<tr>
<td>$r$</td>
<td>Number of items (keyphrase – membership function pairs) in an itemset $I$</td>
</tr>
<tr>
<td>$\mu_{i,j,k}$</td>
<td>Fuzzy value of the $k^{th}$ membership function for phrase $P_j$ in document $D_i$</td>
</tr>
<tr>
<td>$\mu_i(I)$</td>
<td>Fuzzy value of itemset $I$ in document $D_i$</td>
</tr>
<tr>
<td>$Num(I)$</td>
<td>Frequency of an itemset $I$ in the whole corpus (number of documents)</td>
</tr>
<tr>
<td>$Sup(I)$</td>
<td>Sum of fuzzy values of occurrences of the phrases in itemset $I$ in the whole corpus</td>
</tr>
</tbody>
</table>
3.1 The Algorithm

Input: Collection of text documents.
Output: Set of association rules between document keyphrases.

1. Generate the candidate set \( C_1 \)
   \[
   C_1 = \{ (P_j, M_k) \mid 1 \leq j \leq t, n, 1 \leq k \leq m, n \}
   \]

2. For each transaction \( D_i \)
   For each attribute \( P_j \)
   For each membership function \( M_k \)
   \[
   \mu_{i,j,k} = \begin{cases} \frac{E_{i,j} \times NF_{i,j} - a_i}{b_i - a_i} & a_i \leq E_{i,j} \times NF_{i,j} < b_i \\ 1 & b_i \leq E_{i,j} \times NF_{i,j} < c_i \\ \frac{E_{i,j} \times NF_{i,j} - d_i}{c_i - d_i} & c_i \leq E_{i,j} \times NF_{i,j} < d_i \\ 0 & \text{otherwise} \end{cases}
   \]

3. For each 1-itemset \( c \in C_1 \)
   \[
   \text{num}(P_j, M_k) = \sum_{i=1}^{t} \mu_{i,j,k} > 0
   \]
   If \( \text{num}(P_j, M_k) \geq \text{min}_\text{num} \) then add \( (P_j, M_k) \) to \( L_i \)

4. For \( r = 2; L_{r-1} \neq \emptyset; r++ \) do
   Begin
   4.1 \( C_r \) = Set of new \( r \)-itemset candidates
   4.2 For each \( c \in C_r \) do
   Begin
   \[
   \mu(c) = \mu_{i, c_i} \cap \mu_{i, c_{i-1}} \cap \ldots \cap \mu_{i, c_1}
   \]
   \[
   \text{num}(c) = \sum_{i=1}^{t} \mu(c) \subseteq D_i > 0
   \]
   If \( \text{num}(c) \geq \text{min}_\text{num} \) then
   Begin
   \[
   \text{sup}(c) = \sum_{i=1}^{t} \mu(c) | c \subseteq D_i
   \]
   \[
   \text{str}(c) = \frac{\text{sup}(c)}{\text{num}(c)}
   \]
   If \( \text{str}(c) \geq \text{min}_\text{str} \) then
   \[\text{R}_r.\text{add}(\text{find_rules}(c, r))\]
   End
   End
   End
   5. Answer = \( \bigcup R_r \)

find_rules (itemset \( c \), integer \( r \))
For \( l = 1; l < r; l++ \) do
Begin
\( G_i = \) all groups of different \( l \) items from \( c \)
For each \( \{i_1, \ldots, i_l\} \in G_i \) do
Begin
Build rule \( k \)
\[
\{i_1, \ldots, i_l\} \Rightarrow \{i_{l+1}, \ldots, i_r\}
\]
\[
\{i_1, \ldots, i_l\} \cup \{i_{l+1}, \ldots, i_r\} = c,
\]
\[
\{i_1, \ldots, i_l\} \cap \{i_{l+1}, \ldots, i_r\} = \emptyset
\]
\[
\text{conf}(k) = \frac{\text{num}(r)}{\text{num}(\{i_1, \ldots, i_l\})}
\]
If \( \text{conf}(k) \geq \text{min}\text{conf} \) then
Add the rule \( k: l \Rightarrow (r-l) \) to \( R \)
End
End
End
End

Explanation. The algorithm starts with creating all possible permutations of phrase-membership function pairs, and adding them to the candidate set \( C_1 \). Next, \( m \times n \) fuzzy values are calculated for each occurrence of phrase in document, by applying the membership function to the normalized weight of each occurrence. For each 1-itemset from \( C_1 \) consisting of attribute \( P_j \) and membership function \( M_k \), the number of occurrences in the whole corpus, where \( \mu_{i,j,k} > 0 \) is counted. The result is \( \text{num}(P_j, M_k) \). If the result is bigger than the parameter \( \text{min}_\text{num} \) (predefined minimum number of occurrences), then the itemset is added to the set of large 1-itemset \( L_1 \). At step 4, a loop is started with \( r = 2 \) (\( r \) is the number of items that compose the itemsets in the current stage). A subsequent
iteration consists of two phases. First, the candidate set \( C_r \) is generated from the set of all large \((r-1)\)-itemsets \( L_{r-1} \) similarly to apriori candidate generation function [1]: If two \((r-1)\)-itemsets share only their first items (both phrase and membership function are the same), and the last one is different, a new itemset, which is the union of the two itemsets above is added to \( C_r \). We would like to prevent generating itemsets, where some membership functions belonging to the same attribute exist simultaneously in the same itemset (for example: \( \{(P_1,M_1);(P_1,M_2)\} \)), due to the fact that it may lead to meaningless rules (\( P_1 \) is both high and low in the same rule). Moreover, it is insufficient that only the last items of the two itemsets above will be different in order to reject meaningless rules. Therefore during the generation of new itemsets, the following condition has to be added: a new itemset is generated from two \((r-1)\)-itemsets \( \{(P_1,M_1);…;(P_{r-1},M_{r-1})\} \) and \( \{(Q_1,M_1);…;(Q_{r-1},M_{r-1})\} \) only if the last phrase in the first itemset is different from the last phrase in the second itemset: \( P_{r-1} \neq Q_{r-1} \).

Next, in the pruning step, we remove all itemsets from \( C_r \) such that some \((r-1)\)-subset of them is not in \( L_{r-1} \).

In the second phase of each subsequent iteration, the corpus is scanned and the fuzzy value of each candidate \( r \)-itemset \( c \) is calculated as the intersection of all fuzzy values of the \( r \)-items which compose it. Next, the number of documents in the whole corpus, which contain all the items in \( c \), and having \( \mu_i(c) > 0 \) is counted. The result is \( \text{num}(c) \). If the obtained result is bigger than \( \text{min_num} \) then the support and the strength of itemset \( c \) are calculated and it is added to the set of large \( r \)-itemsets \( L_r \). The support of \( c \) is the sum of intersections of fuzzy values of occurrences of \( c \) in the whole corpus. The strength of \( c \) is the division of its support by its frequency for the documents, where the fuzzy value is greater than 0. If \( \text{str}(c) \) is bigger than \( \text{min_str} \) then the function \( \text{find_rules} \) finds possible association rules between the items in \( c \).

### 4 Detailed Example

Consider a small database of Extractor weights and in-document frequencies of keyphrases in Table 2. Each item in a given document, identified by TID, is shown as a set of three numbers: Phrase ID (Extractor Weight, Raw Frequency). The sample corpus in Table 2 consists of 6 different documents and 8 distinct keyphrases. Assume that the minimum number of occurrences (\( \text{min_num} \)) is 2, and the predefined values of \( \text{min_str} \) and \( \text{minconf} \) are 0.8 and 0.9 respectively. There are five different membership functions, thus there are 40 \( (= 8 \times 5) \) items of phrases and membership grades in the initial set of candidate \( 1 \)-itemsets \( C_1 \). The five functions are Very-Low, Low, Medium, High and Very-High.

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
<th>Max freq</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 (4.18,5) ; 2 (4.15,3) ; 3 (15.23,3) ; 4 (6.297,7) ; 5 (22.12,4)</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>1 (5.12,1) ; 3 (6.19,1) ; 4 (4.44,1) ; 5 (15.2,1)</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2 (1.78,13) ; 3 (8.57,10) ; 6 (2.1,7)</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>3 (3.1,1) ; 4 (15.5) ; 5 (21.5) ; 6 (1.15,1)</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>1 (5.11,1) ; 3 (6.23,1) ; 4 (6.676,1) ; 5 (15.7,1)</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>3 (6.28,7) ; 4 (7.92,6) ; 5 (13.95,6) ; 7 (15.1,5) ; 8 (5.1,5)</td>
<td>7</td>
</tr>
</tbody>
</table>
Table 3: Membership grades and frequencies of items associated with the first five phrases

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>VL</td>
<td>0.41</td>
<td>0.99</td>
<td>0.89</td>
<td>0</td>
<td>0.94</td>
<td>0</td>
</tr>
<tr>
<td>L</td>
<td>0</td>
<td>0.9</td>
<td>0.5</td>
<td>0.05</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>M</td>
<td>0</td>
<td>0</td>
<td>0.89</td>
<td>0</td>
<td>0</td>
<td>0.97</td>
</tr>
<tr>
<td>VH</td>
<td>0.91</td>
<td>0.49</td>
<td>0.02</td>
<td>0.99</td>
<td>0.85</td>
<td>0</td>
</tr>
<tr>
<td>H</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.63</td>
</tr>
<tr>
<td>num</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>

In order to determine the arguments of the membership function, we normalize the frequency and Extractor weight of each occurrence of phrase as described in Section 2. The histogram in Figure 6 demonstrates the cumulative percentage of the normalized weights.

The arguments of the five functions were chosen so that there will be a symmetrical distribution of normalized weights between the corresponding functions. For example: 28% of weights will get a non-zero value from VL (point A), and the same percentage of weights will get a non-zero value from VH (point B). Consequently, the following functions were obtained: \( VL(0,0,1.5,4); L(2,3,5,6.25); M(4,6,25,6.25,9); H(6,25,6.3,6.6,12.6); VH(6.8,15,21,\infty) \).

Five membership grades refer to each keyphrase in a given document. Some of these values are shown in Table 3, where each row represents a document and each column stands for an item associated with a given keyphrase.

It is easy to see that all items shown in Table 3 are added to \( L_i \), except 1 VL, 4L, 4VH, 5H (since \( \min_{num} = 2 \)). The loop at step 4 starts with the initialization of \( r=2 \), and the candidate set \( C_2 \) is generated from the items in \( L_i \). As an example, the itemsets \( 1_L3_M; 1_L5_{VH}; 2_{VL}3_{H}; 2_{VL}4_M; 3_M5_{VH}; 4_{H}5_{VH} \) are added to \( C_2 \). The intersection fuzzy value of each occurrence of itemset in \( C_2 \) is calculated at step 4.2, and the number of non-zero fuzzy values is counted for each itemset. Table 4 shows the intersection fuzzy values and the parameter \( num \) of the six itemsets above. Here the minimum operator is used for intersection. Note that the calculated value of \( num \) for five itemsets of the six above is larger than or equal \( \min_{num} \) (except \( 2_{VL}3_{H} \), and therefore those five itemsets are added to the set of large 2-itemsets \( L_2 \). In addition, the values of the support and strength are calculated for each large itemset.

Table 4: Intersection fuzzy values and summarizing parameters for itemsets in \( L_2 \).

<table>
<thead>
<tr>
<th></th>
<th>1_L</th>
<th>1_L</th>
<th>2_VL</th>
<th>2_VL</th>
<th>3_M</th>
<th>3_M</th>
<th>4_H</th>
<th>4_H</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.9</td>
<td>0.71</td>
<td>0.89</td>
<td>0.89</td>
<td>0.71</td>
<td>0.71</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.9</td>
<td>0.9</td>
<td>0</td>
<td>0</td>
<td>0.97</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0.88</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0.91</td>
<td>0.91</td>
<td>0</td>
<td>0</td>
<td>0.99</td>
<td>0.99</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.63</td>
<td>0.63</td>
<td></td>
<td></td>
</tr>
<tr>
<td>num</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sup</td>
<td>2.71</td>
<td>2.52</td>
<td>1.77</td>
<td>3.3</td>
<td>2.33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>str</td>
<td>0.9</td>
<td>0.84</td>
<td>0.89</td>
<td>0.83</td>
<td>0.78</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Since the strengths of the itemsets 1L3M, 1L5VH, 2VL4M, 3M5VH are larger than or equal to min_str, possible association rules with two items are searched. From the large 2-itemset 2VL4M the association rules 2VL⇒4M and 4M⇒2VL are generated with confidence values 1 and 0.4 respectively. Since only the first one is larger than minconf, the rule 2VL⇒4M is added to R2 with num=2, sup=1.77, str=0.89, conf=1.

L2 is not empty, meaning the condition of the loop at step 4 is satisfied. Thus a set of candidate 3-itemsets is generated. For example, the candidate itemset 1L3M5VH is generated from the two itemsets 1L3M and 1L5VH. Since the subset itemset 3M5VH is in L2, the 3-itemset above is not deleted from C3 at the pruning step. The value of num (1L3M5VH) is 3, so it is added to L3 with support=2.52, str=0.84. Now possible association rules with three items from L3 are searched. The rule 1L3M⇒5VH is added to R3 with confidence=1, but the rule 3M5VH⇒1L is not added, since its confidence is 0.75, and it is less than minconf.

When we generate C4 using L3, it turns out to be empty, we return all discovered rules and terminate the algorithm.

5 Real-World Experiments

To evaluate the performance of our algorithm on real-world document collections, we performed a series of experiments using two different corpuses. In order to generate datasets with various characteristics, we used Extractor software described in [6], which returned for each document the most important keyphrases phrases (1-3 words each) with their raw frequency and Extractor weight (numerical score).

The first corpus (Corpus A) is composed of 5555 distinct web documents (of 13330 total documents) containing 38776 distinct phrases. The second corpus (Corpus B) is composed of 579 distinct web documents (of 582 total documents) containing 5275 distinct phrases. Due to space limitations, we only report the results on Corpus A in this paper. Figures 7-9 depict the influence of the parameters min_num and min_str on the number of extracted rules and the number of frequent itemsets.

Figure 7 contains experimental results of the number of extracted rules while varying the parameter min_str; the parameters min_num and minconf were constant with values of 10 and 0.85 respectively. It can be seen that by increasing the minimum strength by 20%, the total number of extracted rules has decreased by 75%, hence this parameter has a significant influence on the results. It can be noticed from Figure 8 that most of the found itemsets were of size 5 (containing 5 items), disregarding the value of min_str.
Figure 9 shows that the parameter \textit{min\_num}, which is the minimum number of occurrences in the whole corpus, is inversely proportional to both the number of extracted rules and the number of found itemsets. Moreover, the average number of rules obtained from each itemset decreases as the number of found itemsets increases.

![Figure 9: Total number of found itemsets and number of extracted rules as a function of \textit{min\_num}.

6 Conclusions

In this paper we have presented a novel algorithm for extracting fuzzy association rules between keyphrases in collections of text documents. The keyphrases are extracted using the Extractor Package and then scored based on their in-document frequencies and Extractor weights. Items are defined as combinations of keyphrases and their membership grades in five fuzzy sets. The algorithm usefulness was demonstrated on two corpuses of real-world web documents. In the future, we plan to explore the potential benefits of this algorithm for some common information retrieval tasks.

Acknowledgements

This work was partially supported by the National Institute for Systems Test and Productivity at University of South Florida under the USA Space and Naval Warfare Systems Command Grant No. N00039-01-1-2248 and by the Fulbright Foundation that has awarded Prof. Kandel the Fulbright Senior Specialists Grant at Ben-Gurion University of the Negev in November-December 2005.

References


