Frequent Closed Itemset Mining problem

Oskar Gross
Karl Blum
Articles

1. CLOSET: An Efficient Algorithm for Mining Frequent Closed Itemsets
2. Generating a Condensed Representation for Association Rules
3. CHARM: An Efficient Algorithm for Closed Itemset Mining
Finding association rules

1. Find all *frequent itemsets*

2. For each frequent itemset $I_1$ found, generate all association rules

$$I_2 \xrightarrow{c} I_1 - I_2 \quad \text{where} \quad I_2 \subseteq I_1$$
Three trends in frequent itemset mining

1. Levelwise – breadth-first exploration; good for weakly correlated data
2. Maximal frequent itemsets
3. Frequent closed itemsets**** – fewer itemsets in dense or correlated data;
Min-Max Association Rules

• Association Rule Redundancy

• Most general non-redundant AR

• Minimal antecedent -> Maximal consequent

• Less noise, more understandable association rules

• MinMaxApprox, MinMaxExact, MinMaxReduc
Relevance and usefulness of association rule

Let's assume we have a database with two transactions
\{(a_1, a_2, \ldots, a_{100}), (a_1, a_2, \ldots, a_{50})\}

\text{min\_sup} = 1

\text{min\_conf} = 50\%
Relevance and usefulness of association rule

Let's assume we have a database with two transactions
\{(a_1, a_2, ..., a_{100}), (a_1, a_2, ..., a_{50})\}
\min\_sup = 1
\min\_conf = 50\%

Traditional method (1. trend) will generate:

\[2^{100} - 1 \approx 10^{30}\]

We get only one association rule when using closed frequent itemset mining.
CLOSET

1. Find frequent items (f_list)
2. Divide search space
3. Construct conditional databases
4. Find subsets of frequent closed itemsets and mine recursively
CLOSET optimizations

1. Using FP-tree structure for transactional and conditional databases
2. Extract items appearing in every transaction of conditional database
3. Directly extract frequent closed itemsets from FP-tree
4. Prune search branches
CHARM

• Closed itemset mining

• IT pair (Itemset-Tidset (cover) pair)

• Example DB (on blackboard)

• Frequent, Closed and Maximal itemsets
IT-Tree

- Example Tree on blackboard
- CHARM is based on 4 properties of IT pairs
- Theorem on next slide
IT-Pair Properties

Let \( X_i \times t(X_i) \) and \( X_j \times t(X_j) \) be any two members of a class \([p]\), with \( X_i \preceq_f X_j \), where \( f \) is a total order. The following four properties hold:

1. If \( t(X_i) = t(X_j) \), then \( c(X_i) = c(X_j) = c(X_i \cup X_j) \)
2. If \( t(X_i) \subset t(X_j) \), then \( c(X_i) \neq c(X_j) \), but \( c(X_i) = c(X_i \cup X_j) \)
3. If \( t(X_i) \supseteq t(X_j) \), then \( c(X_i) \neq c(X_j) \), but \( c(X_j) = c(X_i \cup X_j) \)
4. If \( t(X_i) \neq t(X_j) \), then \( c(X_i) \neq c(X_j) \neq c(X_i \cup X_j) \)
CHARM in Action

• Initialize IT pairs

• Add next node to the following node

• Check which property applies

• If closed set found then add to list
Subsumption Checking

• Itemset $X$ subsumes another $Y$ itemset iff

$$X_j \subseteq X_i \text{ and } \sigma(X_j) = \sigma(X_j)$$

• Comparing all elements takes too many operations

• Let’s create a hash table $\rightarrow$ in this case sum of the tids in the tidset

• We query all sets from table with key $h(X)$

• Then compare supports
Diffsets for Fast Frequency Computations

• Diffset example on blackboard

• Diffsets cut down the size of memory required

• Calculating support with diffset
  \[ \sigma(XY) = \sigma(X) - d(XY) \]

• CHARM algorithm with diffset example on blackboard
Four Properties with Diffsets

When intersecting two tidsets we keep track of the number of mismatches in both the lists, i.e., the cases when a tid occurs in one list but not in the other. Let $m(X_1)$ and $m(X_2)$ denote the number of mismatches in the tidsets for itemsets $X_1$ and $X_2$. There are four cases to consider:

1. $m(X_1) = 0$ and $m(X_2) = 0$, then $t(X_1) = t(X_2)$ — Property 1
2. $m(X_1) = 0$ and $m(X_2) \neq 0$, then $t(X_1) \subset t(X_2)$ — Property 2
3. $m(X_1) \neq 0$ and $m(X_2) = 0$, then $t(X_1) \supset t(X_2)$ — Property 3
4. $m(X_1) \neq 0$ and $m(X_2) \neq 0$, then $t(X_1) \neq t(X_2)$ — Property 4

For $t(A)$ and $t(D)$ from above, $m(A) = 2$ and $m(D) = 2$, and as we can see, $t(A) \neq t(D)$. Next consider $t(A) = 1345$ and $t(W) = 12345$. We find $m(A) = 0$, but $m(W) = 1$, which shows that $t(A) \subset t(W)$. Thus CHARM performs support, subset, equality, and inequality testing simultaneously while computing the intersection itself.
Initialization Optimisation

• Many itemsets of length 2 are infrequent

• First of all compute frequent itemsets of length 2

• We combine two itemsets X and Y only if their union is frequent
Performance