رویکرد مبتنی بر القای مجدد برای کاوش کارآمد مجموعه اقلام با سودمندی بالا از مجموعه داده‌های افزایشی

هادی ناصری
عضو هیئت علمی بخش مهندسی کامپیوتر، دانشگاه آزاد اسلامی واحد استهبان، استهبان، ایران
فاطمه زارع
دانشجوی کارشناسی ارشد دانشگاه آزاد اسلامی واحد استهبان، استهبان، ایران

چکیده
کاوش مجموعه اقلام با سودمندی بالا، یک حوزه تحقیقاتی مهم است که بر شناسایی ترکیباتی از مجموعه اقلام در پایگاه‌های داده تمرکز دارد که دارای مقدار سودمندی بالاتر از یک آستانه مشخص شده توسط کاربر هستند. با این حال، اغلب الگوریتم‌های موجود در این حوزه، فرض می‌کنند که پایگاه‌های داده ایستا و بدون تغییر هستند. این فرض در دنیای واقعی چندان واقع‌بینانه نیست، زیرا مجموعه داده‌های دنیای واقعی به طور مداوم با داده‌های جدید در حال رشد و تغییر هستند. علاوه بر این، الگوریتم‌های موجود تنها بر مقدار سودمندی برای شناسایی مجموعه اقلام مرتبط تکیه می‌کنند. این موضوع منجر به این می‌شود که حتی ترکیباتی که در مراحل اولیه رخ داده‌اند نیز به عنوان خروجی تولید شوند. اگرچه برخی از الگوریتم‌های استخراج یک رویکرد مبتنی بر پشتیبانی را برای محاسبه فراوانی مجموعه اقلام اتخاذ می‌کنند، اما ماهیت زمانی مجموعه‌های آیتم‌ها را در نظر نمی‌گیرند.برای پرداختن به این چالش‌ها، این مقاله الگوریتم Scented Utility Miner (SUM) را پیشنهاد می‌کند که از یک استراتژی القایی مجدد برای ردیابی جدید بودن وقوع مجموعه آیتم‌ها و کاوش مجموعه‌های اقلام از پایگاه‌های داده افزایشی استفاده می‌کند. این مقاله رویکردی نوین برای کاوش مجموعه اقلام با سودمندی بالا از پایگاه‌های داده پویا ارائه می‌دهد و با ارائه چندین آزمایش، کارایی روش پیشنهادی را اثبات می‌کند.

واژگان کلیدی: کاوش مجموعه اقلام با سودمندی بالا، داده کاوی،کاوش کارایی حافظه، مجموعه داده های افزایشی، data mining

مقدمه

مقالات باید در پیشرفت در کشف دانش و داده کاوی منجر به توسعه تکنیک های کارآمد برای استخراج اطلاعات مفید از پایگاه های داده در قالب الگوها شده است. دگرگونی عظیم کلان داده با بهبود فرآیند جمع آوری و بازیابی داده از پایگاه های داده، منجر به کاربرد داده کاوی در زمینه های مختلفی همچون داده کاوی وب، تحلیل سوابق سلامت و بررسی خطاهای دستگاه های هوشمند شده است. کاوش پایگاه‌های داده تراکنش‌ها برای یافتن الگوهای مفید، یکی از حوزه‌های پرکاربرد و مورد توجه در داده‌کاوی است [۲۱]. در ابتدا، مجموعه اقلام پرتکرار از پایگاه‌های داده تراکنش‌ها استخراج می‌شد، اما این رویکرد پارامترهای مرتبط‌تری مانند هزینه و تعداد اقلام را در نظر نمی‌گرفت. داده‌کاوی مجموعه اقلام با سودمندی بالا (HUIM) با استخراج الگوهایی که مقادیر سودمندی آن‌ها بالاتر از یک آستانه مشخص‌شده توسط کاربر است، بر این محدودیت غلبه می‌کند. با این حال، کاوش مجموعه اقلام با سودمندی بالا پیچیده‌تر از مجموعه اقلام پرتکرار است زیرا زیرمجموعه‌های مجموعه اقلام با سودمندی بالا لزوماً مجموعه اقلام با سودمندی بالا نیستند. برای مقابله با این مشکل، مکانیسم‌های هرس کارآمد مورد نیاز است و ویژگی بسته شدن نزولی وزن‌دار تراکنش و سایر تکنیک‌های ذخیره‌سازی و بازیابی در این مقاله پیشنهاد شده‌اند. با این حال، اکثر این الگوریتم‌ها برای مجموعه داده‌های ایستا طراحی شده‌اند و فضای جستجوی بزرگی ایجاد می‌کنند که نیازمند چندین بار اسکن پایگاه داده است. از آنجایی که مجموعه داده‌های دنیای واقعی پیوسته در حال رشد هستند، الگوریتم‌های داده‌کاوی مجموعه اقلام با سودمندی بالا باید ماهیت افزایشی پایگاه‌های داده را در نظر بگیرند.

کاوش اقلام های پرتکرار از یک پایگاه داده در حال گسترش، چالش‌های جدیدی را به همراه دارد. یکی از این چالش‌ها، لزوم کاوش اقلام های پرتکرار با آخرین روندهای موجود در داده‌ها است، زیرا تراکنش‌های جدید به طور مداوم به پایگاه داده اضافه می‌شوند. الگوهای منسوخ می‌توانند گمراه‌کننده باشند و باید از نتایج حذف شوند. علاوه بر این، با توجه به اینکه پایگاه داده ورودی به طور پیوسته در حال تحول است، آستانه حداقل مطلوبیت مورد نیاز برای کاوش ممکن است متغیر باشد و بنابراین باید به صورت پویا و بر اساس مجموعه فعلی رکوردهای موجود محاسبه شود. برای پرداختن به این چالش ها، این مقاله یک الگوریتم جدید به نام Scented Utility Miner (SUM)پیشنهاد می کند. الگوریتم SUM (Scented Utility Miner) بر پایه یک ساختار داده به نام نگاشت باقی مانده که در مرجع [۱۸] پیشنهاد شده است، بنا شده و از یک نگاشت اصلی برای نگهداری اطلاعات ضروری جهت استخراج مجموعه اقلام با مطلوبیت بالا استفاده می‌کند. این الگوریتم از یک استراتژی باز القایی (reinduction) برای ردیابی تازگی وقوع مجموعه اقلام و محاسبه پویای مقدار آستانه حداقل مطلوبیت استفاده می‌کند.

کارهای مرتبط

در مقالات علمی موجود، الگوریتم‌های متعددی برای کاوش مجموعه اقلام با سودمندی بالا از پایگاه‌های داده بزرگ ارائه شده است. راهکارهای ابتدایی برای مسئله کاوش مجموعه اقلام با سودمندی بالا بر اساس دو معیار اصلی طبقه‌بندی می‌شوند: تکنیک‌های مورد استفاده برای کاوش مجموعه اقلام با سودمندی بالا و ساختارهای داده مورد استفاده برای ذخیره‌سازی مجموعه آیتم‌های حاصل. این رویکردها را می‌توان به سه نوع اصلی طبقه‌بندی کرد: تکنیک‌های مبتنی بر تولید و آزمایش، تکنیک‌های الگوی رشد مبتنی بر درخت، و تکنیک‌های مبتنی بر فهرست.

کار پیشنهادی

در این پژوهش، الگوریتم جدیدی با نام Scented Utility Miner (SUM) برای داده‌کاوی مجموعه اقلام با سودمندی بالا معرفی شده است. این الگوریتم از مفهوم residue map برای ذخیره‌سازی اطلاعات مرتبط با سودمندی هر مجموعه اقلام استفاده می‌کند. همانطور که در شکل ۱ نشان داده شده است، residue mapsشامل چهار فیلد می‌باشد: مجموعه اقلام(itemset) ، فهرست شناسه تراکنش(TID-list) ، سودمندی کل (total utility) و مانده.(residue)

مجموعه‌ای مرتب‌شده از residue maps ، master mapرا تشکیل می‌دهد master map .به صورت بازگشتی پیمایش می‌شود تا مجموعه اقلام با سودمندی بالا کاوش گردد. مزیت اصلی استفاده از residue maps در داده‌کاوی مجموعه اقلام با سودمندی بالا، انعطاف‌پذیری بالای آنهاست. این انعطاف‌پذیری به این دلیل است که residue maps با ذخیره‌سازی مستقیم اطلاعات آستانه حداقل برای محاسبه مقدار مانده، مجموعه اقلام با سودمندی بالا را کاوش می‌کنند. با درج تراکنش‌های جدید و تغییر آستانه سودمندی حداقل، تنها residue maps تولید می‌شوند که با توجه به مجموعه رکوردهای فعلی مرتبط هستند.

الگوریتم SUM به منظور برآورده کردن نیازهای به‌روزرسانی‌های تدریجی در مجموعه داده اولیه طراحی شده است. برای هر به‌روزرسانی در مجموعه داده، master mapبه‌روز می‌شود و از مفهوم باز القای اقلام و تنظیم پویای آستانه برای فرآیند داده‌کاوی استفاده می‌شود. الگوریتم رسمی طرح پیشنهادی در الگوریتم ۱ ارائه شده است.

الگوریتم ۱ به عنوان ورودی، یک پایگاه داده D با k تراکنش و n آیتم متمایز را دریافت می‌کند، که هر آیتم دارای یک مقدار سودمندی است. همچنین، آستانه سودمندی حداقل و حداکثر مقدار شمارنده بازالقاء به عنوان ورودی‌های دیگر دریافت می‌شوند. خروجی الگوریتم، مجموعه آیتم‌های با سودمندی بالا است.

الگوریتم با به‌روزرسانی شمارنده بازالقاء تمامی آیتم‌های موجود در master map شروع به کار می‌کند. در ابتدا، شمارنده بازالقاء هر آیتم در master map به میزان یک واحد کاهش می‌یابد (خط ۲). هنگامی که یک تراکنش معین اسکن می شود، شمارنده بازالقاء اقلام موجود در آن تراکنش روی حداکثر مقدار شمارش مجدد (خط:۱۳، خط:۲۱) تنظیم می شود. سپس، برای هر تراکنش در پایگاه داده، الگوریتم برای هر آیتم در آن تراکنش اجرا می شود. اگر آیتمی در master map نباشد (خط:۶)، الگوریتم یک residue maps جدید برای آن آیتم ایجاد می کند و آن را به master map اضافه می کند (خط:۷-۱۲). اگر مورد از قبل در master map باشد (خط: ۱۵)، الگوریتم residue maps آن را بازیابی می کند و فیلدهایTID ، سودمندی کل، و قسمت های باقیمانده را به روز می کند (خط: ۱۵-۲۰).

سپس الگوریتم بررسی می‌کند که آیا TWU (Total Weighted Utility یا سودمندی وزنی کل (آیتم بزرگتر از آستانه سودمندی حداقل است یا خیر (خط ۲۳). در صورت مثبت بودن پاسخ، الگوریتم یک مجموعه مرتب‌شده از residue maps که بزرگتر یا مساوی با سودمندی آن هستند را تولید می‌کند (خط ۲۴) و به صورت بازگشتی master map را پیمایش می‌کند تا ترکیب‌های مختلفی از مجموعه آیتم‌ها را ایجاد کرده و مجموعه آیتم‌های با سودمندی بالا را تولید کند (خط ۲۹).

الگوریتم پیشنهادی به دلیل اینکه تنها residue maps مرتبط را با درج تراکنش‌های جدید و تغییر آستانه سودمندی حداقل تولید می‌کند، کارآمد و مقیاس‌پذیر است. استفاده از residue maps ، سهولت تطبیق‌پذیری را فراهم می‌کند و مفهوم بازالقاء آیتم‌ها، کارایی فرآیند داده‌کاوی را بهبود می‌بخشد.

Algorithm 1 SUM Main Procedure

INPUT
D: database
$MinU$: minimum utility threshold
$MaxR$: maximum value of reinduction counter
OUTPUT
HUI-set: set of high utility itemsets

```
1: for each i ∈ M do
2:    Rindcntr(i) ← Rindcntr(i) - 1
3: end for
4: for each T_k ∈ D do
5:    for each i ∈ T_k do
6:       if i∉ M then
7:          TID-list(TL_i) ← (T_k, U_i)
8:          TotUtil(T_i) ← U_i
9:          Residue(R_i) ← MinU - T_i
10:         rMap_i ← (i, TL_i,T_i,R_i)
11:         TWU_i ← TU_i
12:         M ← add(rMap_i)
13:         Rindcntr(i)← MaxR
14:      else
15:         rMap_i ← get ResidueMap(i)
16:         TL_i ← get TID-list(rMap_i)
17:         T_i ← T_i + U_i
18:         R_i ← MinU - T_i
19:         TWU_i ← TWU_i + TU_i
20:         rMap_i ← (i, TL_i,T_i,R_i)
21:         Rindcntr(i) ← MaxR
22:      end if
23:      if TWU_i >MinUtil then
24:         SM ← M ≻ (U_i)
25:      end if
26:   end for
27: end for
28: for all rMap_i ∈ SM do
29:    HUI_i ← (rMap_i, SM, MinU,sizeof(SM))
30:    HUI-set ← HUI-set ∪ HUI_i
31: end for
```

شکل ۱

۱.به‌روزرسانی‌های پایگاه داده افزایشی تراکنش‌ها

الگوریتم SUM با ساختن یک master map اولیه برای یک پایگاه داده مشخص شروع می شود. این نگاشت بر اساس ارزش کل ابزار از مجموعه مربوطه از residue maps طبقه‌بندی می‌شود، همانطور که در الگوریتم ۱ نشان داده شده است. سپس، الگوریتم به‌طور

متوالی هر تراکنش را در پایگاه داده ورودی برای شناسایی مجموعه‌ای از موارد اسکن می‌کند. برای هر آیتم تازه کشف شده، یک residue mapsجدید ساخته می شود. اگر یک مورد قبلاً در پایگاه داده اسکن شده باشد، residue mapsآن به روز می شود.
برای یک پایگاه دادهD ، اگر مجموعه‌ای از تراکنش‌های جدید N درج شوند به طوری که پایگاه داده به‌روز شده 'D باشد، ویژگی‌های زیر برقرار خواهند بود.
ویژگی۱: اگر یک مجموعه اقلام X یک مجموعه اقلام کاربردی بالا در پایگاه داده اصلی D باشد و X در مجموعه نفی N ظاهر نشود، آنگاه X نیز یک مجموعه اقلام با سودمندی بالا در پایگاه داده تبدیل شده 'D برای حداقل آستانه سودمندی معین خواهد بود.
ویژگی ۲: اگر یک مجموعه آیتم X در پایگاه داده اصلی D یک مجموعه اقلام با سودمندی بالا نباشد و X در مجموعه نفی N ظاهر نشود، آنگاه X در پایگاه داده تبدیل شده 'D نیز برای یک آستانه سودمندی حداقل داده شده، یک مجموعه اقلام با سودمندی بالا نخواهد بود.
ویژگی ۳: اگر یک مجموعه آیتم X یک مجموعه اقلام با سودمندی بالا در پایگاه داده اصلی D نباشد، و X در مجموعه نفی N با کاربرد کلی UX ظاهر می شود، آنگاه X یک مجموعه اقلام با سودمندی بالا در پایگاه داده تبدیل شده 'D خواهد بود اگر و فقط اگر UX بزرگتر یا مساوی با حداقل آستانه سودمندی مشخص شده باشد.
برای محاسبه به‌روزرسانی‌های تدریجی پایگاه داده، الگوریتم SUM از master map موجود برای ترکیب اطلاعات جدید از تراکنش‌ها استفاده می‌کند. هنگامی که مجموعه‌ای جدید از تراکنش‌ها به پایگاه داده اضافه می‌شود، الگوریتم بررسی می‌کند که آیا اقلام جدیدی وجود دارند یا خیر. در صورت وجود، یک Residue Map ایجاد شده و اشاره‌گری به آن به Master Map اضافه می‌شود. اگر یک آیتم قبلاً در پایگاه داده وجود داشته باشد، residue mapمربوطه بازیابی و با اطلاعات جدید به روز می شود. سپس Master Map مجدداً مرتب می‌شود تا مجموعه‌های آیتم‌ها بر اساس سودمندی کل به ترتیب صعودی مرتب شوند. در نهایت، Master Map به روز شده به صورت بازگشتی پیمایش می شود تا مجموعه دقیقی از HUI ها برای پایگاه داده به روز شده کاوش شود.

۲.حفظ ارتباط مجموعه اقلام در به‌روزرسانی‌های متعدد پایگاه داده

برای جلوگیری از تأثیر الگوهای قدیمی بر فرآیند کاوش، تعیین یک فاکتور مرتبط بر اساس مجموعه آیتم ها در پایگاه داده بسیار مهم است. بنابراین، برای ارزیابی ارتباط مجموعه اقلام، مفهوم القاء مجدد به شرح زیر معرفی می شود:
۱. یک شمارشگر القایی مجدد ،برای هر آیتم i در پایگاه داده تعریف شده است. اندازه پنجره به عنوان حداکثر تعداد تراکنش‌هایی که یک آیتم باید حداقل یک بار در آنها ظاهر شود تا مرتبط در نظر گرفته شود، تعریف می‌شود.
۲. مقدار اولیه شمارشگر القایی مجدد برای تمامی اقلام صفر تعیین می‌شود.
۳ .همانطور که دنباله ای از تراکنش ها در پایگاه داده اسکن می شود، برای هر مورد جدید x که کشف می شود، شمارشگر القایی مجدد آن به مقدار مربوط به حداکثر اندازه پنجره، تنظیم می شود .
۴.در هر تراکنش بعدی، در صورتی که آیتم در تراکنش رخ ندهد، شمارشگر القایی مجدد آن یک واحد کاهش می‌یابد. به عبارت دیگر،
۵.در هر تراکنش بعدی، در صورتی که آیتم در تراکنش رخ دهد، شمارشگر القایی مجدد آن به مقدار حداکثر اندازه پنجره بازنشانی می‌شود. به عبارت دیگر، به منظور اطمینان از اینکه تنها آیتم‌های مرتبط در نگاشت اصلی (master map) قرار می‌گیرند،
شمارشگرهای القایی مجدد آیتم‌های منفرد محاسبه می‌شوند. تنها آن دسته از آیتم‌هایی که شمارشگرالقایی مجدد غیرصفر دارند، برای درج در نگاشت اصلی در نظر گرفته می‌شوند. محاسبه شمارشگرهای القایی مجدد برای هر آیتم در پایگاه داده تضمین می‌کند که حتی با افزایش حجم پایگاه داده به دلیل افزوده شدن تراکنش‌های جدید، ارتباط و به‌روز بودن آیتم‌ها حفظ می‌شود. اندازه پنجره برای تعریف اینکه یک آیتم چقدر باید جدید باشد تا مرتبط در نظر گرفته شود، استفاده می‌شود. اندازه پنجره را می‌توان متناسب با نیازهای یک برنامه تنظیم کرد. برای برنامه‌هایی که نیاز به حساسیت بالا نسبت به تغییرات در داده‌های ورودی دارند، مانند تحلیل بازار سهام، می‌توان از یک اندازه پنجره کوچک استفاده کرد تا اطمینان حاصل شود که تغییرات در نگاشت اصلی منعکس می‌شوند. با این حال، برای

کاربردهایی با الگوهای تغییر آهسته، مانند فروش محصولات در یک فروشگاه خرده فروشی، می‌توان از یک اندازه پنجره متوسط تا بزرگ استفاده کرد.

۳.انتخاب پویا از حداقل آستانه سودمندی بر اساس به‌روزرسانی‌های افزایشی در پایگاه داده

برای کاوش مجموعه اقلام با سودمندی بالا (HUIs) از یک پایگاه داده تراکنش در حال رشد، مقدار آستانه سودمندی باید به‌روزرسانی شود تا روندهای متغیر را منعکس کند. در اینجا یک استراتژی پیشنهادی برای پاسخگویی به الزامات مقدار آستانه پویا ارائه شده است:

۱.برای اولین اسکن پایگاه داده، مقدار حداقل آستانه سودمندی (MU_0) تعیین شده توسط کاربر تنظیم شده است. پس از استخراج HUI ها، کمترین مقدار سود کل $(TU_{Min(0)})$ برای مجموعه فعلی نگاشت های باقیمانده شناسایی می شود تا آستانه در اسکن بعدی افزایش یابد.

۲.هنگامی که پایگاه داده به‌روزرسانی می‌شود، یک مقدار حداقل سودمندی جدید با اضافه کردن کمترین سودمندی کل از اسکن اولیه به مقدار آستانه سودمندی حداقل مشخص شده توسط کاربر به صورت زیر محاسبه می‌شود:

$$(MU_1) \leftarrow MU_0 + TU_{Min(0)}$$

۳.کمترین مقدار کل سودمندی نیز بر اساس مجموعه رکوردهای جدید برای اسکن های بعدی به روز می شود.

۴.برای هر به‌روزرسانی بعدی پایگاه داده، مقدار آستانه حداقل سودمندی به طور سیستماتیک بر اساس مقدار سودمندی قبلی و کمترین سودمندی کل از اسکن قبلی به صورت زیر افزایش می‌یابد:

$$(MU_{i+1}) \leftarrow MU_i + TU_{Min(i)}$$

TID	Transaction
1	q:4, w:3, t:1, r:2
2	t:2
3	t:3, y:5
4	q:2, t:4, y:5
5	w:6
6	q:1

شکل ۲- پایگاه داده اصلی

Item	TID-list	Total Utility	Residue
q	1:4, 4:2, 6:1	7	4
r	1:2	2	9
t	1:1, 2:2, 3:3, 4:4	10	1
w	1:3, 5:6	11	0
y	3:5, 4:5	10	1

شکل ۳- مجموعه ای از residue map

این استراتژی تضمین می‌کند که مجموعه آیتم‌های با سودمندی بالا (HUIs) بر اساس یک مقدار آستانه سودمندی حداقل محاسبه شده به صورت روشمند استخراج می‌شوند، که این مقدار با روندهای پایه‌ای در پایگاه داده همخوانی نزدیکی دارد. مقدار کمترین سودمندی کل برای یک به‌روزرسانی افزایشی ممکن است ثابت بماند، افزایش یا کاهش یابد، و مقدار آستانه بر این اساس تنظیم می‌شود. استفاده از یک آستانه محاسبه شده پویا برای جلوگیری از تولید تعداد زیادی نتایج نامرتبط که می‌توانند اطلاعات مفید زیربنایی را پنهان کنند، حیاتی است.

نتایج تجربی

الگوریتم SUM در جاوا بر روی رایانه ای با پردازنده چهار هسته ای اینتل i۵ با فرکانس ۱.۴ گیگاهرتز و ۸ گیگابایت حافظه پیاده سازی شده است. در این بخش، عملکرد الگوریتم SUM را ارزیابی کرده و آن را با الگوریتم‌های کاوش پیشرفته در چندین مجموعه داده واقعی مقایسه می‌کنیم. الگوریتم‌ها تا زمان برآورده شدن یکی از شرایط اجرا می‌شوند: مشخص شدن یک برنده قطعی، طولانی شدن زمان اجرای یک الگوریتم یا تمام شدن حافظه سیستم.

- عملکرد مقایسه ای SUM با سایر الگوریتم های کاوش مجموعه اقلام سودمندی بالا:

الگوریتم SUM کوتاه ترین زمان اجرا را در بین سه الگوریتم نشان می دهد. این بهبود قابل توجه در زمان اجرا را می توان به استفاده از residue maps و master map به عنوان ساختار داده زیربنایی برای استخراج نسبت داد. این ساختارهای داده ذخیره سازی و پردازش کارآمدتری از داده ها را ارائه می دهند و در نتیجه سربار محاسباتی را کاهش می دهند و روند اجرا را سرعت می بخشند.

- عملکرد مقایسه ای SUM با الگوریتم های کاوشی افزایشی:

ما عملکرد SUM را با عملکرد EIHI مقایسه کردیم، که در حال حاضر بهترین الگوریتم برای کاوش مجموعه اقلام با سودمندی بالا افزایشی است. عملکرد الگوریتم SUM به خصوص در هنگام مواجهه با مجموعه داده‌های افزایشی، به دلیل مدیریت کارآمد به‌روزرسانی‌های پویا و انطباق‌پذیری با شرایط متغیر پایگاه داده، نسبت به الگوریتم EIHI برتر است.نتایج به وضوح نشان می‌دهند که SUM عملکرد بهتری نسبت به EIHI دارد.

- عملکرد مقایسه ای SUM با الگوریتم های مبتنی بر پنجره لغزان:

الگوریتم‌های naive-FHMDS و FHMDS که از لیست‌های معکوس و مدل پنجره لغزان برای استخراج HUIها از پایگاه‌های داده افزایشی با تنظیم آستانه پویا استفاده می‌کنند، از نظر زمان اجرا و میزان مصرف حافظه،توسط SUM بهتر عمل می‌کنند.

هر دو الگوریتم naive-FHMDS و FHMDS یک پنجره را به عنوان یک دسته از تراکنش‌ها تعریف می‌کنند که برای تشکیل یک ساختار شبیه لیست معکوس استفاده می‌شود. علاوه بر این، استفاده وزنی از اقلام برای هر دسته در حافظه ذخیره می‌شود. با این حال، با افزایش تعداد دسته‌ها در پایگاه داده، این روش به مقدار بیشتری حافظه نیاز دارد. علاوه بر این، الگوریتم FHMDS نیاز به محاسبات برای هر پنجره دارد که منجر به افزایش زمان محاسبات و ناکارآمدی می‌شود.

نتیجه‌گیری

در این مقاله، ما یک رویکرد مبتنی بر القای مجدد را برای کاوش مجموعه‌های اقلام با سودمندی بالا از پایگاه‌های داده افزایشی پیشنهاد کرده‌ایم. ما مفهوم شمارنده‌های القای مجدد را معرفی کردیم و یک استراتژی افزایش آستانه برای بهبود فرآیند کاوش مورد بحث قرار دادیم. الگوریتم پیشنهادی ما از نظر زمان اجرا و حافظه مورد نیاز از الگوریتم های موجود بهتر عمل کرد. علاوه بر این، مقیاس پذیری را با تغییر اندازه پنجره آزمایش کردیم.

الگوریتم SUM پیشنهادی، چندین نقص مرتبط با الگوریتم‌های موجود استخراج مجموعه اقلام با سودمندی بالا برای پایگاه‌های داده افزایشی را برطرف می‌کند. اولاً، الگوریتم‌های سنتی به دلیل ذخیره‌سازی نتایج میانی و ساختارهای داده بزرگ مانندtire ، که با افزایش اندازه پایگاه داده ممکن است غیرقابل اجرا شوند، از نیاز به حافظه بالا رنج می‌برند. الگوریتم SUM با استفاده از residue map بر این مشکل غلبه می‌کند، که در مقایسه با سایر ساختارهای داده به طور قابل توجهی به حافظه کمتری نیاز دارد. ثانیاً، الگوریتم‌های موجود در حفظ سازگاری و دقت در هنگام مواجهه با پایگاه‌های داده پویا با آستانه‌های متغیر با مشکل مواجه می‌شوند. الگوریتم SUM این مشکل را با استفاده از یک استراتژی القای مجدد مبتنی بر مدل پنجره لغزان که مقدار آستانه سودمندی را بر اساس اسکن‌های قبلی تنظیم می‌کند، برطرف می‌کند. در نهایت، الگوریتم های سنتی اغلب به اندازه کافی مقیاس پذیر نیستند تا بتوانند مجموعه داده های بزرگ با آستانه های پویا را مدیریت کنند. الگوریتم SUM با استفاده از residue map به عنوان یک ساختار میانی برای ذخیره اطلاعات ابزار از اسکن‌های قبلی، مقیاس‌پذیری را بهبود می‌بخشد و آن را برای استخراج مجموعه‌های اقلام مفید از پایگاه‌های داده افزایشی کارآمدتر و مؤثرتر می‌کند.

برخی از مسیرهای ممکن برای توسعه آتی الگوریتم SUM شامل گسترش آن به سایر انواع مسائل استخراج با سودمندی بالا است. الگوریتم SUM از یک رویکرد مبتنی بر residue map برای استخراج مجموعه اقلام با سودمندی بالا از پایگاه‌های داده افزایشی استفاده می‌کند. ساختارهای داده و استراتژی‌های مشابهی را می‌توان برای سایر انواع مسائل کاوش با سودمندی بالا، مانند کاوش با وزن‌های منفی یا کاوش زمانی، بررسی کرد. علاوه بر این، الگوریتم SUM می تواند با سایر تکنیک های داده کاوی مانند طبقه بندی یا خوشه بندی ادغام شود تا تجزیه و تحلیل جامع تری از داده ها ارائه دهد. با افزایش اندازه داده ها، موازی سازی فرآیند کاوش برای اطمینان از اجرای کارآمد ضروری می شود. نسخه های موازی الگوریتم SUM را می توان برای بهره برداری از معماری های چند هسته ای یا توزیع شده توسعه داد.

مسئله استخراج مجموعه اقلام با سودمندی بالا، کاربردهای متعددی در حوزه‌هایی مانند خرده‌فروشی، مراقبت‌های بهداشتی و مالی دارد و الگوریتم SUM می‌تواند برای رفع چالش‌ها و الزامات خاص در این حوزه‌ها تطبیق داده شده و به کار گرفته شود.

منابع

L. Kaufman and P. J. Rousseeuw, Finding Groups in Data: An Introduc tion to Cluster Analysis, vol. ٣٤٤. Hoboken, NJ, USA: Wiley, ٢٠٠٩.

J. Friedman, T. Hastie, and R. Tibshirani, The Elements of Statistical Learning, vol. ١, no. ١٠. New York, NY, USA: Springer, ٢٠٠١.

M. W. Berry and M. Browne, Lecture Notes in Data Mining. Singapore: World Scienti c, ٢٠٠٦.

S.H.Kaisler, F. J. Armour, A. Espinosa, and W. H. Money, Big data and analytics challenges and issues, ٢٠١٤.

X. Wu, X. Zhu, G.-Q. Wu, and W. Ding, Data mining with big data, IEEE Trans. Knowl. Data Eng., vol. ٢٦, no. ١, pp. ٩٧١٠٧, Jan. ٢٠١٤.

D. Agrawal, C. Budak, A. El Abbadi, T. Georgiou, and X. Yan, Big data in online social networks: User interaction analysis to model user behavior in social networks, in Databases in Networked Information Systems, A. Madaan, S. Kikuchi, and S. Bhalla, Eds. Cham, Switzerland: Springer, ٢٠١٤, pp. ١١٦, doi: ١٠,١٠٠٧/٩٧٨-٣-٣١٩-٠٥٦٩٣-٧_١.

H.-L.Nguyen,Y.-K.Woon,andW.-K.Ng, Asurveyondatastreamclus tering and classi cation, Knowl. Inf. Syst., vol. ٤٥, no. ٣, pp. ٥٣٥٥٦٩, Dec. ٢٠١٥.

C. C. Aggarwal and C. K. Reddy, Data clustering, in Algorithms and Application. Boca Raton, FL, USA: CRC Press, ٢٠١٤.

A. B. Bondi, Characteristics of scalability and their impact on per formance, in Proc. ٢nd Int. Workshop Softw. Perform. (WOSP), ٢٠٠٠, pp. ١٩٥٢٠٣.

D. C. Anastasiu, J. Iverson, S. Smith, and G. Karypis, Big data fre quent pattern mining, in Frequent Pattern Mining, C. C. Aggarwal and J. Han, Eds. Cham, Switzerland: Springer, ٢٠١٤, pp. ٢٢٥٢٥٩, doi: ١٠,١٠٠٧/٩٧٨-٣-٣١٩-٠٧٨٢١-٢_١٠. K. Jaseena and J. M. David, Issues, challenges, and solutions: Big data mining, in Proc. NeTCoM, CSIT, GRAPH-HOC, SPTM, ٢٠١٤, pp. ١٣١١٤٠.

A. S. Shirkhorshidi, S. Aghabozorgi, T. Y. Wah, and T. Herawan, Big data clustering: A review, in Computational Science and Its Applications ICCSA ٢٠١٤. Cham, Switzerland: Springer, ٢٠١٤, pp. ٧٠٧٧٢٠, doi: ١٠,١٠٠٧/٩٧٨-٣-٣١٩-٠٩١٥٦-٣_٤٩.

D. Jayalatchumy, P. Thambidurai, and A. A. Vasumathi, Parallel pro cessing of big data using power iteration clustering over MapReduce, in Proc. World Congr. Comput. Commun.Technol., Feb. ٢٠١٤, pp. ١٧٦١٧٨.

W. Kim, Parallel clustering algorithms: Survey, Parallel Algorithms, Spring, vol. ٣٤, p. ٤٣, ٢٠٠٩.

R. Xu and D. Wunsch, Clustering, vol. ١٠. Hoboken, NJ, USA: Wiley, ٢٠٠٨.

G.Karypis, E.-H. Han, and V. Kumar, Chameleon: Hierarchical cluster ing using dynamic modeling, Computer, vol. ٣٢, no. ٨, pp. ٦٨٧٥, ١٩٩٩.

S. Guha, R. Rastogi, and K. Shim, Rock: A robust clustering algorithm for categorical attributes, Inf. Syst., vol. ٢٥, no. ٥, pp. ٣٤٥٣٦٦,Jul.٢٠٠٠.

P. Andritsos, P. Tsaparas, R. J. Miller, and K. C. Sevcik, LIMBO: Scalable clustering of categorical data, in Advances in Database Technology EDBT٢٠٠٤,E.Bertino,S.Christodoulakis,D. Plexousakis, V. Christophides, M. Koubarakis, K. Böhm, and E. Ferrari, Eds. Berlin, Germany: Springer, ٢٠٠٤, pp. ١٢٣١٤٦, doi: ١٠,١٠٠٧/٩٧٨-٣-٥٤٠ ٢٤٧٤١-٨_٩.

R. Bahgat, M. A. Mahdi, S. E. Abdel-Rahman, and I. A. Ismail, Frequency tree for clustering categorical data with similarity measure based on items weights, in Qatar Foundation Annual Research Forum, vol. ٢٠١٢, no. ١. Doha, Qatar: Hamad bin Khalifa Univ. Press (HBKU Press), ٢٠١٢.

M. Li, S. Deng, L. Wang, S. Feng, and J. Fan, Hierarchical clustering algorithm for categorical data using a probabilistic rough set model, Knowl.-Based Syst., vol. ٦٥, pp. ٦٠٧١, Jul. ٢٠١٤.

H. Qin, X. Ma, T. Herawan, and J. M. Zain, MGR: An information theory based hierarchical divisive clustering algorithm for categorical data, Knowl.-Based Syst., vol. ٦٧, pp. ٤٠١٤١١, Sep. ٢٠١٤.

Reinduction-based approach for efficient exploration of high-utility item sets from incremental datasets

Hadi Naseri
Member of the Faculty of Computer Engineering, Islamic Azad University, Estehban Branch, Estehban, Iran
Fatemeh Zare
Master student of Islamic Azad University, Estehban Branch, Estehban, Iran

Abstract

High-utility itemset exploration is an important research area that focuses on identifying combinations of itemsets in databases that have a utility value higher than a user-specified threshold. However, most algorithms in this field assume that databases are static and unchanging. This assumption is not very realistic in the real world, as real-world datasets are constantly growing and changing with new data. In addition, existing algorithms rely only on the utility value to identify sets of related items. This leads to the fact that even compounds that occurred in the early stages are also produced as outputs. Although some mining algorithms adopt a support-based approach to calculate the frequency of itemsets, they do not consider the temporal nature of itemsets. To address these challenges, this paper proposes the Scented Utility Miner (SUM) algorithm, which is based on a The re-induction strategy uses incremental databases to track the recency of occurrences of item sets and to explore item sets.This paper presents a novel approach to explore high utility item sets from dynamic databases and proves the effectiveness of the proposed method by presenting several experiments.

Keywords:Data mining, Big Data, Clustering , high dimensional data, parallel clustering.