MỘT KHẢO SÁT VỀ HAUIM

● PHẠM TUẤN KHIÊM

TÓM TẮT:

Việc trích xuất hiệu quả các mẫu có giá trị từ các bộ dữ liệu quy mô lớn ngày càng trở nên cần thiết trong các ứng dụng khai thác dữ liệu khác nhau. Khai thác tập hữu ích trung bình cao (HAUIM) là một nhiệm vụ khai thác dữ liệu quan trọng liên quan đến việc xác định các tập mục có giá trị hữu ích cao. Trong bài báo này, tác giả khám phá toàn diện tình trạng nghiên cứu hiện tại về HAUIM, bao gồm các thuật toán và tính năng của nó. Tác giả thảo luận về những tiến bộ đáng kể, những thách thức và hướng đi tiềm năng tương lai trong lĩnh vực này, cung cấp những hiểu biết sâu sắc có giá trị về bối cảnh ngày càng tăng của hoạt động khai thác tập hữu ích cao.

Từ khóa: khai tập hữu ích, khai thác dữ liệu, độ hữu ích, tập hữu ích trung bình cao.

1. Đặt vấn đề

Khai thác tập phổ biến (FIs) trong cơ sở dữ liệu giao dịch là bài toán phổ biến và có nhiều ứng dụng trong việc khám phá tri thức trong cơ sở dữ liệu [1, 2]. Nhiều thuật toán đã được đưa ra để giải quyết vấn đề này. Trong đó, cách tiếp cận thường dùng là mở rộng tập mẫu (Pattern-Growth) [3, 4]. FP-growth (Frequent Pattern Growth) ban đầu xây dựng cấu trúc FP-tree bằng cách sử dụng tập phổ biến 1 phần tử. Sau đó, trong quá trình khai thác, các cây FP-trees được tạo đệ quy và mỗi cây chứa một bảng chỉ mục được thiết kế để khai thác các tập phổ biến. Khai thác tập hợp phổ biến truyền thống chỉ xem xét tần suất xuất hiện của các tập mặt hàng trong cơ sở dữ liệu giao dịch nhưng lại bỏ qua các yếu tố quan trọng khác như số lượng, lợi nhuận và trọng lượng của các mặt hàng. Một vấn đề khác là FIs không xem xét các tập mục không phổ biến nhưng lại có thể đóng góp một lượng lớn lợi nhuận. Do đó, nếu chỉ xem xét tần suất xuất hiện là không đủ để xác định các tập phổ biến có lợi nhuận cao. Chính vì vậy, khai thác tập hữu ích cao (HUIM - high utility itemset mining) [5, 6, 7] đã được nghiên cứu trong thời gian gần đây và có những kết quả nhất định. Việc khai thác tập hữu ích cao có thể xuất hiện nhiều tập mặt hàng có số phần tử lớn là tập hữu ích cao, trong đó chỉ có một số mặt hàng có ý nghĩa thực tế về độ hữu ích cao, trong khi các mặt hàng khác trong tập có độ hữu ích thấp nên không có ý nghĩa trong thực tế. Điều này có thể gây khó khăn trong việc phân tích kinh doanh từ tập kết quả.

Khai thác tập hữu ích trung bình cao (HAUIM) đã thu hút được sự chú ý đáng kể trong những năm gần đây do vai trò quan trọng của nó trong việc khám phá các mẫu có giá trị tiện ích cao thuộc cơ sở dữ liệu giao dịch. Không giống như khai thác tập mục truyền thống, HAUIM tập trung vào việc trích xuất các tập mục dựa trên độ hữu ích trung bình của chúng thay vì hỗ trợ. Giá trị độ hữu ích thường gắn liền với tầm quan trọng hoặc lợi nhuận, do một tập

mục tạo ra. Bài báo này nhằm mục đích cung cấp cái nhìn tổng quan toàn diện về các kỹ thuật và thuật toán khác nhau được phát triển cho HAUIM, nêu bật điểm mạnh, điểm yếu và ứng dụng tiềm năng của chúng.

2. Phương pháp nghiên cứu tập hữu ích trung bình cao

2.1. Các khái niệm

Cho cơ sở dữ liệu giao dịch D = {T_1, T_2, ..., T_n}, với n là số lượng giao dịch D trong tập I = {i_1, i_2, ..., i_m} gồm m mục phân biệt trong D. $\forall T_j \in D$, $T_j = \{x_i | l = 1,2, \dots, N_j, x_i \in I\}$, , với N_j là số các mục trong giao dịch T_j. Bảng 1 biểu diễn ví dụ về cơ sở dữ liệu giao dịch D dùng để khai thác tập hữu ích trung bình cao, mỗi mục (cột 2 của Bảng 1) trong giao dịch (cột 1 của Bảng 1) có một số lượng (cột 3 của Bảng 1). Bảng 2 biểu diễn giá trị lợi nhuận của các mục.

Bảng 1. Cơ sở dữ liệu giao dịch

TID	Các mục	Số lượng
1	a, c, d, e	4, 2, 2, 1
2	a, b, c, e, f	7, 4, 7, 5, 1
3	a, b, d, f	1, 4, 1, 1
4	a, b, e, f	4, 7, 1, 1
5	b, c, e	2, 1, 2
6	a, b, c	8, 3, 8
7	c, d, e	3, 1, 1

Nguồn: Tác giả thực hiện

Bảng 2. Lợi nhuận các mục

Các mục	a	b	c	d	e	f
Lợi nhuận	3	1	5	4	6	2

Nguồn: Tác giả thực hiện

Mục tiêu của việc khai thác tập hữu ích trung bình cao là tìm trong cơ sở dữ liệu giao dịch tất cả các tập có độ hữu ích trung bình không nhỏ hơn ngưỡng độ hữu ích nhỏ nhất cho trước. Các định nghĩa sau được sử dụng trong các nghiên cứu trước đây về khai thác tập hữu ích trung bình cao.

Định nghĩa 1. Độ hữu ích của mục *i* trong giao dịch *T* được định nghĩa là: $u(i,T) = p(i) * q(i,T)$. Trong đó $p(i)$ là lợi nhuận của mục i, $q(i,T)$ là số lượng mua của mục i trong giao dịch *T*.

Ví dụ: Độ hữu ích của mục *b* trong T_2 là:

$$u(b,T_2) = p(b) * q(b, T_2) = 1 * 4 = 4$$

Định nghĩa 2. Độ hữu ích trung bình (Average Utility). Gọi $l(X)$ là số phần tử của tập mục *X*, độ hữu ích trung bình của *X* trong *T* ký hiệu là $au(X,T)$, được xác định:

$$au(X,T) = \frac{\Sigma_{i \in X \subseteq T} \; u(I,T)}{l(X)} \qquad (1)$$

Ví dụ: Độ hữu ích trung bình của tập mục {*b, c*} trong T_2 là:

$$au(bc, T_2) = \frac{iu(b,T_2) + iu(c,T_2)}{l(bc)}$$

$$= \frac{1 * 4 + 5 * 7}{2} = 19.5$$

Độ hữu ích trung bình của *X* trong *D*, ký hiệu là $au(X)$, được định nghĩa:

$$au(X) = \Sigma_{X \subseteq T \in D} \; au(X,T) \qquad (2)$$

Ví dụ: Độ hữu ích trung bình của tập mục {*b, c*} trong Bảng 1 là:

$$au(bc) = au(bc, T_2) + au(bc, T_5) + au(bc, T_6)$$

$$= \frac{1 * 4 + 5 * 7}{2} + \frac{1 * 2 + 5 * 1}{2}$$

$$+ \frac{1 * 3 + 5 * 8}{2}$$

$$= 19.5 + 3.5 + 21.5 = 44.5$$

Định nghĩa 3. Tập mục *X* được gọi là tập hữu ích trung bình cao (High Average Utility Itemset - HAUI) nếu độ hữu ích trung bình của *X* lớn hơn hoặc bằng giá trị ngưỡng độ hữu ích tối thiểu (minUtil). Với minUtil được cho trước bởi người dùng, ta có: $HAUIs = \{X \mid au(X) \geq minUtil\}$ [8].

Ví dụ: Với minUtil = 42. Tập {*b, c*} trong Bảng 1 là tập hữu ích trung bình cao, vì $au(bc) = 44.5 > 42$.

2.2. Các thuật toán

Phần này tác giả trình bày về các thuật toán đã được nghiên cứu về khai thác tập hữu ích tring bình cao. Mỗi thuật toán đều có mô tả phương pháp thực hiện, phân tích các ưu điểm và nhược điểm của từng thuật toán, để từ đó có thể đề xuất hướng nghiên cứu mới dựa trên những đặc điểm thu được ở mỗi thuật toán, cũng như dựa trên việc khắc phục từ các nhược điểm của mỗi thuật toán.

2.2.1. Hướng tiếp cận apriori

Phương pháp này dựa trên thuật toán Apriori để tìm kiếm các tập hợp mục có giá trị trung bình cao. Các bước của phương pháp bao gồm xác định ngưỡng giá trị trung bình, tạo ra các ứng viên thông qua kết hợp và cắt tỉa và kiểm tra các ứng viên để xác định tập hợp mục cuối cùng có giá trị trung bình cao.

Các ưu điểm của phương pháp: Dễ hiểu và triển khai, xử lý dữ liệu lớn, phát hiện tập hợp mục quan trọng.

Các nhược điểm của phương pháp: Độ phức tạp tính toán, vấn đề tỷ lệ tăng, không xử lý tốt với dữ liệu thưa: Apriori không xử lý tốt với dữ liệu thưa, trong đó hầu hết các giao dịch chỉ chứa một số ít mục. Khi dữ liệu trở nên thưa, Apriori có thể tạo ra nhiều tập hợp mục con không quan trọng hoặc không có giá trị.

Thuật toán tiêu biểu theo hướng tiếp cận này là TPAU [9]. Trong thuật toán này, tác giả sử dụng chiến lược tỉa dựa trên độ đo auub và số pha trong thuật toán này sử dụng là 2 pha. Thuật toán này tốn nhiều thời gian chạy vì phải quét cơ sở dữ liệu nhiều lần.

2.2.2. Hướng tiếp cận duyệt theo chiều sâu, cấu trúc chỉ mục mới

Phương pháp khai thác tập hữu ích trung bình cao theo hướng "Depth first, new indexing structure" kết hợp cả ưu điểm của việc duyệt theo chiều sâu (depth-first) và sử dụng một cấu trúc chỉ mục mới để tăng hiệu suất của quá trình khai thác.

Duyệt theo chiều sâu (Depth-first): Phương pháp này khai thác các tập hợp mục có giá trị trung bình cao bằng cách duyệt qua các đường đi từ gốc đến lá trên cấu trúc dữ liệu. Điều này đảm bảo các tập hợp mục có giá trị cao nhất được khám phá sớm hơn, giúp tối ưu hóa quá trình khai thác.

Cấu trúc chỉ mục mới (New indexing structure): Phương pháp này sử dụng một cấu trúc chỉ mục mới để lưu trữ thông tin về tập hợp mục và giá trị trung bình của chúng. Cấu trúc này được thiết kế để tăng tốc độ truy vấn và tìm kiếm các tập hợp mục có giá trị trung bình cao. Nó có thể bao gồm các cấu trúc dữ liệu như cây B, cây Trie hoặc các biểu đồ được tối ưu hóa cho việc tìm kiếm nhanh chóng.

Quá trình khai thác: Quá trình khai thác bắt đầu từ nút gốc của cấu trúc chỉ mục. Mỗi lần duyệt qua một nút, phương pháp sẽ kiểm tra giá trị trung bình của tập hợp mục tại nút đó. Nếu giá trị trung bình vượt qua ngưỡng được xác định, tập hợp mục sẽ được ghi lại hoặc xử lý theo yêu cầu của ứng dụng cụ thể. Sau đó, phương pháp tiếp tục duyệt qua các nút con của nút hiện tại, theo chiều sâu. Quá trình này được lặp lại cho đến khi tất cả các tập hợp mục có giá trị trung bình cao đã được khai thác hoặc không còn nút con nào để duyệt.

Phương pháp "Duyệt theo chiều sâu, cấu trúc chỉ mục mới" có nhiều ưu điểm như tìm kiếm hiệu quả, tiết kiệm tài nguyên, dễ dàng thao tác truy vấn và phù hợp cho dữ liệu có cấu trúc phân cấp. Tuy nhiên, nó cũng có nhược điểm như không phù hợp cho truy xuất ngẫu nhiên, cần thực hiện lại phép duyệt và cập nhật cấu trúc chỉ mục khi có thay đổi dữ liệu. Sự lựa chọn của phương pháp biểu diễn dữ liệu phụ thuộc vào yêu cầu cụ thể của ứng dụng và tính chất của dữ liệu.

Thuật toán tiêu biểu theo hướng tiếp cận này là PBAU [10]. Trong thuật toán này, tác giả sử dụng chiến lược tỉa dựa trên độ đo auub và kết hợp chỉ mục để tỉa các ứng viên, số pha trong thuật toán này sử dụng là 2 pha. Thuật toán làm giảm thời gian xuống vì có sử dụng chỉ mục và cơ sở dữ liệu khác nhau.

2.2.3. Hướng tiếp cận phép chiếu (projection)

Phương pháp khai thác tập hợp mục có giá trị trung bình cao theo hướng "Projection-based" sử dụng kỹ thuật chiếu dữ liệu để tìm kiếm và khai thác các tập hợp mục có giá trị trung bình cao. Chi tiết về phương pháp này như sau:

Chiếu dữ liệu: Phương pháp này bắt đầu bằng việc xác định các thuộc tính quan trọng và xây dựng các chiếu dữ liệu dựa trên các thuộc tính này. Các chiếu dữ liệu là một phiên bản giảm kích thước của dữ liệu gốc, giữ lại các thuộc tính quan trọng và loại bỏ các thuộc tính không quan trọng. Qua quá trình này, phương pháp giảm kích thước dữ liệu, làm cho việc khai thác tập hợp mục trở nên hiệu quả hơn.

Tạo các ứng viên: Sau khi có các chiếu dữ liệu, phương pháp sẽ tạo ra các ứng viên thông qua các phép kết hợp và cắt tỉa dựa trên các chiếu dữ liệu này. Các ứng viên là các tập hợp mục tiềm năng có thể có giá trị trung bình cao và sẽ được kiểm tra tiếp để xác định tập hợp mục cuối cùng có giá trị trung bình cao.

Kiểm tra tập hợp mục: Các ứng viên được kiểm tra để xác định tập hợp mục có giá trị trung bình cao. Quá trình này bao gồm tính toán giá trị trung bình của từng ứng viên trong dữ liệu gốc và so sánh với ngưỡng giá trị trung bình đã được xác định trước. Các tập hợp mục có giá trị trung bình cao sẽ được ghi lại hoặc xử lý theo yêu cầu của quá trình khai thác.

Phương pháp "Projection-based" có một số ưu điểm so với các phương pháp khác trong việc khai thác tập hợp mục: giảm chiều dữ liệu, tập trung vào tập hợp mục quan trọng, hiệu quả trong việc khai thác tập hợp mục có giá trị trung bình cao, phù hợp cho dữ liệu lớn.

Phương pháp khai thác tập hợp mục theo hướng "Projection-based" có thể có một số nhược điểm sau: mất thông tin quan trọng, độ chính xác giảm, độ phức tạp tính toán, độ phức tạp của cấu trúc dữ liệu, phụ thuộc vào việc chọn thuộc tính quan trọng.

Thuật toán tiêu biểu theo hướng tiếp cận này là PAI [11]. Trong thuật toán này, tác giả sử dụng chiến lược tỉa dựa trên độ đo auub cải tiến để tỉa các ứng viên, số pha trong thuật toán này sử dụng là 2 pha. Vì phải chiếu cơ sở dữ liệu, nên thuật toán này sử dụng bộ nhớ nhiều.

2.2.4. Hướng tiếp cận cây (tree)

Phương pháp "Tree-based" trong việc khai thác tập hợp mục là một phương pháp dựa trên cấu trúc cây để tìm kiếm và khai thác các tập hợp mục có giá trị trung bình cao.

Xây dựng cây: Phương pháp "Tree-based" bắt đầu bằng việc xây dựng một cây dựa trên tập dữ liệu ban đầu. Cây có thể là cây quyết định (decision tree), cây quan hệ (association tree), hoặc các loại cây khác tùy thuộc vào bài toán cụ thể. Quá trình xây dựng cây nhằm tìm ra các quy tắc, mẫu hoặc mối quan hệ giữa các thuộc tính và tập hợp mục.

Tìm kiếm tập hợp mục: Sau khi xây dựng cây, phương pháp sử dụng cây để tìm kiếm các tập hợp mục có giá trị trung bình cao. Quá trình này thường bao gồm việc duyệt cây từ gốc đến lá và áp dụng các quy tắc hoặc mẫu đã học từ quá trình xây dựng cây để tìm kiếm các tập hợp mục phù hợp.

Kiểm tra tập hợp mục: Các tập hợp mục được tìm thấy thông qua cây được kiểm tra để xác định liệu chúng có giá trị trung bình cao hay không. Quá trình này thường bao gồm tính toán giá trị trung bình của từng tập hợp mục trong dữ liệu gốc và so sánh với ngưỡng giá trị trung bình đã được xác định trước. Các tập hợp mục có giá trị trung bình cao sẽ được ghi lại hoặc xử lý theo yêu cầu của quá trình khai thác.

Tiếp tục tối ưu hóa cây: Sau khi tìm thấy các tập hợp mục có giá trị trung bình cao, phương pháp "Tree-based" có thể tiếp tục tối ưu hóa cây dựa trên thông tin từ các tập hợp mục này. Quá trình này có thể bao gồm việc cắt tỉa cây, chỉnh sửa quy tắc hoặc mẫu để cải thiện hiệu suất khai thác và đơn giản hóa cây.

Phương pháp "Tree-based" tận dụng cấu trúc

cây để tìm kiếm và khai thác các tập hợp mục có giá trị trung bình cao. Bằng cách xây dựng cây dựa trên dữ liệu và tìm kiếm thông qua cây này, phương pháp này tập trung vào việc tìm kiếm các tập hợp mục phù hợp và giúp giảm độ phức tạp của quá trình khai thác.

Phương pháp "Tree-based" có một số nhược điểm so với các phương pháp khác, đó là: độ phức tạp tính toán, khả năng xử lý dữ liệu thiếu, quá khớp (overfitting), khả năng xử lý dữ liệu không phân loại.

Các thuật toán tiêu biểu theo hướng tiếp cận này là HAUI-Tree [12], IMHAUI [13], MPM [14], IHAUPM [15]. Trong các thuật toán này, số pha sử dụng là 2 pha, tác giả sử dụng chiến lược tỉa dựa trên độ đo auub để tỉa các ứng viên, bộ nhớ tốn khá nhiều vì phải thực thi cấu trúc dữ liệu cây.

2.2.5. Hướng tiếp cận danh sách (list)

Phương pháp "List-based" trong việc khai thác tập hợp mục là một phương pháp dựa trên việc xử lý danh sách các mục để tìm kiếm và khai thác các tập hợp mục có giá trị trung bình cao. Chi tiết về phương pháp này như sau:

Xây dựng danh sách: Phương pháp "List-based" bắt đầu bằng việc xây dựng một danh sách các mục từ tập dữ liệu ban đầu. Danh sách này có thể bao gồm tất cả các mục trong dữ liệu hoặc chỉ bao gồm một phần mục quan trọng. Quá trình xây dựng danh sách nhằm tạo ra một tập các mục để tìm kiếm và khai thác.

Tìm kiếm tập hợp mục: Sau khi xây dựng danh sách, phương pháp "List-based", sử dụng danh sách này để tìm kiếm các tập hợp mục có giá trị trung bình cao. Quá trình này thường bao gồm duyệt qua danh sách và áp dụng các quy tắc hoặc thuật toán để tìm kiếm các tập hợp mục phù hợp.

Kiểm tra tập hợp mục: Các tập hợp mục được tìm thấy thông qua danh sách được kiểm tra để xác định liệu chúng có giá trị trung bình cao hay không. Quá trình này thường bao gồm tính toán giá trị trung bình của từng tập hợp mục trong dữ liệu gốc và so sánh với ngưỡng giá trị trung bình đã được xác định trước. Các tập hợp mục có giá trị trung bình cao sẽ được ghi lại hoặc xử lý theo yêu cầu của quá trình khai thác.

Cập nhật danh sách: Sau khi tìm thấy các tập hợp mục có giá trị trung bình cao, phương pháp "List-based" có thể cập nhật danh sách bằng cách thêm hoặc loại bỏ mục dựa trên kết quả khai thác. Quá trình này giúp cải thiện hiệu suất khai thác và tập trung vào việc tìm kiếm các tập hợp mục tiềm năng.

Ưu điểm: Đơn giản và dễ hiểu, Tính khả dụng cao, Dễ dàng cập nhật danh sách, Hiệu quả cho dữ liệu nhỏ và đơn giản.

Phương pháp "List-based" cũng có một số nhược điểm sau: Độ phức tạp tính toán, Hiệu suất khai thác, Giới hạn ứng dụng, Xử lý dữ liệu lớn.

Các thuật toán tiêu biểu theo hướng tiếp cận này là HAUI-Miner [16] sử dụng cơ chế quét 1 pha, cắt tỉa dựa trên auub, rất tốn chi phí để kết hợp danh sách ứng viên, EHAUPM [17] sử dụng cơ chế quét 1 pha, cắt tỉa dựa trên auub, lub và rtub, cải thiện chiến lược tỉa, FHAUM [18] sử dụng cơ chế quét 1 pha, cắt tỉa dựa trên auub, lubau và tubau, cải thiện chiến lược tỉa. Các thuật toán MHAI [19], FUP-HAUIMD [20], FUP-HAUIMI [21], MEMU+ [22] đều sử dụng cơ chế quét 1 pha, cắt tỉa dựa trên auub. Các thuật toán TUB-HAUPM [23], TKAU [24], VMHAUI [25] sử dụng cơ chế quét 1 pha, cắt tỉa dựa trên auub kết hợp với các độ đo khác.

2.2.6. Hướng tiếp cận Level wise

Phương pháp "Level-wise" tận dụng việc khai thác tập hợp mục theo từng cấp độ, giúp tìm kiếm và khai thác các tập hợp mục có giá trị trung bình cao một cách có hệ thống. Bằng cách chia quá trình khai thác thành các level, phương pháp này giúp giảm độ phức tạp và tập trung vào việc tìm kiếm các tập hợp mục tiềm năng.

Phương pháp "Level-wise" có một số ưu điểm so với các phương pháp khác: Tập trung vào các tập hợp mục có giá trị cao, giảm độ phức tạp tính toán,

dễ dàng mở rộng và tùy chỉnh, tối ưu hóa việc sử dụng tài nguyên.

Phương pháp "Level-wise" cũng có một số nhược điểm và hạn chế cần lưu ý: Đòi hỏi bộ nhớ lớn, độ phức tạp tính toán, khả năng khai thác tập hợp mục phức tạp, phụ thuộc vào việc xác định các level.

Thuật toán tiêu biểu theo hướng tiếp cận này là HAUI-MMAU [26]. Trong thuật toán này, tác giả sử dụng chiến lược tỉa dựa trên độ đo auub cải tiến để tỉa các ứng viên, số pha trong thuật toán này sử dụng là 2 pha. Thuật toán này sinh ra số ứng viên lớn.

2.2.7. Hướng tiếp cận cơ sở dữ liệu dọc

Phương pháp "Vertical database representation based" là một phương pháp trong việc biểu diễn cơ sở dữ liệu theo dạng dữ liệu dọc (vertical).

Biểu diễn cơ sở dữ liệu theo dạng dữ liệu dọc: Thay vì sử dụng biểu diễn truyền thống bằng hàng và cột (dữ liệu ngang), phương pháp này biểu diễn cơ sở dữ liệu dọc theo từng thuộc tính (cột) riêng biệt. Mỗi cột chứa các giá trị tương ứng của thuộc tính trong toàn bộ cơ sở dữ liệu.

Phân tách dữ liệu theo thuộc tính: Dữ liệu trong mỗi thuộc tính được phân tách và lưu trữ thành các cấu trúc dữ liệu tương ứng. Ví dụ, một cấu trúc dữ liệu có thể được sử dụng để lưu trữ các giá trị duy nhất của thuộc tính, một cấu trúc dữ liệu khác có thể được sử dụng để lưu trữ các giá trị tần suất của thuộc tính,...

Tối ưu việc truy vấn dữ liệu: Phương pháp này giúp tối ưu việc truy vấn dữ liệu bằng cách giảm thiểu việc truy cập đến các cấu trúc dữ liệu tương ứng với thuộc tính cần truy vấn. Thay vì phải quét qua toàn bộ cơ sở dữ liệu, chỉ cần truy cập đến cấu trúc dữ liệu tương ứng và thu thập thông tin cần thiết.

Giảm bớt dư thừa dữ liệu: Phương pháp "Vertical database representation based" giúp giảm bớt dư thừa dữ liệu bằng cách lưu trữ các giá trị duy nhất của thuộc tính một lần và sử dụng lại cho các bản ghi khác. Điều này giúp giảm kích thước lưu trữ và tối ưu hóa việc sử dụng tài nguyên.

Đơn giản hóa việc thêm/xóa thuộc tính: Với phương pháp này, việc thêm/xóa thuộc tính trong cơ sở dữ liệu trở nên đơn giản hơn. Thay vì phải điều chỉnh cấu trúc dữ liệu ngang, chỉ cần thêm/xóa một cấu trúc dữ liệu mới cho thuộc tính tương ứng.

Tổng quan, phương pháp "Vertical database representation based" có thể mang lại lợi ích trong việc tối ưu hóa truy vấn và giảm dư thừa dữ liệu, nhưng cũng có nhược điểm liên quan đến chi phí lưu trữ và độ phức tạp của một số truy vấn. Việc lựa chọn phương pháp biểu diễn cơ sở dữ liệu phụ thuộc vào yêu cầu cụ thể của ứng dụng và các yếu tố khác như kích thước dữ liệu, công cụ và quy trình xử lý dữ liệu.

Thuật toán tiêu biểu theo hướng tiếp cận này là DHAUIM [27]. Trong thuật toán này, tác giả sử dụng chiến lược tỉa dựa trên độ đo aub, iaub, laub, aub1 cải tiến để tỉa các ứng viên, số pha trong thuật toán này sử dụng là 1 pha. Chiến lược cắt tỉa còn chưa tốt.

2.2.8. Hướng tiếp cận bản đồ băm (hash map)

Phương pháp "hash map based" là một phương pháp biểu diễn dữ liệu dựa trên cấu trúc bản đồ băm (hash map). Đây là một cấu trúc dữ liệu kết hợp giữa một hàm băm và một bảng băm để lưu trữ và truy xuất dữ liệu.

Ưu điểm của phương pháp "hash map based" bao gồm: Tìm kiếm nhanh, độ mở rộng linh hoạt, dễ dàng thêm, xóa và cập nhật dữ liệu, xử lý xung đột, phù hợp cho việc truy xuất dữ liệu ngẫu nhiên.

Tuy nhiên, phương pháp "hash map based" cũng có một số hạn chế, bao gồm: Không duy trì thứ tự, tốn kém về tài nguyên, hiệu suất giảm khi bảng băm quá đầy.

Thuật toán tiêu biểu theo hướng tiếp cận này là PAI [28]. Trong thuật toán này, tác giả sử dụng

chiến lược tỉa dựa trên độ đo ubriu cải tiến để tỉa các ứng viên, số pha trong thuật toán này sử dụng là 1 pha. Vì phải chiếu cơ sở dữ liệu nên thuật toán này sử dụng bộ nhớ nhiều.

3. Kết luận và hướng đề xuất

Khai thác HAUI mất nhiều thời gian và sử dụng nhiều CPU và bộ nhớ. Tìm các mẫu có lợi ích trung bình cao từ một tập dữ liệu lớn là một thách thức.Bài báo đã đưa ra cái nhìn toàn cảnh về các cách tiếp cận khác nhau trong việc giải quyết bài toán khai thác tập hữu ích trung bình cao. Mỗi thuật toán đưa ra đều có mô tả chiến lược, ưu nhược điểm, cùng với các đặc điểm nổi bật của thuật toán.

Một số đề xuất sau đây trong các hướng phát triển tiếp theo của bài toán khai thác tập hữu ích trung bình cao:

- Mạng nơ-ron tái cấu trúc (Recurrent Neural Networks - RNN): RNN là một loại mạng nơ-ron đặc biệt được thiết kế để xử lý dữ liệu chuỗi hoặc dữ liệu có mối liên kết thời gian.

- Mạng nơ-ron hồi quy dài (Long Short-Term Memory - LSTM): LSTM là một biến thể của RNN, được thiết kế để giải quyết vấn đề của việc mất thông tin lâu dài khi sử dụng RNN. LSTM sử dụng các "cổng" để lựa chọn thông tin quan trọng và loại bỏ thông tin không cần thiết, giúp mạng nơ-ron có khả năng học và ghi nhớ thông tin dài hạn.

- Mô hình sinh (Generative Models): Có nhiều mô hình sinh mới được phát triển như Generative Adversarial Networks (GANs) và Variational Autoencoders (VAEs). Các mô hình sinh này cho phép tạo ra dữ liệu mới từ dữ liệu huấn luyện, có ứng dụng trong việc tạo ra hình ảnh, âm thanh, văn bản và nhiều lĩnh vực khác.

- Học tăng cường (Reinforcement Learning): Học tăng cường là một lĩnh vực trong học máy, nơi một hệ thống tự động học cách đưa ra quyết định và hành động dựa trên tương tác với một môi trường. Các thuật toán học tăng cường đã đạt được thành công đáng kể trong các lĩnh vực như chơi game, điều khiển robot, quản lý tài chính và cũng có thể áp dụng vào bài toán khai thác tập hữu ích trung bình cao ■

TÀI LIỆU THAM KHẢO:

1. R. Agrawal, T. Imielinski and A. Swami (1993). Mining Association Rules between Sets of Items in Large Databases. ACM SIGMOD Record, 207-216.

2. M. S. Chen, J. Han and P. S. Yu (1996). Data mining: an overview from a database perspective. IEEE transactions on knowledge and data engineering, 8(6), 866-883.

3. C. W. Lin, T. P. Hong and W. H. Lu (2009). The Pre-FUFP algorithm for incremental mining. Expert Systems with Applications, 36, 9498-9505.

4. J. Han, J. Pei, Y. Yin and R. Mao (2004). Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. Data Mining and Knowledge Discovery, 8,53-87.

5. M. Liu and J. Qu (2012). Mining High Utility Itemsets without Candidate Generation. ACM International Conference on Information and Knowledge Management, 55-64.

6. G. C. Lan, T. P. Hong and V. S. Tseng (2013). An efficient projection-based indexing approach for mining high utility itemsets. Knowledge and Information Systems, 8, 85-107.

7. H. Yao, H. J. Hamilton and C. J. Butz (2004). A Foundational Approach to Mining Itemset Utilities from Databases. SIAM International Conference on Data Mining, 215-221.

8. U. Yun and. K. Donggyu (2017). Mining of high average-utility itemsets using novel list structure and pruning strategy. Future Generation Computer Systems, 68, 346-360.

9. Hong T. P. et al (2011). Effective utility mining with the measure of average utility, Expert Systems with Applications, 38(7), 8259-8265.

10. Lan G. C. et al (2012). A projection-based approach for discovering high average-utility itemsets. Journal of Information Science and Engineering, 28(1), 193-209.

11. Lan G. C. et al (2012). Efficiently mining high average-utility itemsets with an improved upperbound strategy. International Journal of Information Technology and Decision Making, 11(5), 1009-1030.

12. Lu T. et al (2015). A new method for mining high average utility itemsets. In: IFIP international conference on computer information systems and industrial management. Springer, 33-42.

13. Donggyu Kim & Unil Yun (2017). Efficient algorithm for mining high average-utility itemsets in incremental transaction databases SpringerLink. Available at: https://link.springer.com/article/10.1007/s10489-016-0890-z

14. Yun U. et al (2018). Damped window based high average utility pattern mining over data streams. Knowledge-Based Systems, 144, 188–205.

15. Lin J. C.W. et al (2018). Efficiently updating the discovered high average-utility itemsets with transaction insertion. Engineering Applications of Artificial Intelligence, 72, 136-149.

16. Lin J. C. W. et al (2016). An efficient algorithm to mine high average-utility itemsets. Advanced Engineering Informatics, 30(2), 233-243.

17. Lin J. C. W. et al (2017). EHAUPM: efficient high average-utility pattern mining with tighter upper bounds. IEEE Access, 5, 12927-12940.

18. Jerry Chun-Wei Lin et al (2017). A fast algorithm for mining high average-utility itemsets. Applied Intelligence 47(12).

19. Yun U., Kim D. (2017). Mining of high average-utility itemsets using novel list structure and pruning strategy. Future Gener Computer System, 68, 346-360.

20. Jerry Chun-Wei Lin et al (2019). Maintenance algorithm for high average-utility itemsets with transaction deletion. Applied Intelligence, 48(10).

21. Zhang B et al (2018). Maintenance of discovered high average-utility itemsets in dynamic databases. Applied Sciences, 8(5), 769.

22. Jerry Chun-Wei Lin et al (2017). MEMU: More Efficient Algorithm to Mine High Average-Utility patterns with multiple minimum average-utility thresholds. IEEE Access, 14(8).

23. Jimmy Ming-Tai Wu et al (2018). TUB-HAUPM: tighter upper bound for mining high average-utility patterns. IEEE Access, 99,1-1.

24. Ronghui Wu et al (2018). Top-k high average-utility itemsets mining with effective pruning strategies Applied Intelligence, 48(10).

25. Truong T et al (2019). Efficient high average-utility itemset mining using novel vertical weak upper-bounds. Knowledge-based systems, 183, 104847.

26. Lin JC-W et al (2016). Efficient mining of high-utility itemsets using multiple minimum utility thresholds Knowledge-based systems, 113, 100-115.

27. Tin Truong et al (2018). Efficient vertical mining of high average-utility itemsets based on novel upper-bounds. IEEE Transactions on Knowledge and Data Engineering, 1041-4347.

28. M. S. Bhuvaneswari et al (2022). An efficient hash map based technique for mining high average utility itemset. Indian Academy of Sciences.

Ngày nhận bài: 8/12/2023
Ngày phản biện đánh giá và sửa chữa: 25/12/2023
Ngày chấp nhận đăng bài: 8/1/2024

Thông tin tác giả:
PHẠM TUẤN KHIÊM
Khoa Công nghệ Thông tin
Trường Đại học Công Thương Thành phố Hồ Chí Minh
Email: khiempt@huit.edu.vn

A SURVEY OF THE HAUIM

● **PHAM TUAN KHIEM**
Faculty of Information Technology,
HCMC University of Industry and Trade

ABSTRACT:

The efficient extraction of valuable patterns from large-scale datasets has become increasingly essential in various data mining applications. High Average-Utility Itemset Mining (HAUIM) is a crucial data mining task that involves the identification of itemsets with high utility values. This study comprehensively explored the current state of research in HAUIM, including its algorithms and features. The study discussed the significant advancements, challenges, and potential future directions in this field, providing valuable insights into the growing landscape of high-utility itemset mining.

Keywords: high average-utility itemset mining, utility itemset, data mining, utility, high-average utility itemset.