روشی جدید در تشخیص تکراری رکوردها با استفاده از خوشه‌بندی سلسله‌مراتبی

نگین دانشپور*[1] و علی برزگری[2]

[1]دانشکده مهندسی کامپیوتر، دانشگاه تربیت دبیر شهید رجایی، تهران، ایران

[2]دانشکده فنی مهندسی، گروه کامپیوتر، دانشگاه آزاد اسلامی واحد علوم تحقیقات، تهران، ایران

چکیده

به‌دلیل اهمیت بالای کیفیت داده‌ها در عملکرد سامانه‌های نرم‌افزاری، فرآیند پاک‌سازی داده به‌خصوص تشخیص رکوردهای تکراری، طی سالیان اخیر یکی از مهم‌ترین حوزه‌های علوم رایانه به حساب آمده است. در این مقاله روشی برای تشخیص رکوردهای تکراری ارائه شده است که با خوشه‌بندی سلسله‌مراتبی رکوردها بر اساس ویژگی‌های مناسب در هر سطح، میزان شباهت میان رکوردها تخمین زده می‌شود. این کار سبب می‌شود تا خوشه‌هایی در سطح آخر به‌دست آیند که رکوردهای درون آن‌ها بسیار مشابه یکدیگر باشند. برای کشف رکوردهای تکراری نیز مقایسه تنها بر روی رکوردهای درون یک خوشه از سطح آخر انجام می‌گیرد. همچنین در این مقاله برای مقایسه میان رکوردها، یک تابع تشابه نسبی بر پایه تابع فاصله ویرایشی ارائه شده که دقت بسیار بالایی به همراه دارد. مقایسه نتایج ارزیابی سامانه نشان می‌دهد که روش ارائه‌شده، در زمان کمتری، ۹۰٪ تکراری‌های موجود را با دقت ۹۷٪ کشف می‌کند و بهبود داشته است.

واژگان کلیدی: تشخیص تکراری، پاک‌سازی داده، خوشه‌بندی سلسله‌مراتبی، تابع تشابه، انتخاب ویژگی.

A New Method for Duplicate Detection Using Hierarchical Clustering of Records

Negin Daneshpour[1*] & Ali Barzegari[2]

[1]Faculty of Computer Engineering, Shahid Rajaee Teacher Training University, Tehran, Iran

[2]Faculty of Engineering-Department of Computer, Islamic Azad University Science And Research Branch, Tehran, Iran

Abstract

Accuracy and validity of data are prerequisites of appropriate operations of any software system. Always there is possibility of occurring errors in data due to human and system faults. One of these errors is existence of duplicate records in data sources. Duplicate records refer to the same real world entity. There must be one of them in a data source, but for some reasons like aggregation of data sources and human faults in data entry, it is possible to appear several copies of an entity in a data source. This problem leads to error occurrence in operations or output results of a system; also, it costs a lot for related organization or business. Therefore, data cleaning process especially duplicate record detection, became one of the most important area of computer science in recent years. Many solutions presented for detecting duplicates in different situations, but they almost are all time-consuming. Also, the volume of data is growing up every day. hence, previous methods don't have enough performance anymore. Incorrect detection of two different records as duplicates, is another problem that recent works are being faced. This becomes important because duplicates will usually be deleted and some correct data will be lost. So it seems that presenting new methods is necessary.

In this paper, a method has been proposed that reduces required volume of process using hierarchical clustering with appropriate features. In this method, similarity between records has been

* نویسندهٔ عهده‌دار مکاتبات

* Corresponding author

● تاریخ ارسال مقاله: ۱۳۹۸/۰۳/۲۸ ● تاریخ پذیرش: ۱۳۹۹/۰۵/۲۸ ● تاریخ انتشار: ۱۴۰۰/۱۲/۲۹ ● نوع مطالعه: پژوهشی

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]

[DOI: 10.52547/jsdp.18.4.3]

estimated in several levels. In each level, a different feature has been used for estimating similarity between records. As a result, clusters that contain very similar records will be created in the last level. The comparisons are done on these records for detecting duplicates. Also, in this paper, a relative similarity function has been proposed for comparing between records. This function has high precision in determining the similarity. Eventually, the evaluation results show that the proposed method detects 90% of duplicate records with 97% accuracy in less time and results have improved.

Keywords: Duplicate Record Detection, Data Cleaning, Hierarchical Clustering, Similarity Function, Feature Selection.

۱- مقدمه

امروزه داده‌ها مهم‌ترین دارایی یک سازمان هستند؛ اما انجام برخی از تراکنش‌ها مانند به‌روزرسانی، ورود دستی اطلاعات و تجمیع داده‌ها، ممکن است منجر به بروز خطاهایی در داده شود. اطلاعات اولیه مورد نیاز برای سامانه‌های مختلف به‌طور معمول از طریق منابع تولید داده مانند پایگاه داده‌های عملیاتی، تارنماها، صفحات گسترده و غیره جمع‌آوری می‌شوند. تجمیع اطلاعات از منابع داده مختلف ممکن است، باعث پدیدارشدن رکوردهای تکراری شود. وجود رکوردهای تکراری در مجموعه‌دادهٔ یک سامانه نرم‌افزاری، موجب اتلاف فضای ذخیره‌سازی، افزایش زمان پاسخ‌گویی سامانه و در برخی موارد بازگرداندن نتایج اشتباه به‌وسیلهٔ سامانه می‌شود [1]. این هزینه‌ها و مشکلات به ویژه در سامانه‌های تصمیم‌یار[1] و تحلیل‌گر داده که با استفاده از داده‌های پایگاه داده تحلیلی[2]، به گزارش‌دهی، تصمیم‌سازی و برنامه‌ریزی می‌پردازند، تأثیرات مخربی دارند [2]. حال در صورت وجود داده‌های تکراری و سایر مشکلات داده، این سامانه‌ها نتایج نادرستی تولید می‌کنند و آمار و تحلیل‌های به‌دست‌آمده از آن‌ها موجب اتخاذ تصمیمات اشتباه توسط مدیران می‌شود و به مرور زمان اعتماد مشتریان به سبب این اشتباهات کاهش می‌یابد و درنهایت ممکن است هزینه زیادی را بر سازمان تحمیل کند [3]. برای مقابله با این مشکل، فرآیند تشخیص تکراری[3] توسط پژوهش‌گران ارائه شده است که به شناسایی رکوردهای تکراری در یک مجموعه‌داده می‌پردازد.

رکوردهایی که از منابع مختلف جمع‌آوری شده‌اند اغلب دارای قالب‌های مختلف هستند. همچنین خطاهایی مانند مقادیر تهی و غلط‌های املایی در رکوردها قابل مشاهده است. به همین جهت مقایسه رکوردها برای تشخیص وجود تکراری کاری هزینه‌بر و دشوار است [4]. ساده‌ترین روش برای تشخیص رکوردهای تکراری، مقایسه دوبه‌دوی تمامی رکوردها و تعیین رکوردهای تکراری است؛ ولی با افزایش حجم اطلاعات و بزرگ‌ترشدن پایگاه داده‌ها، این روش به‌هیچ‌وجه عملی نیست؛ زیرا تعداد مقایسه‌ها بسیار زیاد می‌شود و انجام آن به زمان و منابع نامحدودی نیاز دارد [5].

با توجه به مشکلات موجود در زمینه تشخیص رکوردهای تکراری، معیارهای مختلفی مورد توجه قرار گرفته‌اند. زمان کلی فرآیند، یکی از معیارهایی است که در تمامی زمینه‌ها اهمیت بالایی دارد. از آنجایی که تشخیص تکراری یک فرآیند زمان‌بر است و بایستی بر روی حجم بالایی از داده‌ها پردازش انجام دهد، زمان کلی فرآیند در روش‌های ارائه‌شده حائز اهمیت است. یکی دیگر از معیارهای مورد توجه، دقت فرآیند تشخیص تکراری است؛ به این منظور که رکوردهایی که به‌درستی تکراری هستند شناسایی شوند. در بسیاری از موارد به‌علت تشابه زیاد میان رکوردها، فرآیند تشخیص تکراری، دو رکورد را که درواقع دو موجودیت متفاوت هستند، به‌عنوان تکراری تشخیص می‌دهد که این خطای فرآیند است. رکوردهای شناسایی‌شده درنهایت از مجموعه‌داده حذف می‌شوند. پس لازم است تا روش‌های ارائه‌شده از دقت کافی برخوردار باشند. معیار دیگری که درواقع هدف اصلی یک فرآیند تشخیص تکراری است، نرخ کشف رکوردهای تکراری است. با توجه به مجموعه‌داده‌های بسیار بزرگ کنونی که گاهی حجمشان به چندین ترابایت می‌رسد، مانند [6] که حدود ۵۴ میلیون دیدگاه ثبت‌شده در یک تارنما را شامل می‌شود، هزینه مقایسه رکوردها بسیار بالا می‌رود و فرآیند تشخیص تکراری به‌عنوان یک فرآیند بسیار پرهزینه شناخته می‌شود؛ و از آنجا که نرم‌افزارهای بی‌درنگ[4] نمی‌توانند پردازش این فرآیند را بیش‌تر از زمان مشخصی تحمل کنند، مهم است که یک روش بتواند در زمان محدودی، تعداد تکراری بیشتری را پیدا کند [7].

برای مقابله با مشکلات رکوردهای تکراری، راه‌کارهای مختلفی برای کاهش حجم پردازش مورد نیاز ارائه شده‌اند. بیش‌تر روش‌های ارائه‌شده در تلاش برای

[1] Decision support system
[2] Data Warehouse
[3] Duplicate Detection
[4] Real-Time Applications

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]

[DOI: 10.52547/jsdp.18.4.3]

کاهش زمان کل فرآیند تشخیص تکراری بوده‌اند. درحالی‌که این فرآیند به‌قدری پرهزینه و زمان‌بر است که با وجود این بهبودها باز هم در بسیاری از سامانه‌های عملیاتی غیر قابل اجرا است. همچنین در بیش‌تر روش‌ها که از دسته‌بندی رکوردها استفاده می‌کنند، به موضوع انتخاب ویژگی توجهی نشده است؛ درحالی‌که انتخاب ویژگی مناسب می‌تواند تأثیر به‌سزایی در نتایج نهایی داشته باشد. در این مقاله یک روش تشخیص تکراری جدید ارائه می‌شود که بتواند در زمان کوتاه‌تری تعداد تکراری بیشتری را نسبت به کارهای ارائه‌شده در گذشته کشف کند و خطای کمتری نیز داشته باشد. برای این منظور، روشی برای انتخاب ویژگی مناسب برای خوشه‌بندی رکوردها ارائه شده که مطابق با نحوهٔ خوشه‌بندی ارائه شده است. این نحوه خوشه‌بندی به این ترتیب است که رکوردها در چندین سطح به‌صورت سلسله‌مراتبی خوشه‌بندی می‌شوند. خوشه‌بندی در هر سطح بر روی خوشه‌های سطح بالاتر انجام می‌گیرد. در هر سطح از یک ویژگی متفاوت برای خوشه‌بندی استفاده می‌شود. به این ترتیب ساختار درختی از خوشه‌ها به‌وجود می‌آید که رکوردهای درون یک خوشه از سطح پایین درخت، بر اساس چندین ویژگی با یکدیگر مشابه هستند. خوشه‌بندی سلسله‌مراتبی در مسائلی که دارای حجم بالای پردازشی هستند کاربرد به‌سزایی دارد. همان‌طورکه در [39] از آن جهت شاخص‌گذاری اطلاعات تصویری به‌منظور بازیابی تصاویر استفاده شده است. این نوع خوشه‌بندی با کاهش قابل ملاحظه فضای جستجو، منجر به کاهش حجم پردازش نهایی و در نتیجه کاهش زمان کلی فرآیند می‌شود. برای مقایسه نهایی میان رکوردها، تابع فاصله ویرایشی[1] [8] بهبود پیدا کرده است تا با دقت بیشتری میزان شباهت میان رشته‌ها را محاسبه کند. عملکرد مناسب این تابع باعث افزایش تعداد تکراری‌های کشف شده و کاهش خطای فرآیند می‌شود.

به‌طور کلی نوآوری‌های این مقاله عبارتند از:

- ارائهٔ روشی برای انتخاب ویژگی مناسب برای خوشه‌بندی سلسله‌مراتبی رکوردها
- استفاده از ویژگی‌های متفاوت برای هر سطح از خوشه‌بندی
- ارائه یک تابع مقایسه میان رکوردها با اقزایش دقت تابع فاصله ویرایشی

[1] Edit Distance

ساختار مقاله به این ترتیب است: در بخش ۲ به مرور برخی از کارهای انجام‌گرفته در حوزه تشخیص رکوردهای تکراری پرداخته می‌شود. در بخش ۳ روش پیشنهادی برای تشخیص رکوردهای تکراری به تفکیک مراحل، شرح داده می‌شود. در بخش ۴ نتایج اجرای روش بر روی چندین مجموعه‌داده ارائه می‌شود و مقایسه نتایج این روش با چند روش موجود انجام می‌گیرد. در بخش ۵ نیز نتیجه‌گیری کلی انجام می‌گیرد.

۲- پیشینه پژوهش

روش‌های مختلفی برای تشخیص تکراری ارائه شده که خوشه‌بندی از پرتکرارترین آنها است. در [9] یک دسته‌بندی از روش‌های خوشه‌بندی ارائه شده است که طبق آن خوشه‌بندی به دو دسته با نظارت[2] و بدون نظارت[3] تقسیم می‌شود. خوشه‌بندی‌های بدون نظارت (خودکار) خود به دو دسته تفکیکی[4] و سلسله‌مراتبی تقسیم می‌شوند. بیش‌تر کارهای ارائه‌شده در حوزه تشخیص تکراری، در دسته خوشه‌بندی تفکیکی قرار می‌گیرند. خوشه‌بندی تفکیکی نیز به دو دسته انحصاری[5] و غیر انحصاری[6] تقسیم می‌شود. در روش‌های انحصاری هیچ رکوردی درون بیش از یک خوشه قرار نمی‌گیرد؛ ولی در روش‌های غیرانحصاری این حالت ممکن است. همچنین یک ویژگی برای الگوریتم‌های خوشه‌بندی تفکیکی معرفی شده است که نشان می‌دهد الگوریتم تک‌گذره[7] یا چندگذره[8] است. در الگوریتم‌های تک‌گذره برای خوشه‌بندی تنها یک مرتبه پویش بر روی فهرست ورودی رکوردها انجام می‌گیرد، این در حالی است که الگوریتم‌های چندگذره چندین مرتبه بر روی فهرست ورودی پویش و پردازش انجام می‌دهند. با این تعریف مشخص است که خوشه‌بندی‌های سلسله‌مراتبی چندگذره است؛ زیرا در هر سطح سلسله‌مراتب، دست‌کم یک مرتبه بر روی لیست ورودی پویش صورت می‌گیرد. یکی از این راه‌کارهای خوشه‌بندی، استفاده از الگوریتم SNM[9] است که جزء روش‌های غیر انحصاری است. به این ترتیب که رکوردها بر اساس میزان شباهتشان در یک ویژگی مرتب،

[2] Supervised
[3] Unsupervised
[4] Partitional
[5] Exclusive
[6] Non-Exclusive
[7] Single-pass
[8] Multi-pass
[9] Sorted Neighborhood Method

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]
[DOI: 10.52547/jsdp.18.4.3]

سپس پنجره‌ای با اندازه محدود بر روی فهرست مرتب‌شده رکوردها حرکت داده و تنها رکوردهایی با یکدیگر مقایسه می‌شوند که درون یک پنجره قرار گرفته باشند [10]. در یکی از روش‌هایی که از SNM استفاده می‌کند [11]، رکوردها بر اساس چندین ویژگی که دارای اهمیت بیشتری در مجموعه‌داده هستند، مرتب می‌شوند. برای هر مرتب‌سازی، با نمونه‌برداری و تقریب گرفتن از تعداد تکراری‌های پیداشده بررسی می‌شود که مرتب‌سازی مورد نظر مناسب است یا خیر. الگوریتم SNM بر روی مرتب‌سازی‌های انتخاب‌شده اعمال می‌شود. در این روش برای تطابق رکوردها از الگوریتم Smith-Waterman [12] استفاده شده است. همچنین به ویژگی‌های مهم‌تر وزن بیشتری داده می‌شود تا دقت روش بیشتر شود. در این روش درنهایت تنها یک مرتب‌سازی با استفاده از تقریب جهت تشخیص تکراری‌ها انتخاب می‌شود که علاوه‌بر زمان‌بربودن این کار، دقت آن نیز به‌دلیل تقریبی‌بودن انتخاب، متغیر است.

در روش دیگری که مبتنی بر SNM است [13]، با توجه به تعداد تکراری‌های یافته‌شده، اندازه و سرعت حرکت پنجره تنظیم می‌شود. به این ترتیب اگر در بخشی از داده تراکم رکورد تکراری کم باشد، سریع‌تر از آن بخش گذر می‌شود. به این ترتیب زمان اجرای الگوریتم بهبود قابل توجهی دارد. مشکلی که این روش دارد این است که خوشه‌بندی تنها بر اساس یک ویژگی صورت گرفته و راه‌کاری نیز برای انتخاب ویژگی ارائه نشده است. در این صورت با وجود تفاوت میان رکوردهای تکراری، ممکن است درصد قابل توجهی از تکراری‌ها یافت نشوند.

روش دیگری [14] با ترکیب روش‌های قدیمی مبتنی بر مقایسه مکرر رکوردها و روش‌های مبتنی بر SNM، سعی کرده تا از مزیت‌های هر دو روش بهره‌مند شود. در این روش ابتدا با استفاده از یک پنجره لغزنده، رکوردهای مشابه یکدیگر را پیدا کرده و در یک خوشه قرار می‌دهد. این خوشه‌ها در یک صف پردازش قرار می‌گیرند. سپس خوشه‌ها به‌ترتیب فراخوانی و رکوردهای درون آن‌ها با یکدیگر مقایسه می‌شوند. پس از انجام پردازش بر روی خوشه‌ها، ادغام بین خوشه‌های فهرست صورت می‌پذیرد تا با انجام مقایسه‌های بیشتر، رکوردهای تکراری بیشتری کشف شود. در این روش نقطه پایانی برای مقایسه میان رکوردها در نظر گرفته نشده است. به این ترتیب تعداد مقایسه‌های کم اهمیت زیادی انجام می‌گیرد که شانس کمی برای یافتن رکورد تکراری دارند. در مجموعه‌داده‌های بزرگ نیز زمان اجرا بسیار زیاد می‌شود.

در مقاله [15]، دو تابع تطابق و ادغام معرفی شده است. تابع تطابق با استفاده از تابع Jaro-Winkler [16] که مناسب برای مقایسه رشته‌های کوتاه است و یک مقدار آستانه، تشابه دو رکورد را مشخص می‌کند. تابع ادغام بر روی دو رکوردی اعمال می‌شود که به‌وسیلهٔ تابع تطابق مشابه تشخیص داده شوند. در ادغام دو رکورد یک رکورد جدید ساخته می‌شود که دارای اطلاعات بیشتر است. برای کاهش تعداد مقایسه‌ها، دو مجموعه تشکیل می‌شود که اولی شامل رکوردهایی است که هنوز مقایسه نشده‌اند و دومی شامل رکوردهای نتیجه نهایی است و در ابتدا رکورد نخست در آن قرار می‌گیرد. در هر مرتبه یک رکورد از مجموعه نخست با تمامی رکوردهای مجموعه دوم با تابع تطابق مقایسه می‌شود. اگر تطابقی یافت شد، رکورد جدیدی به‌وسیله تابع ادغام ساخته شده و به مجموعه نخست اضافه می‌شود و هر دو رکورد قبلی حذف می‌شوند. اگر تطابقی یافت نشد، رکورد مقایسه‌شده از مجموعه نخست حذف و به مجموعه دوم اضافه می‌شود. الگوریتم تا تهی‌شدن مجموعه نخست ادامه دارد. استفاده از تابع ادغام در این روش کمک می‌کند تا تکراری‌های بیشتری کشف شوند. از طرفی خطای روش را نیز بالا می‌برد. تعداد مقایسه‌های نهایی نیز کاهش چندانی نداشته و همچنان از مرتبه دو است.

یکی دیگر از راه‌کارهای خوشه‌بندی برای کاهش حجم پردازش مورد نیاز در فرآیند تشخیص تکراری، بلاک‌بندی[1] رکوردها است که در دسته روش‌های انحصاری قرار می‌گیرد. روش‌های مبتنی بر بلاک‌بندی با انتخاب یک کلید، اقدام به بلاک‌بندی رکوردها می‌کنند تا رکوردهای مشابه یکدیگر را در کنار هم قرار دهند. در مقاله [17] یک روش بلاک‌بندی با استفاده از مفهوم مجموعه اقلام تکرارشونده بیشینه[2] [18] معرفی شده است. در این روش با استفاده از ترکیبی از توابع تشابه، مقداری برای میزان شباهت میان اعضای هر بلاک محاسبه می‌شود. همچنین ضوابط مجموعه فشرده[3] و محله پراکنده[4] [19] بر روی هر بلاک پیاده می‌شود تا تعداد مقایسه‌ها کاهش یابد. هزینه محاسبه نمره برای هر بلاک در این روش زیاد است. در مجموعه‌داده‌هایی که تنوع

[1] Blocking
[2] Maximal Frequent Itemsets (MFI)
[3] Compact Set
[4] Sparse Neighborhood

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]
[DOI: 10.52547/jsdp.18.4.3]

موجودیت در آن‌ها زیاد است، تعداد بلاک زیادی ساخته می‌شود که زمان اجرا بسیار بالا می‌رود.

در روشی دیگر [20]، ابتدا بلاک‌بندی استاندارد SB [21] با استفاده از چند کلید بر روی رکوردهای چندین مجموعه‌داده انجام می‌گیرد؛ سپس رکوردهای درون هر بلاک با یکدیگر مقایسه می‌شوند و گرافی وزن‌دار شکل می‌گیرد که در آن رکوردها به‌صورت رئوس نمایش داده می‌شوند و بین رکوردهای مطابق، ارتباط برقرار است؛ همچنین میزان تشابه نیز وزن هر یال است؛ سپس با روش خوشه‌بندی اجزای متصل‌شده[1]، مجموعه‌هایی از رکوردهای تکراری را تشکیل می‌دهد. در این روش خوشه‌بندی، هر رأس به‌صورت مکرر همسایه‌های مستقیمش را به خوشه خودش اضافه می‌کند. در این روش، مرحله خوشه‌بندی بعد از بلاک‌بندی اضافه شده تا صحت رکوردهای مشابه را بررسی کند و رکوردهای مشابه جا مانده را اضافه کند. در این حالت باید انتظار داشت که این روش درصد تکراری بالا و خطای کمی داشته باشد. با این حال عملکرد این روش وابستگی زیادی به‌نحوه بلاک‌بندی و تابع مقایسه دارد که از روش‌های قدیمی برای آن‌ها استفاده شده است. به‌همین دلیل همچنان مشکل تعداد زیاد مقایسه‌ها و زمان اجرای طولانی برطرف نشده است.

سبک جدیدی از روش‌های تشخیص تکراری، روش‌های تصاعدی هستند. هدف این روش‌ها شناسایی درصد بیشتری از تکراری‌ها در لحظات ابتدایی فرآیند است. در این روش هرچه زمان بگذرد تعداد تکراری کمتری در واحد زمان یافت می‌شود. در مقاله [22] چهار روش تصاعدی بر مبنای SNM و بلاک‌بندی ارائه شده است. در روش مبتنی بر SNM اندازه پنجره در ابتدا کوچک انتخاب می‌شود تا رکوردهای نزدیک‌تر به هم زودتر مقایسه شوند و بعد از هر گام، به اندازه پنجره اضافه می‌شود. در این روش نیز نقطه پایانی برای اندازه پنجره در نظر گرفته نشده و هرچه از شروع فرآیند بگذرد مقایسه‌ها کم اهمیت‌تر می‌شوند و شانس یافتن تکراری کاهش میابد. در روش بلاک‌بندی نیز بعد از مقایسه رکوردهای درون هر بلاک، بلاکی که تعداد تکراری بیشتری داشته باشد؛ در ادامه گسترش داده و با بلاک‌های کناری ادغام می‌شود. ایده به‌کاررفته در این روش احتمالی است، به این صورت که چون در یک دسته تکراری زیادی یافت شده باید آن را با همسایه‌اش ادغام کرد تا تکراری بیشتری یافت شود؛ درحالی‌که ممکن است این اتفاق نیافتد. در دو روش دیگر، SNM و بلاک‌بندی با استفاده از چند ویژگی و در چند مرتبه پیاده می‌شوند؛ به این صورت که در هر مرتبه با یک ویژگی متفاوت فرآیند انجام می‌گیرد تا تکراری‌های بیشتری کشف شود.

روش تصاعدی دیگر [23] که نوعی خوشه‌بندی سلسله‌مراتبی محسوب می‌شود، با مرتب‌کردن زوج‌رکوردهای نامزد بر اساس میزان احتمال مشابه بودنشان، مقایسه بین رکوردها را بر اساس این ترتیب انجام می‌دهد تا رکوردهای تکراری سریع‌تر کشف شوند. این مرتب‌سازی با استفاده از توابع تخمین فاصله بین رکوردها انجام می‌گیرد که به‌مراتب هزینه کمتری نسبت به تابع مقایسه نهایی رکوردها دارد. تابع تخمین فاصله بین تمام زوج‌رکوردهای موجود در مجموعه‌داده اعمال می‌شود که بسیار زمان‌بر است. همچنین تخمین شباهت تنها بر اساس یک ویژگی صورت می‌گیرد که در صورت وجود تفاوت بین دو رکورد تکراری در آن ویژگی، آن دو رکورد شناسایی نمی‌شوند.

یکی از روش‌های دیگر برای کشف رکوردهای تکراری روش LSH[2] است [24] که در دسته‌بندی روش‌های انحصاری قرار می‌گیرد. این روش توسط توابع درهم‌ساز، یک مجموعه از نزدیک‌ترین همسایه‌ها را برای هر رکورد به‌دست می‌آورد که برای پیداکردن تکراری‌ها از آن مجموعه استفاده می‌شود؛ درواقع تعداد مقایسه‌های نهایی را کاهش می‌دهد. از این روش در مقاله [25] استفاده شده است. در این روش ابتدا از روی عناوین محصولات مختلف لغات کلیدی استخراج می‌شود و با استفاده از وجود یا عدم وجود هر یک از این لغات در عنوان یک محصول، بردار باینری به ازای هر محصول به‌دست می‌آید؛ سپس به‌علت زیادبودن طول بردارهای دودویی، با استفاده از Min-hashing و تعدادی جایگشت به طول تعداد لغات کلیدی، هر بردار دودویی به یک بردار نشان تبدیل می‌شود. بردارهای نشان که ماتریس نشان را شکل می‌دهند، ورودی الگوریتم LSH هستند. برای اعمال LSH، ماتریس به b دسته تقسیم می‌شود که هر دسته شامل r ردیف است. طبق LSH اگر دو محصول برای دست‌کم یک دسته خود با یک محصول دیگر یکسان باشند، دو محصول به‌عنوان زوج نامزد در نظر گرفته می‌شوند؛ سپس با استفاده از MSM[3] [26] (روش تشابه چندبخشی) تکراری‌بودن زوج‌های نامزد بررسی می‌شود.

[1] Connected Components

[2] Locality Sensitive Hashing

[3] Multi-component Similarity Method

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]

[DOI: 10.52547/jsdp.18.4.3]

تعداد مقایسه‌های نهایی با استفاده از LSH در این روش کاهش زیادی داشته، ولی محاسبات مرحله دسته‌بندی برای مجموعه‌های بزرگ زیاد است.

در جدول (۱) مقایسه‌ای بر روی کارهای ارائه‌شده در این بخش صورت گرفته است.

درنهایت می‌توان گفت که کارهای انجام‌شده در این حوزه بیشتر با استفاده از دسته‌بندی رکوردهای مشابه و انجام مقایسه‌های بیشتر سعی در افزایش نرخ کشف رکوردهای تکراری داشته‌اند. درواقع مهمترین مرحله و ایده اصلی کارهای ارائه‌شده، نحوه دسته‌بندی رکوردها و ترتیب انجام مقایسه‌ها بوده است. با این حال، عدم توجه به موضوعات انتخاب ویژگی، کاهش تعداد مقایسه‌های نهایی و همچنین دقت تابع مقایسه رکوردها، نتایج نهایی را تحت تاثیر قرار داده است و باعث شده بیش‌تر روش‌ها در پارامترهای مقیاس‌پذیری و دقت دچار ضعف باشند. در این مقاله تلاش شده تا با گروه‌بندی دقیق‌تر با استفاده از ویژگی‌های مناسب، تعداد مقایسه‌های نهایی و درواقع حجم پردازشی تا حدود زیادی کاهش یابد. ازاین‌رو از رویکرد خوشه‌بندی استفاده شده تا فرآیند مقایسه تنها بر روی شبیه‌ترین رکوردها انجام پذیرد. همچنین با بهبود تابع مقایسه، سعی شده دقت نتایج افزایش یابد.

(جدول-۱): مقایسه روش‌های پیشین
(Table-1): Comparison of previous methods

شماره مرجع روش	نوآوری روش	گذر	کارآیی	مقیاس‌پذیری	دقت
[11]	نمونه‌برداری و تقریب گرفتن از تعداد تکراری‌های پیدا شده پس از مرتب‌سازی جهت تشخیص مناسب‌بودن مرتب‌سازی	چند	متوسط	ضعیف	متوسط
[13]	تنظیم اندازه و سرعت حرکت پنجره بر روی رکوردها با توجه به تعداد تکراری‌های یافت شده پس از مرتب‌سازی	تک	متوسط	خوب	ضعیف
[14]	ترکیب روش مبتنی بر مقایسه مکرر رکوردها و روش SNM – ادغام خوشه‌ها بعد از انجام مقایسه‌ها	تک	خوب	متوسط	متوسط
[15]	معرفی یک تابع ادغام و اعمال آن بر روی زوج رکوردهایی که توسط تابع تطابق مشابه تشخیص داده شده‌اند و استفاده از رکورد ساخته شده در تطابق‌های بعدی	چند	متوسط	ضعیف	متوسط
[17]	استفاده از مفهوم مجموعه اقلام تکرار شونده بیشینه که نیازی به انتخاب ویژگی ندارد. اعمال ویژگی‌های مجموعه فشرده و محله پراکنده بر روی بلاک‌ها برای کاهش تعداد مقایسه‌ها	تک	متوسط	متوسط	متوسط
[20]	بعد از بلاک‌بندی و مقایسه رکوردهای درون هر بلاک، به ازای هر رکورد خوشه‌بندی می‌کند تا دقت و نرخ کشف رکوردهای تکراری افزایش یابد	چند	خوب	خوب	ضعیف
[22]	افزایش اندازه پنجره بعد از هر گذر – ادغام بلاکی که بیشترین تکراری در آن یافت شده با بلاک‌های کناری – تکرار این روش‌ها با استفاده از چند ویژگی	چند	خوب	متوسط	متوسط
[23]	مرتب کردن زوج رکوردهای کاندید بر اساس میزان احتمال مشابه بودنشان با استفاده از توابع تخمین فاصله که هزینه کمتری نسبت به تابع مقایسه نهایی دارا می‌باشد و انجام مقایسه نهایی بر اساس این ترتیب تا رکوردهای تکراری سریع‌تر کشف شوند	چند	خوب	متوسط	خوب
[25]	تبدیل عناوین به بردارهای باینری و فشرده‌کردن بردارها با استفاده از درهم‌سازی و درنهایت استفاده از LSH برای مشخص‌کردن زوج‌های کاندید و کاهش مقایسه‌های نهایی	تک	متوسط	متوسط	خوب

۳- روش پیشنهادی

در این بخش ابتدا فرآیند تشخیص رکوردهای تکراری و مراحل آن معرفی، سپس روش پیشنهادی این مقاله ارائه می‌شود.

۳-۱- فرآیند تشخیص رکوردهای تکراری

یکی از رایج‌ترین روش‌ها که در بیش‌تر کارهای گذشته [11]، [13]، [14]، [22]، [23]، [7]، [27]، [28] و [29]، از آن استفاده شده است، روش خوشه‌بندی رکوردها است. در روش پیشنهادی این مقاله نیز از خوشه‌بندی استفاده شده است. خوشه‌بندی پیشنهادی این مقاله دارای سه مرحله است که این مراحل به همراه زیرگام‌ها در نمودار فعالیت فرآیند در شکل (۱) مشخص است.

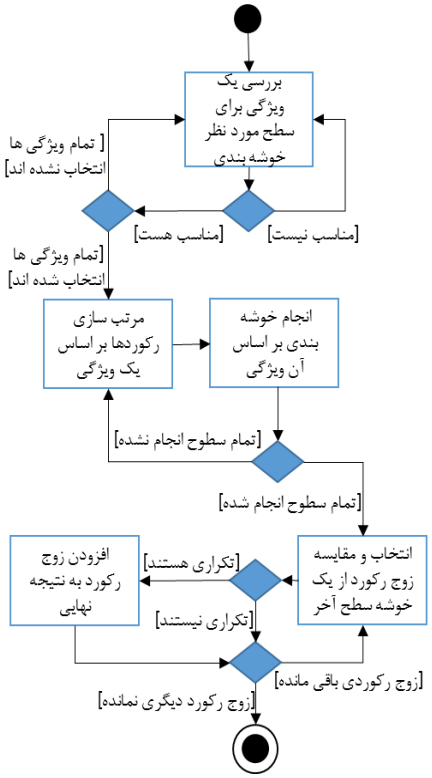
[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]
[DOI: 10.52547/jsdp.18.4.3]

۱) انتخاب ویژگی: در خوشه‌بندی از چند ویژگی مجموعه‌داده برای تخمین فاصله میان رکوردها استفاده می‌شود. انتخاب این که کدام ویژگی‌ها برای انجام این کار استفاده شوند، یکی از مراحل مهم روش‌های خوشه‌بندی است.

۲) خوشه‌بندی رکوردها: در این مرحله میزان شباهت میان رکوردها تخمین زده می‌شود و بر این اساس رکوردهایی که شباهت بیشتری دارند در یک خوشه قرار می‌گیرند.

۳) مقایسه رکوردها: این مرحله بعد از خوشه‌بندی رکوردها، بین رکوردهایی که در یک خوشه قرار گرفته‌اند، انجام می‌گیرد. این بخش شامل ترتیب مقایسه میان رکوردها، نحوه تشخیص دو رکورد تکراری و تابع مقایسه دو مقدار است.

در روش‌های تشخیص تکراری با استفاده از خوشه‌بندی، به‌طورمعمول برای هر یک از این مراحل راه‌کارهای متفاوتی ارائه شده است. در ادامه این مقاله به‌صورت دقیق‌تر هر یک از این مراحل مورد بررسی قرار گرفته و روش پیشنهادی برای هرکدام تشریح شده است.



(شکل-۱): نمودار فعالیت فرآیند تشخیص تکراری

(Figure-1): Activity diagram of duplicate detection process

۲-۳- انتخاب ویژگی

به‌طور کلی انتخاب ویژگی، پیداکردن یک زیرمجموعه از ویژگی‌ها است که برای هدف مورد نظر اطلاعات لازم و کافی را در برداشته باشد [30]. در فرآیند تشخیص رکوردهای تکراری، هدف انتخاب ویژگی عبارت است از انتخاب ویژگی‌هایی از مجموعه‌داده که توزیع مقادیرشان به‌گونه‌ای باشد تا خوشه‌بندی رکوردها به شکل مطلوبی انجام گیرد. در این مقاله برای انتخاب ویژگی از دو روش دستی و خودکار استفاده شده است؛ روش دستی توسط فرد انجام می‌گیرد و برای مجموعه‌داده با تعداد ویژگی‌های کم مناسب است. روش خودکار نیز یکی از کارهای انجام‌گرفته در گذشته است که فرآیند انتخاب ویژگی را پیاده‌سازی کرده است.

۱) روش دستی: این روش متناسب با نوع خوشه‌بندی پیشنهادی در این مقاله است. درواقع یک‌سری قواعد با توجه به جنبه‌های معنایی و ساختاری مجموعه‌داده ایجاد شده که بر اساس آن‌ها بتوان بهترین ویژگی‌ها را برای خوشه‌بندی رکوردها انتخاب کرد. برای مثال تعداد مقادیر تهی برای هر ویژگی که هرچه بیشتر باشد، نشان از کم‌اهمیت‌بودن آن ویژگی دارد و بر اساس این قاعده، ویژگی‌ای که مقادیر تهی زیادی دارد مناسب برای خوشه‌بندی نیست.

در جنبه معنایی، نیاز به داشتن دانش کافی از مجموعه‌داده وجود دارد. بایستی ماهیت ویژگی‌ها مشخص باشد؛ باید مشخص باشد که مقادیر یک ویژگی چه معنایی را می‌رسانند تا بتوان از آن‌ها در راستای هدف که تشخیص رکوردهای تکراری است، درست استفاده کرد. برای مثال مقدار ویژگی تاریخ تولد برای یک فرد همیشه ثابت است؛ پس می‌شود از این ویژگی برای خوشه‌بندی رکوردها استفاده کرد؛ ولی سن برای یک فرد مقدار ثابتی ندارد و در طول زمان تغییر می‌کند. اگر ویژگی سن برای خوشه‌بندی انتخاب شود، دو رکورد تکراری که مشخصات یک فرد در دو سال مختلف است، در یک خوشه قرار نمی‌گیرند و درنهایت تشخیص داده نمی‌شوند. میزان تأثیرگذاری ویژگی‌ها در شناسایی موجودیت‌ها نیز بسیار حائز اهمیت است. برای مثال ویژگی شماره شناسایی به‌طورمعمول بایستی یکتا باشد و یک فرد را مشخص کند. در حالی که ویژگی سطح تحصیلات در شناسایی یک فرد کمک چندانی نمی‌کند، ولی می‌تواند افراد را طبقه‌بندی کند.

در جنبه ساختاری، در ابتدا تعداد ویژگی‌های مجموعه‌داده اهمیت دارد. تعداد ویژگی‌ها بایستی به تعدادی باشند که به‌راحتی بتوان به‌صورت دستی

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]

[DOI: 10.52547/jsdp.18.4.3]

بررسی‌های مورد نیاز را انجام داد. در گام بعدی نوع ویژگی‌ها مطرح می‌شود که باید شناسایی شوند. برای مثال ویژگی نشانی از نوع رشته‌ای و بلند است و مشخص است که برای خوشه‌بندی مناسب نیست. مورد دیگر تعداد مقادیر تهی برای هر ویژگی است. تعداد زیاد مقادیر تهی برای یک ویژگی، نشان‌دهندهٔ اهمیت پایین آن ویژگی در مجموعه‌داده است. مورد آخر، بررسی میزان یکتابودن مقادیر ویژگی است. هرچه تعداد مقادیر یکتا برای یک ویژگی کمتر باشد، آن ویژگی رکوردها را به‌صورت درشت‌تری خوشه‌بندی می‌کند و بر عکس. برای مثال ویژگی شناسه رکورد که در تمام مجموعه داده‌ها است و برای هر رکورد مقدار یکتایی دارد، در صورت انتخاب به‌عنوان ویژگی خوشه‌بندی، به تعداد رکوردها خوشه ایجاد می‌کند و درواقع هیچ کاری انجام نمی‌دهد.

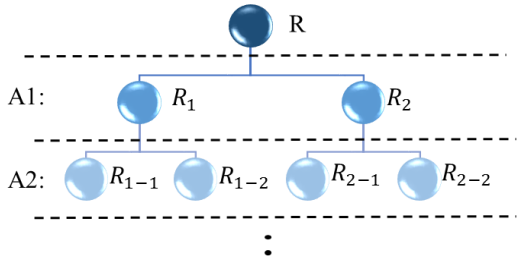
در ارتباط با روش خوشه‌بندی سلسله‌مراتبی که در این مقاله ارائه خواهد شد، گفتنی است که برای هر سطح خوشه‌بندی، یک ویژگی مورد نیاز است. ویژگی انتخاب‌شده برای هر سطح باید به‌گونه‌ای باشد که خوشه‌های سطح بالا شامل رکوردهای بیشتری یا به‌اصطلاح بزرگ‌تر باشند و هرچه به سطوح پایین‌تر می‌رسیم، خوشه‌ها کوچک‌تر شوند. برای این کار اندازه‌گیری میزان یکتابودن مقادیر ویژگی بسیار حائز اهمیت است.

در این مقاله با توجه به اینکه تعداد ویژگی‌های مجموعه‌داده‌ها خیلی زیاد نیست و ماهیت آنها نیز مشخص است، از روش دستی برای انتخاب ویژگی‌ها استفاده شده است. به این ترتیب برای انتخاب ویژگی سطح اول خوشه‌بندی، ویژگی انتخاب می‌شود که علاوه بر اینکه تمامی جنبه‌های ساختاری و معنایی را رعایت می‌کند، میزان یکتابودن مقادیرش کم باشد تا خوشه‌های کمتر و با اندازه بزرگ‌تری ایجاد کند. برای این کار ویژگی‌هایی که نوع طبقه‌بندی دارند، مانند شهر یا کشور، بیش‌تر مواقع مناسب هستند؛ اما برای انتخاب ویژگی سطوح پایین‌تر، ویژگی مناسب است که میزان یکتایی مقادیرش زیاد باشد. برای مثال شماره شناسایی فرد یا عنوان مقاله می‌تواند انتخاب‌های مناسبی باشد.

۲) روش خودکار: در برخی مواقع که دانش کافی از مجموعه‌داده مورد نظر وجود ندارد، یا اینکه تعداد ویژگی‌های مجموعه‌داده به‌قدری زیاد است که امکان اجرای روش دستی نیست، از روش خودکار استفاده می‌کنیم. روش خودکار استفاده‌شده، روش انتخاب تک‌متغیره[1] [8] است که جزو روش‌های فیلتر گذاری است. در این روش آزمون‌های آماری بر روی مقادیر ویژگی‌ها اعمال می‌شود تا ویژگی‌هایی انتخاب شوند که ارتباط قوی‌تری با متغیر خروجی دارند. متغیر خروجی، ویژگی ردهٔ رکوردها است. ویژگی رده در حوزه کاری تشخیص رکوردهای تکراری مشخص کننده این است که آیا دو رکورد تکراری هستند یا خیر. این ویژگی برای رکوردهایی که با هم یکی هستند مقدار مشترک دارد و برای سایر رکوردها فاقد مقدار است. در فرآیند انتخاب ویژگی، وجود ویژگی رده الزامی است. این ویژگی در مجموعه‌داده‌ها به منزله مقدار خروجی در فرآیند یادگیری با نظارت[2] [31] است. در فرآیند یادگیری با نظارت، هر رکورد مجموعه‌داده، زوجی شامل یک شیء ورودی و یک مقدار خروجی دلخواه است. یک الگوریتم یادگیری با نظارت، داده‌ها را تحلیل می‌کند تا ارتباط بین مقادیر ورودی و خروجی را به‌دست آورد تا پس از آن بتواند برای هر مقدار ورودی، مقدار خروجی مناسب را حدس بزند؛ درنتیجه ویژگی رده در انتخاب ویژگی، نقش متغیر خروجی را دارد و بایستی ارتباط بین مقادیر سایر ویژگی‌ها با مقادیر این ویژگی مورد ارزیابی قرار گیرد. در روش خودکار تک‌متغیره، آزمون آماری chi-square [32] بر روی ویژگی‌ها انجام می‌گیرد. این آزمون وابستگی میان مقادیر هر ویژگی با ویژگی رده را اندازه‌گیری می‌کند. هرچه میزان وابستگی بیشتر باشد، ویژگی مورد نظر بهتر است و انتخاب می‌شود.

۳-۳- خوشه‌بندی سلسله‌مراتبی

در روش خوشه‌بندی پیشنهادی، هدف ساخت سلسله‌مراتبی از خوشه‌ها است. برای این کار نیاز به چندین سطح خوشه‌بندی است که در هر سطح، از یک ویژگی متفاوت که در مرحله قبل انتخاب شده است، استفاده می‌شود. شکل (۲) نحوه تشکیل سلسله‌مراتب خوشه‌ها را نشان می‌دهد.



(شکل-۲): سلسله مراتب خوشه‌ها

(Figure-2): Hierarchy of clusters

[1] Univariate Selection
[2] Supervised Learning

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]
[DOI: 10.52547/jsdp.18.4.3]

در ابتدا تمام رکوردهای موجود در مجموعه‌داده در خوشه R قرار دارند که نقش گره پدر در سلسله‌مراتب را ایفا می‌کند. در هر سطح میزان شباهت میان رکوردهای یک خوشه بر اساس ویژگی متناظر با آن سطح تخمین زده و خوشه‌های جدید ایجاد می‌شوند. در ابتدا رکوردهای درون خوشه R بر اساس مقادیر ویژگی A1 با یکدیگر مقایسه می‌شوند تا خوشه‌های سطح اول ایجاد شوند. همان‌طور که در شکل (۲) مشخص است، خوشه‌های R_1 و R_2 در سطح اول ایجاد شده‌اند. برای سطح دوم، رکوردهای هر خوشه در سطح اول بر اساس ویژگی A2 مقایسه می‌شوند تا خوشه‌های سطح دوم ایجاد شوند. به همین ترتیب تا جایی که نیاز باشد می‌توان خوشه‌بندی را ادامه داد. برای تعیین تعداد سطوح خوشه‌بندی، تعداد مقایسه‌های نهایی مورد نیاز پس از هر سطح خوشه‌بندی را محاسبه کرده و میزان کاهش این تعداد نسبت به سطح قبلی مقایسه می‌شود. اگر کاهش چشم‌گیری ایجاد نشده باشد، خوشه‌بندی تا همان سطح کافی است. در بیش‌تر مواقع خوشه‌بندی تا سطح دوم کافی است.

برای ایجاد هر خوشه در سطح اول، مقادیر ویژگی A1 در تمام رکوردهای مجموعه‌داده که درون خوشه R قرار دارند بایستی به‌صورت دو به دو مقایسه شوند. تعداد کل زوج‌رکوردهای یک مجموعه‌داده از رابطه (۱) به‌دست می‌آید:

$$\frac{n \times (n-1)}{2} \qquad (۱)$$

که در آن n تعداد رکوردها است؛ بنابراین اگر بخواهیم مقادیر ویژگی تمام زوج‌رکوردها را با یکدیگر مقایسه کنیم، بسیار زمان‌بر خواهد بود. همچنین تعداد رکوردهایی که مقادیر ویژگی مشابهی دارند، محدود است و نیازی نیست که برای تمام زوج‌رکوردها این مقادیر مقایسه شوند.

در این روش، ابتدا رکوردها بر اساس مقادیر ویژگی A1 و به‌ترتیب حروف الفبا و ترتیب صعودی اعداد مرتب می‌شوند؛ سپس رکوردها در پنجره‌هایی با اندازه مشخص قرار می‌گیرند تا رکوردهای درون هر پنجره با یکدیگر مقایسه شوند. اندازه پنجره باید در حدی باشد که ضمن کم نگه‌داشتن تعداد مقایسه‌ها، پیوستگی زنجیره نیز حفظ شود. اگر اندازه پنجره‌ها خیلی کوچک باشد، در صورت بروز یک اشتباه زنجیره قطع می‌شود و برخی از رکوردهای مشابه در یک خوشه قرار نمی‌گیرند. هرچه اندازه پنجره‌ها بزرگ‌تر باشد، نیز تعداد زوج‌رکوردهایی که باید مقایسه شوند، زیاد می‌شود و زمان زیادی برای خوشه‌بندی نیاز خواهد بود. اگر اندازه پنجره را w و تعداد کل رکوردها را n در نظر بگیریم، تعداد مقایسه‌های مورد نیاز برای خوشه‌بندی در یک سطح از رابطه (۲) به‌دست می‌آید:

$$\frac{n}{w} \times \sum_{w=1}^{w-1} w \qquad (۲)$$

در این رابطه سمت چپ برابر تعداد پنجره‌های ایجاد شده و سمت راست برابر تعداد مقایسه‌های مورد نیاز در هر پنجره است. در شکل (۳) نحوه قرارگیری رکوردها در پنجره‌ها نشان داده شده است. مقادیر ویژگی تمام رکوردهای موجود در هر پنجره با یکدیگر مقایسه می‌شوند. اگر میزان شباهت دو مقدار از آستانه[1] بیشتر باشد، دو رکورد در یک خوشه قرار می‌گیرند. در غیر این صورت خوشه جدیدی ایجاد می‌شود و رکورد جدید در آن قرار می‌گیرد. در اینجا رابطه تعدی نیز برقرار است، برای مثال اگر دو رکورد r1 و r2 در یک خوشه بودند و میزان شباهت r1 و r3 کمتر از مقدار آستانه، ولی میزان شباهت r2 با r3 از آستانه بیشتر بود، r3 نیز به خوشه دو رکورد قبلی اضافه می‌شود. پس از مقایسه مقادیر درون یک پنجره، مقدار ویژگی رکورد آخر پنجره بالایی با رکورد اول پنجره بعدی مطابق فلش‌های شکل (۳)، با یکدیگر مقایسه می‌شوند تا هیچ دو رکورد مشابهی شانس قرارگرفتن در یک خوشه را از دست ندهند.

برای ساخت خوشه‌های سطوح پایین‌تر نیز به همین صورت عمل می‌شود؛ با این تفاوت که از ویژگی متناظر با سطح خوشه‌بندی برای تخمین شباهت میان رکوردها استفاده می‌شود. همچنین به جای کل رکوردهای موجود، عملیات تنها بر روی رکوردهای یک خوشه از سطح بالاتر انجام می‌گیرد؛ همچنین از انجام عملیات بر روی خوشه‌هایی که تعداد رکورد کمی دارند جلوگیری می‌شود؛ زیرا نه تنها سودی ندارد، بلکه زمان زیادی نیز هدر می‌رود.

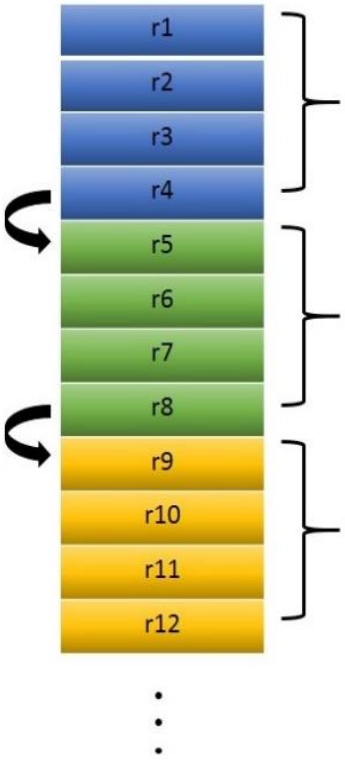
در سلسله‌مراتب خوشه‌ها، گره‌های سطوح پایین‌تر تعداد رکوردهای کمتری دارند. این رکوردها در ویژگی‌های بیشتری مشترک هستند؛ زیرا در هر سطح از خوشه‌بندی، از ویژگی متفاوتی استفاده شده است. این موضوع سبب می‌شود تا مقایسه‌هایی درنهایت باقی بمانند که شانس زیادی برای کشف تکراری‌ها دارند؛ درواقع سرعت کشف رکوردهای تکراری افزایش می‌یابد.

[1] Threshold

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]

[DOI: 10.52547/jsdp.18.4.3]

در الگوریتم (۱) شبه‌کد تابع خوشه‌بندی سلسله‌مراتبی ارائه شده است.



(شکل-۳): نحوه مقایسه رکوردها در فرآیند ایجاد خوشه‌ها
(Figure-3): Comparing records in the process of creating clusters

ورودی‌های تابع شامل سلسله‌مراتب خوشه‌ها (H)، سطح فعلی سلسله‌مراتب (i)، ویژگی انتخاب‌شده برای خوشه‌بندی (f)، مقدار آستانه (th) و اندازه پنجره (Wsize) است. در شروع کار H[0][0] فهرستی شامل کلیه رکوردهای مجموعه‌داده است. در خط ۲ مشخص شده که به‌ازای هر خوشه از سطح قبلی سلسله‌مراتب، الگوریتم اجرا می‌شود. در خط ۳، نخستین رکورد خوشه جاری از سطح بالاتر، به نخستین خوشه سطح پایینی اضافه می‌گردد. سپس به‌ازای هر پنجره به اندازه Wsize، دو رکورد از پنجره در هر مرحله انتخاب می‌شوند و مقدار ویژگی f آن‌ها به‌وسیلهٔ تابع تشابه نسبی مقایسه می‌شود (خطوط ۴و۵). اگر میزان تشابه بیشتر از مقدار آستانه th بود، دو رکورد در یک خوشه مشابه قرار می‌گیرند. در غیر این صورت رکورد دوم در خوشه جدیدی قرار می‌گیرد (خطوط ۶ تا ۱۰). اگر دو رکورد در مقایسه‌های قبلی خوشه‌بندی شده باشند و تطابق نیز حاصل شود، قبل از اضافه‌کردن رکورد دوم به خوشه رکورد اول، رکورد دوم بایستی از خوشه قبلی خود حذف شود (خطوط ۱۱ تا ۱۴). این فرآیند تا جایی که تمامی خوشه‌های سطح قبلی به‌وسیلهٔ پنجره پویش شود، ادامه پیدا می‌کند؛ درنهایت در خط آخر، سلسله‌مراتب خوشه‌ها که سطح جدید به آن اضافه شده است، بازگردانده می‌شود. درواقع خروجی الگوریتم (H) فهرستی از فهرست‌ها می‌باشد که به‌صورت سلسله‌مراتبی توانایی ذخیره خوشه‌های رکوردها را دارا است.

الگوریتم ۱: شبه کد خوشه‌بندی سلسله مراتبی

Input: H: Hirarchy of Clusters, i: Level of Hirarchy, f: Selected Feature, th: Threshold, Wsize: Window Size
Outout: H
Begin
1. **Function** Clustering(H,i,f,th,Wsize)
2. **for each** cluster of previous level:
3. H[i-1][ClusterId][0] **add to** H[i][0]
4. **for each** window with size Wsize:
5. s = Edite-Likeness(f from two records of window)
6. **if** one of records did not cluster:
7. **if** s > th:
8. second record add to firs record's cluster
9. **else:**
10. second record add to a new cluster
11. **else if** both records clustered before:
12. **if** s > th and they weren't in a same cluster:
13. **del** second record from it's cluster
14. second record **add to** first record's cluster
15. **return** H
End

۴-۳- مقایسه رکوردها

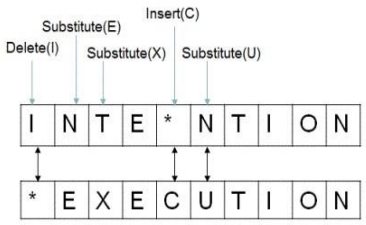
بخش اصلی و پرهزینه فرآیند تشخیص رکوردهای تکراری، مقایسه زوج‌رکوردها است. درواقع در این بخش زوج‌رکوردهای منتخب در یک فرآیند با یکدیگر مقایسه می‌شوند تا مشخص شود که تکراری یکدیگر یا اینکه دو رکورد مجزا هستند.

همان‌طور که در مراحل قبلی مشخص شد، فرآیند مقایسه رکوردها تنها بر روی خوشه‌های پایین‌ترین سطح سلسله‌مراتب اجرا می‌شود تا صرفه‌جویی در زمان فرآیند انجام گیرد. فرآیند به‌ترتیب از نخستین خوشه آغاز می‌شود. تمام رکوردهایی که درون خوشه قرار دارند بایستی با تمام رکوردهای دیگر هم‌خوشه خود مقایسه شوند. نحوه مقایسه به این صورت است که تک‌تک مقادیر ویژگی‌های دو رکورد به‌وسیلهٔ تابع فاصله ویرایشی نسبی که در ادامه ارائه خواهد شد، مقایسه می‌شوند تا میزان شباهتشان به‌دست آید. اگر میزان شباهت دو مقدار ویژگی بیشتر از آستانه بود، دو رکورد از لحاظ آن ویژگی مشابه در نظر گرفته می‌شوند. پس از اتمام مقایسه تمامی مقادیر ویژگی، اگر تعداد ویژگی‌هایی که دو رکورد در آن مشترک

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]
[DOI: 10.52547/jsdp.18.4.3]

هستند از مقدار آستانه دیگری بیشتر بود، دو رکورد تکراری تشخیص داده می‌شوند. به همین ترتیب برای خوشه‌های دیگر آخرین سطح عمل می‌شود.

تابع تشابه ویرایشی نسبی که در این مقاله ارائه شده با افزایش دقت تابع فاصله ویرایشی [8] به‌دست آمده است. در تابع فاصله ویرایشی، فاصله دو مقدار رشته‌ای برابر تعداد تغییراتی است که یک رشته لازم دارد تا به رشته دیگر تبدیل شود؛ این تغییرات شامل حذف، تغییر و اضافه‌کردن نویسه است.



(شکل-۴): عملکرد تابع فاصله ویرایشی
(Figure-4): Operation of edit distance Function

در این مثال فاصله ویرایشی دو رشته برابر پنج است؛ اما ایراد اصلی این روش این است که نسبت اندازه رشته‌های ورودی را در نظر نمی‌گیرد. فاصله پنج بین دو رشته کوتاه باقطعیت فاصله زیادی است؛ اما همین فاصله بین دو رشته بلند شاید کم به نظر آید و بتوان نتیجه گرفت که دو رشته بلند شبیه یکدیگر هستند. با استفاده از همین نکته در این پژوهش، تشابه ویرایشی نسبی تعریف شده است. نکته اولی که در این تابع حائز اهمیت است، محاسبه میزان شباهت میان دو رشته به جای میزان تفاوت است. این کار به‌منظور ساده‌کردن مفاهیم فرآیند صورت گرفته است. اگر طول رشته اول برابر L1، طول رشته دوم برابر L2 و تعداد نویسه‌های مشابه دو رشته که در یک موقعیت قرار دارند برابر i باشد، میزان شباهت دو رشته از رابطه (۳) به‌دست می‌آید:

$$\frac{2\times i}{L_1+L_2} \qquad (۳)$$

درواقع تعداد نویسه‌های مشابه در دو رشته، تقسیم بر مجموع طول دو رشته شده تا میزان شباهت نسبی به‌دست آید. مقدار محاسبه‌شده عددی بین صفر و یک خواهد بود و هرچه بزرگ‌تر باشد، نشان‌دهنده تشابه بیشتر دو رشته است.

الگوریتم (۲) شبه کد تابع تشابه ویرایشی نسبی را نشان می‌دهد.

الگوریتم ۲: شبه کد تابع تشابه ویرایشی نسبی

Input: str1: A String Value, str2: Another String Value
Output: Likeness Value of str1 & str2
Begin
1. **function** Edite-Likeness(str1,str2)
2. **if** str1 or str2 **is** null:
3. **return** 0
4. **if** Length(str1) < Length(str2):
5. **set** Lmin and Lmax
6. check both str1 and str2 **in** lowercase;
7. i = 0
8. **for** n = 0 to Lmin:
9. **if** str1[n] == str2[n]:
10. i += 1
11. **return** 2*i/(Lmin+Lmax)

End

تابع تشابه نسبی، دو رشته نویسه را به‌عنوان ورودی می‌گیرد. در ابتدا چک می‌شود که هیچ‌کدام از رشته‌ها تهی نباشند در غیر این صورت مقدار صفر برای تشابه دو رشته برگردانده می‌شود (خطوط ۲و۳). در خط ۶ هر دو رشته به حروف کوچک تبدیل می‌شوند؛ سپس بین خطوط ۷ تا ۱۰ در حلقه، نویسه‌های متناظر میان دو رشته مقایسه می‌شوند و تعداد نویسه‌های مشابه محاسبه شده و طبق فرمول ارائه‌شده در خط ۱۱، میزان تشابه میان دو رشته بازگردانده می‌شود.

فرآیند مقایسه رکوردها نیز مطابق الگوریتم (۳) می‌باشد.

ورودی تابع فرآیند مقایسه، سلسله مراتب خوشه‌ها (H)، آخرین سطح سلسله‌مراتب (i)، کلیه ویژگی‌های مجموعه‌داده (A)، آستانه اول (th1) و آستانه دوم (th2) است. گفته شد فرآیند مقایسه تنها بر روی خوشه‌های آخرین سطح سلسله‌مراتب انجام می‌گیرد؛ بنابراین در خط ۳، به‌ازای هر خوشه از سطح آخر سلسله‌مراتب، الگوریتم اجرا می‌شود. در خط ۴ در هر مرحله دو رکورد مجزا از خوشه انتخاب، سپس تک‌تک مقادیر ویژگی‌های متناظر دو رکورد توسط تابع تشابه نسبی مقایسه می‌شوند؛ اگر میزان شباهت میان دو مقدار ویژگی بیشتر از آستانه نخست بود، دو رکورد از لحاظ آن ویژگی مشترک در نظر گرفته می‌شوند و یک واحد به شمارنده اضافه می‌شود (خطوط ۵ تا ۸). درنهایت اگر تعداد ویژگی‌هایی که میان دو رکورد مشترک هستند، بیشتر از آستانه دوم بود، دو رکورد به فهرست تکراری‌ها اضافه می‌شوند (خطوط ۹ تا

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]
[DOI: 10.52547/jsdp.18.4.3]

۱۳). فرآیند تا جایی که تمامی رکوردهای هر خوشه دو به دو با یکدیگر مقایسه شوند ادامه پیدا می‌کند؛ درنهایت فهرستی شامل رکوردهای تکراری به‌عنوان نتیجه بازگردانده می‌شود.

الگوریتم ۳: شبه کد فرآیند مقایسه رکوردها

Input: H: Hirarchy of Clusters, i: Last Level of Hirarchy, A: Set of DataSet Attributes, th1: Threshold1, th2: Threshold2
Output: result: List of Detected Duplicate Records
Begin
1. **Function** Comparing(H,i,A,th1,th2)
2. **list** result
3. **for each** cluster of level i:
4. **for each** two records m,n of cluster:
5. f = 0
6. **for each** a in A:
7. **if** Edite-Likeness(H[i][ClusterId][m][a], H[i][ClusterId][n][a]) > th1:
8. f += 1
9. **if** f > th2:
10. **if** m **not in** result:
11. m **add to** result
12. **if** n **not in** result:
13. n **add to** result
14. **return** result
End

۴- نتایج

در این بخش، نتایج حاصل از اجرای روش پیشنهادی بر روی چندین منبع داده محاسبه و ارائه، سپس این نتایج با نتایج تعدادی از روش‌های گذشته مقایسه می‌شود. روش پیشنهادی به‌وسیلهٔ زبان برنامه‌نویسی پایتون[1] ۳.۶.۰ و در محیط مایکروسافت ویژوال استودیو[2] پیاده‌سازی و در سیستمی با هشت هسته با سرعت ۲.۲۰ GHz و ۷.۵ GB RAM اجرا شده است.

۴-۱- مجموعه‌داده‌ها

در این کار از مجموعه‌داده‌های واقعی استفاده شده است؛ زیرا در این مجموعه داده‌ها مقادیر تهی و اشتباه وجود دارد و همچنین دو رکورد تکراری به‌طور حتم به‌صورت کامل شبیه یکدیگر نیستند. همچنین سعی شده تا مجموعه‌داده‌هایی که انتخاب می‌شوند از لحاظ خصوصیات با یکدیگر متفاوت باشند.

۱. مجموعه‌داده رستوران: دارای ۸۶۴ رکورد است که ۱۱۲ زوج‌رکورد آن تکراری است. تعداد ویژگی‌ها نیز ۵ عدد است [33].
۲. مجموعه‌داده CD: دارای ۹۴۳۱ رکورد است که ۴۲۵ رکورد آن تکراری است. تعداد ویژگی‌های آن هجده عدد است. همچنین مقادیر تهی، خطاهای تایپی و تفاوت‌های فراوان میان دو رکورد تکراری در این مجموعه داده دیده می‌شود [34].
۳. مجموعه‌داده CORA: این مجموعه‌داده شامل ۱۸۷۹ رکورد است که از این بین ۱۸۱۵ رکورد دارای دست‌کم یک رکورد متناظر است. درواقع با یک مجموعه‌داده بسیار کثیف سر و کار داریم. تعداد ویژگی‌ها ۳۸ عدد است که نسبت به دو مجموعه‌داده قبلی بیشتر است. مقادیر تهی زیادی در این مجموعه‌داده وجود دارد [35].
۴. مجموعه‌داده حقوق سالیانه: تعداد رکوردها ۳۴۹۷۶۲ عدد است که نشان‌دهنده بزرگی مجموعه‌داده است. در نسخه اصلی مجموعه‌داده رکورد تکراری وجود ندارد؛ ولی برای آزمایش روش ارائه‌شده بر روی یک مجموعه‌داده بزرگ‌تر، این مجموعه‌داده انتخاب شده و از روی ۹۹۹۹ رکورد آن یک نسخه تکراری‌شده ایجاد شده تا مجموعه‌داده دارای ۹۹۹۹ زوج رکورد تکراری شود. تعداد ویژگی‌های مجموعه‌داده نیز ۸ عدد است [36].

۴-۲- تنظیمات و پیش فرض‌ها

در این روش از چندین مقدار پیش‌فرض استفاده شده است که نحوه تعیین مقدار هر کدام در این بخش بررسی می‌شود.

۴-۲-۱- اندازه پنجره‌های خوشه‌بندی

برای تعیین اندازه پنجره‌های مرحله خوشه‌بندی، سه متغیر تأثیر گذارند. زمان صرف شده برای مرحله خوشه‌بندی، تعداد مقایسه‌های نهایی انجام شده و تعداد تکراری‌هایی که به‌درستی شناسایی شده‌اند. اندازه پنجره باید به‌صورتی انتخاب شود که بیشترین تعداد تکراری در انتهای فرآیند کشف شود؛ در عین حال زمان صرف‌شده برای خوشه‌بندی و همچنین تعداد مقایسه‌های نهایی مورد نیاز، بیش از حد زیاد نشوند. در جدول (۲) روند تغییرات این سه متغیر با تغییر اندازه پنجره‌ها در هر چهار مجموعه داده، نشان داده شده است. همان‌طور که در جدول (۲) مشاهده می‌شود، اندازه پنجره از دو تا شش تغییر کرده و

[1] Python
[2] Microsoft Visual Studio

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]
[DOI: 10.52547/jsdp.18.4.3]

برای هر اندازه، متغیرهای یادشده برای هر چهار مجموعه‌داده محاسبه شده‌اند. برای هر مجموعه‌داده بهترین نتایج به‌صورت پر رنگ مشخص شده است.

(جدول-۲): تأثیرات تغییر اندازه پنجره در مرحله خوشه‌بندی بر روی متغیرها

(Table-2): The effects of window resizing in clustering stage on variables

اندازه پنجره	متغیر	رستوران	CD	CORA	حقوق سالیانه
2	زمان خوشه‌بندی	0.47	7.7	3.1	196
	تعداد مقایسه‌ها	104	1821	14143	187820
	تعداد تکراری	184	360	1643	19998
3	زمان خوشه‌بندی	**0.55**	7.6	**3.4**	**194**
	تعداد مقایسه‌ها	**105**	1847	**14171**	**187817**
	تعداد تکراری	**186**	360	**1645**	**19998**
4	زمان خوشه‌بندی	0.65	7.3	4.1	198
	تعداد مقایسه‌ها	105	1849	14239	187820
	تعداد تکراری	186	360	1644	19998
5	زمان خوشه‌بندی	0.76	**7.5**	4.5	215
	تعداد مقایسه‌ها	110	**1799**	14219	187817
	تعداد تکراری	186	**360**	1645	19998
6	زمان خوشه‌بندی	0.74	7.6	4.6	205
	تعداد مقایسه‌ها	107	1822	14171	187822
	تعداد تکراری	186	360	1645	19998

در اندازه پنجره سه، تعداد تکراری بیشتری نسبت به اندازه پنجره دو یافت شده، ولی زمان خوشه‌بندی و تعداد مقایسه‌ها کمی بیشتر است. از آنجایی که یافتن تکراری‌ها اهمیت بیشتری دارد، اندازه پنجره برای سه مجموعه‌داده سه انتخاب شده است. برای مجموعه‌داده CD نیز اندازه پنج انتخاب می‌شود؛ درنهایت با توجه به نحوه مقایسه رکوردها در فرآیند ایجاد خوشه‌ها مطابق با شکل (۳)، پیش‌بینی می‌شود افزایش اندازه پنجره از یک حدی بیشتر کمکی به افزایش تعداد تکراری‌های شناسایی‌شده نخواهد کرد و تنها زمان فرآیند را افزایش می‌دهد. در نتایج به‌دست‌آمده در جدول (۲) نیز مشخص است که نتایج در اندازه‌های چهار، پنج و شش پنجره تا حدودی پایدار شده‌اند؛ بنابراین روند آزمایش‌ها در اندازه پنجره شش پایان می‌یابد.

۴-۲-۲- تعداد سطوح خوشه‌بندی

یکی دیگر از تنظیمات، تعداد سطوح خوشه‌بندی است که مشخص می‌کند، خوشه‌بندی سلسله‌مراتبی تا چند سطح ادامه پیدا می‌کند. برای تعیین این مقدار، تعداد مقایسه‌های نهایی مورد نیاز پس از هر سطح خوشه‌بندی محاسبه و میزان کاهش این تعداد نسبت به سطح قبلی مقایسه می‌شود. اگر کاهش چشم‌گیری ایجاد نشده باشد، خوشه‌بندی تا همان سطح کافی است. تعداد مقایسه‌های مورد نیاز در هر سطح، وابسته به تعداد خوشه‌های آن سطح و اندازه هر خوشه است. تعداد کل مقایسه‌های ممکن هم از رابطه (۱) به‌دست می‌آید. جدول (۳) تعداد مقایسه‌های مورد نیاز پس از هر سطح خوشه‌بندی در چهار مجموعه‌داده را نشان می‌دهد. همچنین نسبت تعداد مقایسه‌ها در دو سطح پی در پی نیز در این جدول مشخص می‌باشد.

همان‌طور که در جدول (۳) مشخص است، برای هر چهار مجموعه‌داده، خوشه‌بندی تا سطح دوم به میزان چشم‌گیری تعداد مقایسه‌های نهایی مورد نیاز را کاهش می‌دهد. تنها در مورد مجموعه داده CORA میزان کاهش مقایسه‌ها از سطح نخسن به سطح دوم حدود $\frac{1}{3}$ است که علت آن زیاد بودن نسبت تعداد تکراری‌ها به کل رکوردها در این مجموعه‌داده است. در سه مجموعه‌داده دیگر میزان کاهش مقایسه‌ها در سطح دوم نسبت به سطح نخست، دست‌کم $\frac{1}{380}$ است. همچنین در هر چهار مجموعه‌داده مشخص است که سطح سوم خوشه‌بندی کارآیی چندانی نداشته و تعداد مقایسه‌ها را به نسبت سطح دوم، به میزان قابل قبولی کاهش نداده است.

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]

[DOI: 10.52547/jsdp.18.4.3]

(جدول-۳): تعداد مقایسه‌های مورد نیاز پس از هر سطح خوشه‌بندی

(Table-3): Number of comparisons required after each clustering level

مجموعه داده	کل مقایسه‌های ممکن	سطح اول	نسبت سطح اول	سطح دوم	نسبت سطح دوم	سطح سوم	نسبت سطح سوم
رستوران	372 816	40 253	0.11	105	2.8×10^{-4}	92	$\mathbf{2.4\times10^{-4}}$
CD	44 467 165	1 315 432	0.03	1 799	4×10^{-5}	1 017	$\mathbf{2.2\times10^{-5}}$
CORA	1 764 381	38 720	0.02	14 171	8×10^{-3}	10 307	$\mathbf{5.9\times10^{-3}}$
حقوق سالیانه	61 166 553 441	6 928 197 867	0.11	187 817	3×10^{-6}	98 547	$\mathbf{1.6\times10^{-6}}$

۴-۲-۳- مقادیر آستانه

در روش پیشنهادی، چهار مقدار آستانه وجود دارد که شامل آستانه خوشه‌بندی سطح اول، آستانه خوشه‌بندی سطح دوم، آستانه اول فرآیند مقایسه و آستانه دوم فرآیند مقایسه می‌شود. این آستانه‌ها بایستی به‌نحوی انتخاب شوند تا بهترین نتیجه ممکن به‌دست آید. نتایج شامل تعداد تکراری صحیح پیدا شده، تعداد تکراری که به‌اشتباه یافت شده و همچنین تعداد مقایسه‌های نهایی است. به همین جهت برای به‌دست‌آوردن مقادیر آستانه، الگوریتم چندین‌بار با تغییردادن مقادیر آستانه اجرا می‌شود. در نخستین اجرا یک مقدار متوسط برای هر آستانه در نظر گرفته و سپس تغییرات در یکی از آستانه‌ها برای هر اجرا آغاز می‌شود. هر زمان بهترین نتیجه با تغییر آستانه اول به‌دست آمد، نوبت به تغییر آستانه دوم می‌رسد. اگر پس از انتخاب آستانه اول، آستانه دیگری با تغییرش نتیجه بهتری به‌دست آورد، فرآیند آزمایش‌ها از ابتدا و با تغییر آستانه اول از سر گرفته می‌شود. سه آستانه اول، مطابق با نتیجه‌ای که توسط تابع تشابه نسبی حاصل می‌شود، عددی بین صفر تا یک می‌توانند داشته باشند. آستانه دوم فرآیند مقایسه نیز که تعداد ویژگی مشابه بین دو رکورد برای تثبیت تکراری بودنشان است، وابسته به تعداد ویژگی‌های مجموعه‌داده و عددی طبیعی بزرگتر مساوی یک است.

در انتها بهترین مقادیر کلیه تنظیمات و آستانه‌ها که برای هر مجموعه‌داده استخراج شده در جدول (۴) نشان داده شده است.

(جدول-۴): بهترین مقادیر تنظیمات و آستانه‌ها

(Table-4): The best values of settings and thresholds

مجموعه داده	اندازه پنجره خوشه‌بندی	تعداد سطوح خوشه‌بندی	آستانه خوشه‌بندی سطح اول	آستانه خوشه‌بندی سطح دوم	آستانه اول فرآیند مقایسه	آستانه دوم فرآیند مقایسه
رستوران	3	2	0.09	0.64	0.6	1
CD	5	2	0.48	0.45	0.34	5
CORA	3	2	0.41	0.31	0.09	2
حقوق سالیانه	3	2	0.9	0.9	0.9	6

۴-۳- ارزیابی الگوریتم خوشه‌بندی

با استفاده از روش ارزیابی Dunn-Index که یک روش ارزیابی داخلی است [37]، خوشه‌بندی‌های انجام‌شده برای هر چهار مجموعه‌داده مورد ارزیابی قرار گرفته است. در روش‌های ارزیابی داخلی بهترین نمره به الگوریتمی اختصاص می‌یابد که خوشه‌هایی با بیشترین شباهت داخلی و کمترین شباهت میان خوشه‌ای ایجاد کند. روش Dunn نیز طبق این تعریف یک نمره کیفی به خوشه‌بندی انجام شده اختصاص می‌دهد که مطابق رابطه (۴) محاسبه می‌شود.

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]

[DOI: 10.52547/jsdp.18.4.3]

$$D = \frac{\min_{1\le i<j\le n} d(i,j)}{\max_{1\le k\le n} d'(k)} \quad (۴)$$

در این رابطه d(i,j) فاصله بین دو خوشه و d'(k) فاصله درونی خوشه k است؛ بنابراین مقدار ارزیابی D برای یک الگوریتم خوشه‌بندی، با تقسیم کمترین فاصله موجود بین دو خوشه بر بیشترین فاصله درونی موجود برای یک خوشه به‌دست می‌آید. هرچه مقدار D بیشتر باشد، عملکرد الگوریتم خوشه‌بندی بهتر بوده است. در جدول (۵) نتایج ارزیابی خوشه‌بندی پیشنهادی بر روی هر چهار مجموعه‌داده مشخص شده است.

(جدول-۵): نتایج ارزیابی الگوریتم خوشه‌بندی
(Table-5): Evaluation results of clustering algorithm

مجموعه داده	Min d(i,j)	Max d'(k)	Dunn
رستوران	0.21	0.66	0.32
CD	0.56	0.47	1.2
CORA	0.78	0.25	3.12
حقوق سالیانه	0.52	0.29	1.8

مقادیر فاصله بین صفر تا یک محاسبه شده‌اند و هرچه بزرگتر باشند فاصله بیشتر است. برای محاسبه فاصله میان دو خوشه، رکورد اول هر خوشه با استفاده از تابع فاصله ویرایشی مقایسه شده است. تمامی خوشه‌ها با یکدیگر مقایسه شده‌اند تا کمترین فاصله محاسبه شود. برای محاسبه فاصله درونی هر خوشه نیز، فاصله تمامی رکوردهای آن خوشه با رکورد اول خوشه به‌وسیلهٔ تابع فاصله ویرایشی محاسبه شده و میانگین فاصله‌ها به‌عنوان فاصله درونی خوشه در نظر گرفته شده است؛ سپس بیشترین فاصله درونی برای یک خوشه به‌دست آمده که در جدول (۵) قابل مشاهده است. درنهایت مقدار Dunn برای هر یک از مجموعه‌داده‌ها محاسبه شده است که نشان می‌دهد عملکرد الگوریتم خوشه‌بندی بر روی سه مجموعه‌داده CD، CORA و حقوق سالیانه بسیار مطلوب بوده و خوشه‌هایی که تولید شده به‌خوبی از یکدیگر تفکیک شده‌اند. عملکرد الگوریتم بر روی مجموعه‌داده رستوران نیز متوسط بوده است.

۴-۴- پارامترهای مورد ارزیابی

پارامترهایی که در فرآیند تشخیص رکوردهای تکراری حائز اهمیت هستند شامل این موارد هستند:

- زمان کلی فرآیند: کل زمانی که طول می‌کشد برنامه بر روی یک مجموعه‌داده اجرا شود. می‌توان این زمان را به سه قسمت بارگذاری، پیش‌پردازش و پردازش اصلی تقسیم کرد. زمان بارگذاری مدت زمانی است که مجموعه‌داده بر روی RAM بارگذاری می‌شود که با توجه به حجم بالای مجموعه‌داده‌ها وجودش انکارناپذیر است. منظور از زمان پیش‌پردازش نیز زمان صرف‌شده برای خوشه‌بندی رکوردها است. درنهایت زمان پردازش اصلی، مدت زمانی است که رکوردهای درون خوشه‌های سطح آخر با یکدیگر مقایسه می‌شوند و تکراری‌ها یافت می‌شود. برای محاسبه دقیق‌تر این پارامتر، فرآیند را چندین مرتبه اجرا کرده و میانگین زمان‌های به‌دست‌آمده را محاسبه می‌کنیم.
- تعداد مقایسه‌های نهایی: زمان پردازش اصلی در مجموعه‌داده‌های بسیار بزرگ سهم اصلی را از زمان کل فرآیند دارد؛ زیرا هر مقایسه نهایی هزینه بالایی دارد. هرچه عملکرد مرحله خوشه‌بندی بهتر باشد، تعداد مقایسه‌های نهایی کاهش می‌یابد.
- نرخ فراخوانی[1]: نسبت تعداد تکراری‌های شناسایی‌شده به کل تکراری‌های موجود در مجموعه‌داده است.
- نرخ دقت[2]: نسبت تعداد رکوردهای تکراری یافت‌شده صحیح را به کل رکوردهای یافت‌شده محاسبه می‌کند. درواقع این پارامتر مشخص می‌کند که فرآیند چه مقدار خطا داشته است.

۴-۵- ارائه و ارزیابی نتایج

در جدول (۶) زمان کلی فرآیند به تفکیک زمان بارگذاری، پیش‌پردازش و پردازش اصلی برای هر چهار مجموعه‌داده نشان داده شده است. فرآیند بر روی هر مجموعه‌داده چندین‌بار اجرا شده تا محاسبه زمان دقیق‌تر انجام شود. درنهایت میانگین زمان این اجراها محاسبه شده و در جدول (۶) قابل مشاهده است.

برای مجموعه‌داده کوچک رستوران میانگین زمان کلی فرآیند برابر ۷/۴ ثانیه است. به‌علت کوچکی مجموعه‌داده مشاهده می‌شود که زمان بارگذاری بیشترین سهم را دارد. میانگین زمان برای مجموعه‌داده‌های CD، CORA و حقوق سالیانه نیز به‌ترتیب برابر ۲۳/۴، ۶۶/۷ و ۴۹۸ ثانیه بوده است.

تعداد مقایسه‌های نهایی مورد نیاز در اجرای فرآیند بر روی هر مجموعه‌داده در جدول (۷) مشخص است. در این جدول مشاهده می‌شود که تعداد مقایسه‌های نهایی برای تمام مجموعه‌داده‌ها به نسبت کل مقایسه‌های ممکن،

[1] Recall
[2] Precision

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]
[DOI: 10.52547/jsdp.18.4.3]

بسیار کاهش یافته است. این نتایج نشان می‌دهد زمانی که برای مرحله پیش‌پردازش فرآیند در جدول (۶) صرف شده، که در برخی موارد به اندازه زمان پردازش اصلی نیز طول کشیده است، نقش مؤثری در کاهش زمان کلی فرآیند داشته و درواقع مرحله خوشه‌بندی به‌خوبی عمل کرده است. به‌عنوان مثال در مجموعه‌داده حقوق سالیانه ۱۸۷۸۱۷ مقایسه نهایی در مرحله پردازش اصلی صورت گرفته که انجام این مقایسه‌ها به‌مدت ۲۵۸ ثانیه به‌طول انجامیده است. این تعداد مقایسه به نسبت کل مقایسه‌های ممکن در این مجموعه‌داده، ۳۲۵۶۷۱ مرتبه کاهش داشته است. این کاهش در مرحله خوشه‌بندی، یعنی در طول زمان پیش‌پردازش حاصل شده است. در واقع اگر مرحله خوشه‌بندی نبود، زمان پردازش اصلی به نسبت بسیار زیادی افزایش میافت و امکان پیاده‌سازی آن در محیط مشابه وجود نداشت.

نرخ فراخوانی و نرخ دقت حاصل از اجرای فرآیند بر روی هر چهار مجموعه‌داده، در شکل (۵) نمایش داده شده است. نرخ فراخوانی برای مجموعه‌داده‌های رستوران، CD، CORA و حقوق سالیانه به‌ترتیب ۸۳٪، ۸۵٪، ۹۱٪ و ۱۰۰٪ محاسبه شده که قابل توجه است. میانگین نرخ فراخوانی روش پیشنهادی برابر ۹۰٪ شده است. به این معنی که روش پیشنهادی ۹۰٪ رکوردهای تکراری در چهار مجموعه‌داده را یافت کرده است. نرخ دقت نیز برای این چهار مجموعه‌داده به‌ترتیب برابر ۹۵٪، ۹۴٪، ۹۹٪ و ۹۹/۸٪ به‌دست آمده است که میانگینی برابر ۹۷٪ است. مشخص است که روش پیشنهادی از دقت بسیار بالایی برخوردار است.

(جدول-۶): زمان کلی فرآیند به تفکیک مراحل

(Table-6): Process time for each step

مجموعه داده	زمان بارگذاری	زمان پیش پردازش	زمان پردازش اصلی	زمان کل
رستوران	6.76	0.5	0.14	7.4
CD	7.62	7.32	8.44	23.4
CORA	7.53	3.86	55.36	66.7
حقوق سالیانه	12	228	258	498

(جدول-۷): تعداد مقایسه‌های نهایی

(Table-7): Number of final comparisons

مجموعه داده	کل مقایسه‌های ممکن	مقایسه‌های نهایی
رستوران	372 816	105
CD	44 467 165	1 799
CORA	1 764 381	14 171
حقوق سالیانه	61 166 553 441	187 817

(شکل-۵): نرخ فراخوانی و نرخ دقت. (الف) مجموعه‌داده رستوران؛ (ب) مجموعه‌داده CD؛ (ج) مجموعه‌داده CORA؛ (د) برای مجموعه‌داده حقوق سالیانه

(Figure-5): Recall and precision rates. (A) Restaurant dataset; (B) CD dataset; (C) CORA dataset; (D) Salary dataset.

۶-۴- مقایسه نتایج

هدف اصلی روش ارائه‌شده در این مقاله، بهبود نتایج کارهای گذشته در زمینه تشخیص رکوردهای تکراری است؛ ازاین‌رو در این قسمت، نتایج به‌دست‌آمده توسط روش پیشنهادی، با سه روش مطرح گذشته مقایسه می‌شود. دلیل انتخاب روش‌های Famer [20]، Progressive [22] و Pay-As-You-Go [23] برای مقایسه با روش پیشنهادی این است که هر سه با نوآوری‌های خود

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]

[DOI: 10.52547/jsdp.18.4.3]

به نتایج قابل توجهی در حوزه تشخیص رکوردهای تکراری دست یافته‌اند. همچنین این روش‌ها در نشریات معتبری منتشر شده‌اند. شرح هر سه روش در بخش پیشینه پژوهش انجام شده و خلاصه‌ای از عملکرد آن‌ها به همراه تعداد دیگری از روش‌های این حوزه، در جدول (۱) ارائه گردیده است. برای مقایسه روش پیشنهادی با سه روش دیگر در شرایط یکسان، این روش‌ها نیز در محیطی مشابه پیاده‌سازی شده و بر روی چهار مجموعه‌داده اجرا شده‌اند. نتایج به‌دست‌آمده هر چهار روش برای پارامترهای زمان اجرا، تعداد مقایسه‌ها، نرخ فراخوانی و نرخ دقت به تفکیک هر مجموعه‌داده، در جدول (۸) مشخص است. بهترین نتیجه به‌ازای هر پارامتر در هر مجموعه‌داده به‌صورت پررنگ نمایش داده شده است.

(جدول-۸): مقایسه نتایج روش پیشنهادی و سه روش دیگر

(Table-8): Comparison of the results of the proposed method and three other methods

مجموعه داده	پارامتر ارزیابی	روش پیشنهادی	Famer	Progressive	Pay-As-You-Go
رستوران	زمان اجرا	**7.4**	52.4	11.8	7.9
	تعداد مقایسه‌ها	**105**	30821	12930	185
	نرخ فراخوانی	83%	**84%**	73%	76%
	نرخ دقت	**95%**	25%	93%	93%
CD	زمان اجرا	23.4	**17.1**	102	29.5
	تعداد مقایسه‌ها	1799	**357**	339462	4831
	نرخ فراخوانی	**85%**	77%	74%	83%
	نرخ دقت	**94%**	60%	90%	93%
CORA	زمان اجرا	66.7	**50.9**	55	74.6
	تعداد مقایسه‌ها	14171	**9425**	285228	16041
	نرخ فراخوانی	**91%**	83%	**91%**	89%
	نرخ دقت	**99%**	98%	97%	97%
حقوق سالیانه	زمان اجرا	**498**	594	1026	620
	تعداد مقایسه‌ها	**187817**	289648	5596168	209439
	نرخ فراخوانی	**100%**	**100%**	**100%**	**100%**
	نرخ دقت	**99.8%**	8%	99.3%	**99.8%**

از شانزده مورد مقایسه موجود در جدول (۸)، روش پیشنهادی در یازده مورد بهترین عملکرد را نسبت به روش‌های دیگر داشته است. از نتایج به‌دست‌آمده مشخص است که هر روشی که تعداد مقایسه‌های کمتری داشته، بهترین زمان اجرا را به‌دست آورده است. روش پیشنهادی در دو مجموعه‌داده رستوران و حقوق سالیانه کمترین زمان اجرا را داشته است و در دو مجموعه‌داده دیگر فاصله چندانی با روش Famer ندارد. همچنین داشتن کمترین تعداد مقایسه و زمان اجرا در مجموعه داده حقوق سالیانه، به‌دلیل بزرگی این مجموعه‌داده بسیار حائز اهمیت و نشان‌دهنده مقیاس‌پذیربودن روش پیشنهادی است. در پارامتر نرخ فراخوانی نیز روش پیشنهادی عملکرد قابل قبولی داشته و در سه مجموعه‌داده بهترین نتیجه را کسب کرده است. تنها در مجموعه‌داده رستوران، نرخ فراخوانی روش پیشنهادی ۱٪ کمتر از روش Famer بوده است. در مجموعه‌داده حقوق سالیانه، تمام رکوردهای تکراری موجود توسط هر چهار روش شناسایی شده‌اند و نرخ فراخوانی ۱۰۰٪ شده است. دلیل این مورد نیز شباهت کامل رکوردهای تکراری در این مجموعه‌داده است که کار کشف رکوردها را آسان می‌کند؛ اما بهترین عملکرد روش پیشنهادی در پارامتر نرخ دقت بوده است. مطابق نتایج جدول (۸)، دقت روش پیشنهادی در هر چهار مجموعه داده بیشترین مقدار را در مقایسه با سایر روش‌ها داشته است. میانگین نرخ دقت روش پیشنهادی ۹۷٪ بوده و تنها ۳٪ خطا در شناسایی رکوردهای تکراری داشته است. درنهایت اگر میانگین نتایج به‌دست‌آمده برای هر روش بر روی چهار مجموعه‌داده در نظر گرفته شود، روش Pay-As-You-Go در سه پارامتر نرخ فراخوانی، نرخ دقت و تعداد مقایسه‌ها، بین سه روش قدیمی بهترین عملکرد را داشته است؛ با این حال روش پیشنهادی در مقایسه با روش Pay-As-You-Go به‌صورت میانگین ۲/۷۵٪ نرخ فراخوانی و ۱/۲۵٪ نرخ دقت را افزایش داده است. همچنین تعداد مقایسه‌های نهایی در روش پیشنهادی ۱۱/۶٪ کاهش پیدا کرده است. در ارتباط با پارامتر زمان اجرا بین روش‌های قدیمی نیز، روش Famer بهترین عملکرد را داشته است. با این حال روش پیشنهادی به‌طور میانگین زمان اجرا را به نسبت روش Famer ۱۷٪ کاهش داده است.

یکی از دلایل اصلی بهبود روش جدید در تمام پارامترها، دقت بیشتر در خوشه‌بندی رکوردها و ساختار

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]

[DOI: 10.52547/jsdp.18.4.3]

سلسله‌مراتبی ایجاد شده است. طی آزمایش‌ها نیز مشاهده شد که زمان صرف شده برای پیش‌پردازش در روش جدید، نسبت به روش‌های قبلی بیشتر است. این نکته هرچند باعث می‌شود که کشف رکوردهای تکراری دیرتر آغاز شود، ولی با کاهش تعداد مقایسه‌های نهایی، زمان کلی اجرای فرآیند را کاهش می‌دهد. از طرف دیگر عملکرد بهتر مرحله خوشه‌بندی در روش جدید، در کشف رکوردهای تکراری بیشتر نیز تاثیرگذار است؛ همچنین ارائه تابع جدید جهت مقایسه میان رکوردها، باعث افزایش دقت روش پیشنهادی نسبت به روش‌های قدیمی شده است.

۴-۷- اهمیت بهبود نتایج

با استفاده از آمار می‌توان اهمیت بهبود نتایج را محاسبه کرد. در این راستا آزمون T-student [38] نتایج دو نمونه را با یکدیگر مقایسه و مشخص می‌کند بهبود نتایج چه میزان حائز اهمیت است. منظور از اهمیت آماری[1] این است که چه میزان می‌توان انتظار داشت که نتایج بر روی مجموعه‌داده‌های دیگر قابل تکرار باشند؛ زیرا نتایج بر روی تعداد محدودی مجموعه‌داده، ممکن است به‌صورت تصادفی حاصل شده باشد.

در این بخش، نرخ فراخوانی و دقت حاصل از اعمال روش پیشنهادی بر روی چهار مجموعه‌داده با سایر روش‌های جدول (۸)، به‌وسیلهٔ آزمون T نمونه‌های وابسته[2] مقایسه می‌شود. کاربرد این آزمون، مقایسه نتایج حاصل از دو وسیله اندازه‌گیری متفاوت بر روی یک شیء است. بنابراین روش‌ها دو به دو با یکدیگر مقایسه می‌شوند. آزمون به این ترتیب انجام می‌گیرد که ابتدا یک فرضیه صفر[3] تعریف می‌شود که میانگین نتایج دو روش با یکدیگر تفاوتی ندارد؛ سپس یک فرضیه جایگزین[4] مطرح می‌شود، که میانگین نرخ فراخوانی و دقت روش پیشنهادی بیشتر از روش‌های قبلی می‌باشد. سپس مقدار t طبق رابطه (۵) برای نتایج دو روش محاسبه می‌گردد. مقدار t یک نسبت بین تفاوت‌های بین دو گروه نتیجه و تفاوت‌های درون گروه‌ها می‌باشد.

$$t = \frac{(\sum D)/N}{\sqrt{\frac{\sum D^2 - (\frac{(\sum D)^2}{N})}{(N-1)(N)}}} \qquad (۵)$$

در رابطه (۵)، N تعداد مجموعه‌داده‌هایی است که بر رویشان آزمایش انجام گرفته و D_i حاصل تفریق نتیجه روش پیشنهادی با روش دیگر در مجموعه‌داده iام می‌باشد. در جدول (۹)، مقادیر t محاسبه شده میان روش پیشنهادی و هر یک از روش‌های مورد مقایسه به تفکیک برای نرخ فراخوانی و نرخ دقت نشان داده شده است.

(جدول-۹): مقادیر t محاسبه‌شده
(Table-9): Calculated t-values

t-values	معیار ارزیابی	Famer	Progressive	Pay-As-You-Go
روش پیشنهادی	نرخ فراخوانی	1.56	1.75	1.86
	نرخ دقت	2.46	3.07	2.6

درنهایت مقدار محاسبه‌شده t با مقدار t بحرانی مقایسه می‌شود. t بحرانی از روی جدول توزیع t و به‌ازای میزان فاصله اطمینان، درجه آزادی[5] و یک‌طرفه یا دوطرفه‌بودن فرضیه جایگزین، تعیین می‌شود. در اینجا فرضیه جایگزین یک طرفه و درجه آزادی نیز N-1=3 است. فاصله اطمینان نیز ۰/۱ انتخاب می‌شود به این معنی که وجود ۱۰٪ خطا قابل قبول است. بنابراین t بحرانی برابر ۱.۶۳۸ خواهد بود. این مقدار با هریک از مقادیر t محاسبه‌شده در جدول (۹) مقایسه می‌شود. از آنجایی که مقادیر t محاسبه‌شده در ۵ مورد از ۶ مورد موجود در جدول (۹) بزرگ‌تر از مقدار t بحرانی هستند، فرضیه صفر رد می‌شود و می‌توان نتیجه گرفت که نتایج حاصل‌شده توسط روش‌ها، قابل اطمینان و تکرارپذیر هستند. از سوی دیگر روش پیشنهادی طبق نتایج جدول (۸)، در معیارهای نرخ فراخوانی و نرخ دقت در اکثر موارد بهترین نتیجه را میان سایر روش‌ها به‌دست آورده است؛ بنابراین می‌توان نتیجه گرفت که این بهبود حاصل‌شده حائز اهمیت است. تنها در ارتباط با مقایسه نتایج نرخ فراخوانی روش پیشنهادی با روش Famer، مقدار t محاسبه‌شده کمتر از t بحرانی است که در این مورد نیز فاصله چندانی وجود ندارد.

۵- نتیجه‌گیری

امروزه در تمام زمینه‌هایی که با داده‌ها سر و کار دارند نیاز به روش‌های پاک‌سازی داده وجود دارد. برای تشخیص رکوردهای تکراری که بخشی از پاک‌سازی داده است، همواره بایستی روش‌هایی ارائه شود که علاوه‌بر این‌که زمان اجرای فرآیند کنترل شده باشد، تعداد تکراری بیشتری نیز کشف کند و در عین حال خطای کمتری نیز داشته باشد. در این مقاله برای کاهش زمان اجرا، ساختار سلسله‌مراتبی خوشه‌بندی رکوردها ارائه شد که باعث

[1] Statistical Significance
[2] Paired Samples T Test
[3] Null Hypothesis
[4] Alternative Hypotheses
[5] Degrees Of Freedom

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]
[DOI: 10.52547/jsdp.18.4.3]

می‌شود حجم مقایسه میان رکوردها به میزان چشم‌گیری کاهش یابد. همچنین در انتخاب ویژگی‌های مناسب برای خوشه‌بندی، توجه ویژه‌ای به بحث زمان مورد نیاز برای پردازش شده است. برای افزایش نرخ تکراری‌های کشف‌شده، علاوه‌بر خوشه‌بندی دقیق رکوردها، یک تابع تشابه نسبی ارائه شد که کمک زیادی به دقت بالای روش می‌کند. این تابع در پایین نگه‌داشتن نرخ خطای روش نیز تأثیر به‌سزایی دارد؛ درنهایت نتایج به‌دست‌آمده از اجرای روش بر روی چهار مجموعه‌داده و مقایسه این نتایج با روش‌های گذشته نشان داد که روش ارائه‌شده دستاوردهای بسیار خوبی داشته است.

با وجود پیشرفت در روش‌های جدید ارائه‌شده، همچنان مشکلاتی وجود دارد که زمینه کار در این حوزه را باز نگه می‌دارد. از جمله این مشکلات می‌توان به اهمیت بالای رکوردهایی که به‌اشتباه تکراری شناسایی و حذف می‌شوند اشاره کرد. می‌توان با رتبه‌بندی تکراری‌های شناسایی شده از نظر شباهت و کمک‌گرفتن از اپراتور انسانی روشی برای این مشکل ارائه داد. در رابطه با داده‌های غیر متنی از جمله داده‌های صوتی و تصویری نیز روش‌هایی برای پاک‌سازی آنها نیاز است که بایستی با کمک روش‌های پردازش صوت و تصویر ارائه شوند. همچنین می‌توان با استفاده از روش‌های تشابه معنایی، رکوردهایی را که از نظر معنایی مشابه هستند شناسایی نمود. این روش می‌تواند در مجموعه‌داده‌هایی که رکوردهایشان ماهیت متفاوتی با یکدیگر دارند مورد استفاده قرار گیرد.

۶- مراجع 6- References

[1] E. Rahm, and H.H. Do, "Data cleaning: Problems and current approaches", IEEE Data Eng. Bull., 23(4), pp. 3-13, 2000.

[2] L. Bradji, and M. Boufaida, "Knowledge based data cleaning for data warehouse quality", in Digital Information Processing and Communications., Springer. pp. 373-384, 2011.

[3] D.K. Koshley, and R. Halder, "Data cleaning: An abstraction-based approach. in Advances in Computing, Communications and Informatics (ICACCI)," 2015 International Conference on. 2015. IEEE.

[4] M. Alian, , A. Awajan, and B .Ramadan, "Unsupervised learning blocking keys technique for indexing Arabic entity resolution", International Journal of Speech Technology, pp. 1-8, 2018.

[5]Y. Li, , H. Wang, and H. Gao, "Efficient entity resolution based on sequence rules", in Advanced Research on Computer Science and Information Engineering, Springer. pp. 381-388, 2011.

[6] https://www.reddit.com/r/datasets/comments/3bxlg7/i_have_every_publicly_available_reddit_comment./

[7] Y. Altowim, , D.V. Kalashnikov, and S. Mehrotra, "ProgressER: Adaptive Progressive Approach to Relational Entity Resolution", ACM Transactions on Knowledge Discovery from Data (TKDD),vol. 12(3), pp. 33, 2018.

[8] J.H. Martin, and D. Jurafsky, "Speech and language processing", International Edition, vol. 710: pp. 25, 2000.

[9] B. Hussain, et al., "An evaluation of clustering algorithms in duplicate detection," Technical Report CSRG-620, University of Toronto, Department of Computer Science, 2013.

[10] M.A. Hernández, and S.J. Stolfo, "Real-world data is dirty: Data cleansing and the merge/purge problem", Data mining and knowledge discovery, vol. 2(1), pp. 9-37, 1998.

[11] L. He, et al, "An efficient data cleaning algorithm based on attributes selection", in Computer Sciences and Convergence Information Technology (ICCIT), 2011 6th International Conference on. 2011. IEEE.

[12] T. Smith, and M. Waterman, ªIdentification of Common Molecular Subsequences. º J. Molecular Biology, vol. 147, pp. 195-197, 1981.

[13] Li, M., Q. Xie, and Q. Ding, An Improved Data Cleaning Algorithm Based on SNM, in Cloud Computing and Security. 2015, Springer. p. 259-269.

[14] T. Wang, et al, "SIER: An Efficient Entity Resolution Mechanism Combining SNM and Iteration". in Web Information System and Application Conference (WISA), 2014 11th. 2014. IEEE.

[15] L. Alami, I. Hafidi, and A. Metrane, "Entity Resolution in NoSQL Data Warehouse", in International Conference on Information Technology and Communication Systems. 2017. Springer.

[16] M. Bilenko, and R.J. Mooney, "Adaptive duplicate detection using learnable string similarity measures". in Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. 2003. ACM.

[17] B. Kenig, and A. Gal, "MFIBlocks: An effective blocking algorithm for entity resolution", Information Systems, vol. 38(6), pp. 908-926, 2013.

[18] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases", in Acm sigmod record. 1993. ACM.

[19] S. Chaudhuri, V. Ganti, and R. Motwani, "Robust identification of fuzzy duplicates. in Data Engineering," 2005 .ICDE 2005. Proceedings. 21st International Conference on. 2005. IEEE.

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]

[DOI: 10.52547/jsdp.18.4.3]

[20] A. Saeedi, E. Peukert, and E. Rahm, "Comparative evaluation of distributed clustering schemes for multi-source entity resolution", in Advances in Databases and Information Systems, 2017, Springer.

[21] P. Christen, "Data matching: concepts and techniques for record linkage", entity resolution, and duplicate detection. 2012: Springer Science & Business Media.

[22] T. Papenbrock, , A. Heise, and F. Naumann, "Progressive duplicate detection," IEEE Transactions on knowledge and data engineering,vol. 27(5), p p. 1316-1329, 2015.

[23] S.E. Whang, D. Marmaros, and H. Garcia-Molina, "Pay-as-you-go entity resolution", IEEE Transactions on Knowledge and Data Engineering, vol.25(5), pp. 1111-1124, 2013.

[24] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality", in Proceedings of the thirtieth annual ACM symposium on Theory of computing, 1998, ACM.

[25] I. Van Dam, et al, "Duplicate detection in web shops using LSH to reduce the number of computations", in Proceedings of the 31st Annual ACM Symposium on Applied Computing, 2016, ACM.

[26] R. van Bezu, et al, "Multi-component similarity method for web product duplicate detection," in Proceedings of the 30th annual ACM symposium on applied computing, 2015, ACM.

[27] D. Vatsalan, and P. Christen, "Sorted nearest neighborhood clustering for efficient private blocking", in Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2013, Springer.

[28] Z. Yuhang, W. Yue, and Y. Wei, "Research on Data Cleaning in Text Clustering", in Information Technology and Applications (IFITA), 2010 International Forum on, 2010, IEEE.

[29] S. Thampi, and D. Loganathan, Progressive of Duplicate Detection Using Adaptive Window Technique.

[30] M. Dash, and H. Liu, "Feature selection for classification. Intelligent data analysis",vol. 1(1-4), pp. 131-156. 1997.

[31] M. Mohri, A. Rostamizadeh, and A. Talwalkar, Foundations of machine learning. 2012, MIT press.

[32] P.E. Greenwood, and M.S. Nikulin, "A guide to chi-squared testing," Vol. 280, 1996.

[33] https://www13.hpi.uni-potsdam.de/fileadmin/user_upload/fachgebiete/naumann/projekte/dude/restaurant.csv.

[34] https://www13.hpi.uni-potsdam.de/fileadmin/user_upload/fachgebiete/naumann/projekte/dude/cd.csv.

[35] https://www13.hpi.uni-potsdam.de/fileadmin/user_upload/fachgebiete/naumann/projekte/dude/CORA.xml.

[36] https://data.wa.gov/api/views/y3ds-rkew/rows. csv?accessType=DOWNLOAD.

[37] J.C. Dunn, Well-separated clusters and optimal fuzzy partitions. Journal of cybernetics, 1974. 4(1): p. 95-104.

[38] https://www.statisticshowtodatascience central .com/probability-and-statistics/t-test./

[39] M. keyvanpour, "A Divisive Hierarchical Clustering-based Method for Indexing Image Information" , JSDP, vol. 11 (2), pp. 91-109, 2015.

[۳۹] ایزدپناه نجوا، کیوان پور محمدرضا، رنجبران سعیده. یک روش مبتنی بر خوشه‌بندی سلسله‌مراتبی تقسیم‌کننده جهت شاخص‌گذاری اطلاعات تصویری . پردازش علائم و داده‌ها. ۱۳۹۳؛ ۱۱ (۲): ۹۱-۱۰۹

علی برزگری مدرک کارشناسی خود را در رشته مهندسی کامپیوتر-نرم‌افزار در سال ۱۳۹۳ از دانشگاه تربیت دبیر شهید رجایی و در سال ۱۳۹۷ مدرک کارشناسی ارشد خود را از دانشگاه آزاد واحد علوم و تحقیقات تهران دریافت کرده است. زمینه‌های پژوهشی مورد علاقه ایشان عبارتند از: پایگاه داده تحلیلی، داده‌کاوی و پیش‌پردازش داده.
نشانی رایانامه ایشان عبارت است از:
Ali.barzegari70@gmail.com

نگین دانشپور مدرک کارشناسی خود را در سال ۱۳۷۷ از دانشگاه شهید بهشتی و درجه کارشناسی ارشد و دکترای خود را در رشته مهندسی کامپیوتر-نرم‌افزار از دانشگاه صنعتی امیرکبیر در سال‌های ۱۳۸۰ و ۱۳۸۹ دریافت کرده است. در حال حاضر وی عضو هیأت علمی و دانشیار دانشکده مهندسی کامپیوتر دانشگاه تربیت دبیر شهید رجایی است. زمینه‌های پژوهشی مورد علاقه ایشان عبارتند از: پایگاه داده تحلیلی، سیستم‌های تصمیم‌یار، داده‌کاوی و پیش‌پردازش داده.
نشانی رایانامه ایشان عبارت است از:
ndaneshpour@sru.ac.ir

[Downloaded from jsdp.rcisp.ac.ir on 2026-11-11]

[DOI: 10.52547/jsdp.18.4.3]