یک الگوریتم خوشه‌بندی سلسله مراتبی ترکیبی برپایه روش مبتنی بر تراکم

علیرضا لطیفی پاکدهی[۱]، نگین دانشپور[۲*]
[۱]دانشجوی کارشناسی ارشد، دانشگاه تربیت دبیر شهید رجایی
[۲]استادیار دانشکده مهندسی کامپیوتر ، دانشگاه تربیت دبیر شهید رجایی، ndaneshpour@srttu.edu

چکیده

خوشه‌بندی یکی از شاخه‌های مهم موجود در داده‌کاوی است که هدف آن تقسیم داده‌ها به زیرمجموعه‌های معناداری است که خوشه نامیده می‌شوند. این تکنیک شامل فرآیند پیدا کردن گروه‌بندی طبیعی در مجموعه داده‌ها، بر اساس شباهت و تفاوت است به نحوی که اطلاعات قبلی کمی در مورد داده‌ها در دسترس است و یا اصلا اطلاعاتی در دسترس نیست. در طی دهه‌های متمادی الگوریتم‌های فراوانی برای خوشه‌بندی در رویکردهای مختلف و متفاوت و یا ترکیبی از آنها ایجاد شده‌اند. در این مقاله الگوریتمی بر پایه رویکردهای مبنی بر تراکم و سلسله مراتبی ارائه می‌شود. DBSCAN یکی از الگوریتم‌های مطرح شده در رویکرد مبتنی بر تراکم است. این الگوریتم نیاز به دو پارامتر دارد که تعیین آن هنوز یک چالش بزرگ است. در روش پیشنهادی پارامترهای الگوریتم DBSCAN طوری تنظیم می‌شود که بدون نیاز به دخالت کاربر، خوشه‌های احتمالی بصورت خودکار یافت شوند. سپس خوشه‌های نزدیک به یکدیگر به قدری باهم ادغام می‌شوند تا کیفیت خوشه‌های نهایی به نحو مطلوبی ارتقا یابد. بدین ترتیب خوشه‌های باکیفیت و دقیقی بدست خواهد آمد. در انتها برای آزمایش این الگوریتم ترکیبی جدید از داده‌های واقعی موجود در پایگاه داده UCI استفاده شد. نتایج نشان می‌دهد که الگوریتم ترکیبی جدید کارایی بیشتر و دقیق‌تر و سرعت مناسبی نسبت به روش‌های قبلی دارد.

کلیدواژه

داده‌کاوی، خوشه‌بندی ترکیبی، خوشه بندی سلسله مراتبی، خوشه بندی مبتنی بر تراکم

مقدمه

داده‌کاوی فرآیند استخراج اطلاعات مفید از یک مجموعه داده است [۱]. هدف خوشه‌بندی تقسیم مجموعه‌ی داده‌ها[۱] به زیر مجموعه‌هایی از آن (خوشه‌ها) است، بطوریکه اشیا موجود در یک خوشه به یکدیگر شبیه‌اند و نسبت به سایر خوشه‌ها متفاوت هستند [۲]. الگوریتم‌های خوشه‌بندی برای تقسیم-بندی مجموعه داده‌ها اغلب از یک معیار فاصله (مثلا اقلیدسی) برای معیار شباهت بهره می‌گیرند. در نتیجه نقاط موجود در هر خوشه به یکدیگر بیشتر از نقاط موجود در خوشه‌های دیگر شباهت دارند.

این تکنیک یکی از روش‌های بدون نظارت جستجو و تحلیل داده‌هاست که در رشته‌های متفاوتی چون آمار، یادگیری ماشین، داده‌کاوی، شناسایی الگو و بیوانفورماتیک استفاده شده است [۶-۳]. هم‌چنین در بسیاری از زمینه‌ها مثل ستاره-شناسی، فیزیک، داروسازی و بازاریابی کاربرد دارد[۷].

خوشه‌بندی بطور کلی به چهار دسته تقسیم می‌شود: روش‌های تقسیم‌بندی، روش‌های سلسله مراتبی، روش‌های مبتنی بر تراکم و روش مبتنی بر گرید [۸]. در ادامه به[۵]توضیح دو مورد پرکاربردترین این روش‌ها که در این مقاله نیز مورد استفاده قرار گرفته‌اند، پرداخته می‌شود.

روش سلسله مراتبی: به دو دسته پایین به بالا و بالا به پایین تقسیم می‌شود. در روش پایین به بالا، ابتدا هر شیی به عنوان خوشه‌ای مجزا در نظر گرفته می‌شود و سپس خوشه‌هایی که شباهت بیشتری با یکدیگر دارند ترکیب می‌شوند و یک خوشه جدید به‌وجود می‌آورند. این فرآیند ادامه می‌یابد تا در نهایت تعداد خوشه‌ها به مقدار مورد نظر برسد. روش بالا به پایین برعکس این روش می‌باشد؛ یعنی ابتدا کل داده‌ها یک خوشه در

[۱] Data Sets
[۲] Partitioning method
[۳] hierarchical method
[۴] Density based method
[۵] Grid based method

تاریخ دریافت: ۱۳۹۶/۰۶/۲۳ تاریخ پذیرش: ۱۳۹۷/۰۷/۱۵

نظر گرفته می‌شوند و آنقدر خوشه‌ها تقسیم می‌شوند که تعداد خوشه‌ها به تعداد مورد نظر برسد[۷].

روش مبتنی بر تراکم: این روش‌های خوشه‌بندی بر این اصل استوارند که خوشه‌ها، ناحیه‌هایی از فضای داده با چگالی زیادی هستند که توسط نواحی با چگالی کمتر از یکدیگر جدا شده‌اند. تراکم یک شی داده می‌تواند بوسیله تعداد اشیای نزدیک به آن شیی معین شود. مزیت عمده روش خوشه‌بندی بر پایه تراکم این است که می‌تواند خوشه‌هایی با شکل غیر دایره‌ای را پیدا کند [۸]. از شناخته شده ترین و مهمترین الگوریتم‌های مبتنی بر تراکم، الگوریتم DBSCAN[۱] است.

الگوریتم DBSCAN ابتدا شیی P را بصورت تصادفی انتخاب می‌کند. اگر در شعاعی به اندازه Eps، حداقل به مقدار minPts عدد شیی وجود داشت، یک خوشه جدید ایجاد می‌شود و آن نقاط درون آن قرار می‌گیرند (minPts و Eps پارامترهایی هستند که مقدار آن‌ها می‌بایست توسط کاربر معین شود). سپس کلیه نقاط درون خوشه نیز بر اساس دو پارامتر فوق ارزیابی می‌شوند تا سایر نقاطی که در مرحله قبل بررسی نشده بودند، تا حد امکان به خوشه اضافه گردند. روال فوق تا جایی که کلیه نقاط بررسی شوند ادامه می‌یابد [۹].

این الگوریتم، خوشه‌های با اندازه و شکل‌های دلخواه و متفاوت را از یک مجموعه داده بزرگ استخراج می‌کند [۱۰]. هم‌چنین این الگوریتم بخصوص در زمانی که تعداد داده‌ها زیاد نباشد، سرعت خوبی دارد [۱۱]. اما همانطوریکه گفته شد این الگوریتم دو پارامتر ورودی به نام‌های minPts و Eps دارد. اولین پارامتر آستانه ناحیه با تراکم بالاست. دومین پارامتر نیز شعاع همسایگی می‌باشد. هر دوی این پارامترها به نحوه توزیع داده‌ها و پراکندگی آنها مربوط است و تعیین آنها بسیار مشکل است.

در این مقاله ابتدا روشی ارائه می‌شود تا بدون نیاز کاربر پارامترها تعیین شوند. پارامتر Eps همانطور که گفته شد، شعاع همسایگی را مشخص می‌کند. به عبارت دیگر برای تشخیص اینکه یک ناحیه تراکم بالایی دارد یا خیر، می‌بایست در اطراف هر شیی به اندازه شعاعی که مقداری برابر Eps دارد یک مقدار حداقلی داده وجود داشته باشد. در اغلب داده‌ها تعیین یک مقدار برای Eps، منجر به نتایج خوبی نمی‌شود. بنابراین در چنین مواردی، باید چندین مقدار برای آن درنظر گرفت. در این مقاله، برای تعیین Eps یک روش مبتنی بر گراف k-dist ارائه می‌شود. این گراف در حقیقت یک نمودار است که به ازای هر شیی داده، فاصله آن داده تا k امین نزدیکترین داده به آن را مشخص می‌کند. در روش‌های موجود این k می‌بایست توسط کاربر تعیین شود که تشخیص آن بسیار مشکل است. در قسمت روش پیشنهادی نشان داده می‌شود که بین این k و پارامتر minPts در الگوریتم DBSCAN یک ارتباط تشابه بسیار قوی وجود دارد. پس از تعیین پارامترها به ازای هر مقدار Eps یکبار الگوریتم DBSCAN اجرا می‌شود. سپس خوشه‌های بدست آمده به روش خوشه‌بندی سلسله مراتبی بالا به پایین ادغام می‌شوند تا تعداد خوشه‌ها به حد مطلوب کاربر برسد.

از طرفی روش پیشنهادی را می‌توان نوعی الگوریتم سلسله مراتبی پایین به بالا در نظر گرفت که خوشه‌بندی در سطوح پایین آن را یک الگوریتم مبتنی بر تراکم انجام می‌دهد. این کار باعث می‌شود تا سرعت این الگوریتم‌ها بطور چشمگیری افزایش یابد؛ چرا که کم بودن سرعت الگوریتم های سلسله مراتبی مربوط به سطوح پایین سلسله مراتب است.

بقیه مطالب مقاله به اینصورت است: ابتدا در بخش پیشینه پژوهش کارهای مشابه تشریح شده است. در بخش بعدی به تبیین الگوریتم‌های مورد استفاده در مقاله و در ادامه به بیان روش پیشنهادی و چارچوب کلی راه حل پرداخته شده است. در بخش آزمایشات نتایج راه حل پیشنهادی روی داده‌های واقعی برگرفته از UCI نشان داده شده است. نهایتا نتیجه گیری در بخش آخر ذکر شده است.

پیشینه پژوهش

الگوریتم DBSCAN، اولین و معروفترین روش مبتنی بر تراکم است که سایر روش‌ها یا از این روش الهام گرفته‌اند، یا این الگوریتم را بهبود داده‌اند؛ بهبود سرعت [۱۲]، دقت [۱۱] و تلاش برای خود مختار نمودن این الگوریتم به نحوی که دو پارامتر ورودی این الگوریتم (minPts و Eps) حذف گردند. مقدار این دو پارامتر می‌تواند در یک دامنه وسیع قرار گیرد و بنابراین عدم دقت در انتخاب آن ممکن است نتایج ناخوشایندی به بار آورد و علی‌رغم همه‌ی محاسن این الگوریتم در بخش‌های مختلف، عملکرد آن‌را زیر سوال ببرد. برای حل مشکل پارامترها تا کنون روش‌هایی برپایه گرید و گراف k-dist و غیره ایجاد شده‌اند که در ادامه به آنها پرداخته خواهد شد.

الگوریتم GRIDBSCAN [۱۳] یک راه حل سه مرحله‌ای برای خوشه‌بندی ارائه می‌کند. در مرحله اول یک گرید ایجاد می‌کند که تراکم سلول‌های آن مشابه باشد. این الگوریتم با استفاده از

[۱] Density-Based Spatial Clustering of Application with Noise

پارامتر ورودی λ هر بعد داده را به λ قسمت تقسیم می‌کند. در مرحله دوم سلول‌های با تراکم یکسان را ادغام می‌کند. در این مرحله مقادیر مناسب minPts و Eps برای هر سلول بدست می‌آید. در این مرحله، این الگوریتم از یک پارامتر دیگری بنام perc استفاده می‌کند تا دو سلول قابل دسترس را شناسایی کند. در مرحله سوم با استفاده از پارامترهای بدست آمده از مرحله قبل، الگوریتم DBSCAN بکار گرفته می‌شود تا نتیجه‌ی نهایی حاصل شود. این الگوریتم دقت بیشتری نسبت به DBSCAN دارد ولی پیچیدگی بیشتری نسبت به آن دارد.

الگوریتم GRPDBSCAN [۱۴] نیز ابتدا فضای حالت را مشابه الگوریتم‌های مبتنی بر گرید به سلول‌هایی تقسیم‌بندی می‌کند. این الگوریتم بصورت خودکار پارامترهای minPts و Eps را استخراج می‌کند. در این روش بر اساس مقدار این پارامترها سه نوع گرید خواهیم داشت: گرید مرکزی، گرید حاشیه‌ای و گرید نویز. همانند الگوریتم DBSCAN، هر گریدی که بیشتر از آستانه عضو داشته باشد، گرید مرکزی است. اگر همسایه گرید مرکزی نیز خود مرکزی باشد بهم متصل می‌شوند. بنابراین از اتصال آنها چندین گراف جهتدار ایجاد می‌شود که همان خوشه‌های نهایی می‌باشند. در این روش نیز اگرچه پارامترهای minPts و Eps بصورت خودکار کشف می‌شوند ولی همانند سایر روش‌های مشابه خود نیازمند پارامتر اولیه برای ایجاد گرید است.

در [۱۵] بهبودی برای الگوریتم DBSCAN ارائه شده که در آن پارامتر Eps حذف شده و جای آن به پارامتر دیگری به نام ρ داده شده است. این پارامتر نسبت وجود نویز در مجموعه داده می‌باشد. این کار باعث کاهش پارامترهای الگوریتم اصلی نمی‌شود اما حدس آن توسط کاربر بسیار ساده‌تر از حدس پارامتر Eps می‌باشد.

در سال ۲۰۱۳ یک نسخه جدید از DBSCAN ایجاد شد که DMDBSCAN [۱۶] نام دارد. این تکنیک این فرضیه را در نظر می‌گیرد که ممکن است در مجموعه داده سطوح مختلفی از تراکم وجود داشته باشد. در نتیجه ممکن است در این مجموعه داده‌ها بیش از یک Eps وجود داشته باشد. این تکنیک بصورت خودکار Eps های موجود در هر سطح از مجموعه داده را شناسایی می‌کند. برای این منظور از گراف k-dist استفاده می‌کند. این گراف فاصله هر نقطه با k امین همسایه را نشان می‌دهد. و این k می‌بایست توسط کاربر تعیین شود. از محاسن این روش این است که خوشه‌ها به آسانی تفسیر می‌شوند و هیچ محدودیتی در شکل خوشه ایجاد شده وجود ندارد. عیب مهم این روش تعیین k است. این روش به ازای برخی مقادیر k هیچ نتیجه‌ای تولید نمی‌کند.

مقاله [۱۷] نیز از گراف k-dist برای یافتن Eps ها استفاده می‌کند. در این روش میانگین فاصله هر نقطه نسبت به k عدد نزدیکترین نقطه مبنای ترسیم گراف می‌شود. سپس شیب خط در فواصل منظم روی این گراف محاسبه می‌شود. نقاطی که به یک باره افزایش شیب قابل توجهی دارند، مبنای محاسبه Eps قرار می‌گیرند. برای اینکه مشخص شود یک نقطه افزایش قابل توجه شیب دارد یا خیر، از یک آستانه استفاده می‌کند. این روش minPts را نیز بصورت خودکار محاسبه می‌کند. به ازای هر Eps، میانگین مجموع نقاطی که در همسایگی هر نقطه قرار دارد minPts را مشخص می‌کند. این روش نیز مانند روش قبلی نیازمند تعیین k توسط کاربر است.

روش موجود در [۱۸]، در واقع یک بهبود و تلاش مجدد نویسنده مقاله قبلی برای محاسبه بهتر پارامترهاست. این روش نیز از گراف k-dist برای یافتن Eps ها استفاده می‌کند و همانند روش قبلی پارامتر minPts را محاسبه می‌کند. در این روش ایراد آستانه نقاط با شیب بالا همچنان باقی است. اما برخلاف روش قبل نیازی به تعیین k توسط کاربر نیست.

مقاله [۱۰] نیز با ارائه روشی سعی می‌کند مقداری برای Eps بیابد. نویسنده این روش نیز معتقد است که ممکن است در مجموعه داده‌ها بیش از یک مقدار برای Eps وجود داشته باشد. این روش با استفاده از مفاهیم k نزدیکترین همسایه و آماری سعی می‌کند تا بعدهای‌کم اهمیت مجموعه داده را از بین ببرد؛ بنابراین این روش نیازمند پارامتر ورودی k می‌باشد. همچنین این مقاله برای پیدا کردن پارامتر minPts روشی ارائه نمی‌کند.

الگوریتم BDE-DBSCAN [۱۹] روشی را براساس الگوریتم‌های تکاملی ارائه می‌کند تا بتواند مقدار پارامترهای Eps و minPts را محاسبه نماید. الگوریتم تکاملی مورد استفاده در این روش، الگوریتم تکامل تفاضلی است. از آنجایی[۷] که انتخاب نادرست پارامتر Eps می‌تواند عملکرد الگوریتم DBSCAN را به‌شدت تحت تاثیر قرار دهد، از روش‌های TS[۸] نیز استفاده شده است.

روش‌هایی که در این قسمت به آن‌ها اشاره شد، سعی نموده‌اند راه‌حلی برای بدست آوردن پارامترهای ورودی DBSCAN ارائه کنند. اما اکثر این الگوریتم‌ها به جای پارمترهای الگوریتم DBSCAN، نیازمند پارامترهای ورودی دیگری هستند و در اغلب موارد، تخمین مقدار پارامتر جدید از تخمین مقدار پارامترهای الگوریتم DBSCAN سخت‌تر است. در این مقاله با ارائه یک روش مبتنی بر گراف k-dist، نیاز به تعیین پارامترهای ورودی توسط کاربر، رفع می‌شود. هم‌چنین این روش ترکیبی باعث می‌شود تا ضمن حفظ ویژگی‌های کلیدی الگوریتم سلسله مراتبی، سرعت آن نیز افزایش یابد.

---

۷ Differential Evolution
۸ Tournament Selection

الگوریتم‌های پایه

در این بخش به توضیح الگوریتم‌های به کار رفته در الگوریتم ترکیبی پرداخته می‌شود.

خوشه‌بندی مبتنی برتراکم و مفاهیم الگوریتم DBSCAN

الگوریتم DBSCAN [۹] یکی از الگوریتم‌های خوشه‌بندی مبتنی بر تراکم می‌باشد. الگوریتم‌های خوشه‌بندی مبتنی بر تراکم بر این اصل استوارند که خوشه‌ها، ناحیه‌هایی از فضای داده با تراکم بالا هستند که توسط نواحی با تراکم کمتر جدا شده‌اند.

الگوریتم DBSCAN برای پیدا کردن ناحیه‌های با تراکم بالا نیاز به دو پارامتر دارد که مقدار آن‌ها توسط کاربر تعیین می‌شود. یکی از پارامترها که Eps نام دارد، شعاع همسایگی هر شیی را مشخص می‌کند. Eps-همسایگی شیی P، فضایی داخل شعاع Eps و به مرکزیت شیی P است. همچنین برای اینکه مشخص شود همسایگی یک شیی تراکم بالایی دارد یا خیر، DBSCAN از یک پارامتر دیگری به نام minPts استفاده می‌کند. در واقع این پارامتر، آستانه[۹] یک ناحیه با تراکم بالا را تعیین می‌کند.

قبل از تشریح الگوریتم DBSCAN ابتدا لازم است چند تعریف مهم بیان شوند.

با در نظر گرفتن تعداد اشیای موجود در همسایگی، سه نوع شیی را می‌توان شناسایی کرد: اشیای مرکزی؛[۱۰] اشیای حاشیه‌ای[۱۱] و اشیای نویز.[۱۲] اگر Eps-همسایگی شیی p حداقل minPts شیی داشته باشد، آن شیی، شیی مرکزی نامیده می‌شود، در غیر اینصورت، اگر نویز نباشد، شیی حاشیه‌ای است [۱۴].

هم‌چنین برای هر شیی مرکزی q و شیی p، می‌توان گفت که p از q ، directly-density-reachable است، اگر p داخل شعاع همسایگی q باشد. می‌توان گفت که p از q، density-reachable است اگر زنجیره‌ای از اشیای p1,..., pn وجود داشته باشد، به قسمی که $p_1=q$ و $p_n=p$ و p_{i+1} به ازای هر i بین ۱ تا n ، از p_i، directly-density-reachable باشد.

الگوریتم DBSCAN

در الگوریتم BDSCAN ابتدا همه اشیای درون مجموعه داده D به عنوان ملاقات‌نشده[۱۳] نشانه‌گذاری می‌شود. این الگوریتم به شکل تصادفی، شیی ملاقات‌نشده P را انتخاب می‌کند و آنرا به عنوان ملاقات‌شده[۱۴] نشانه‌گذاری می‌کند. سپس بررسی می‌کند که Eps-همسایگی شیی P چند شیی دارد. اگر حداقل تعداد MinPts شیی را داشته باشد، خوشه C ایجاد می‌شود و P درون آن قرار می‌گیرد و Eps-همسایگی شیی P درون مجموعه NeighborPts قرار می‌گیرد. اگر Eps-همسایگی شیی P به اندازه تعداد MinPts شیی نداشته باشد، P به عنوان نویز در نظر گرفته می‌شود. شکل ۱ شبه کد این روال را نشان می‌دهد.

```
DBSCAN Algorithm.
Input: Data set D , Eps , MinPts
Output: clustering Set.
Begin
C = 0
for each point P in dataset D
    if P is visited
        continue next point
    mark P as visited
    NeighborPts = getNeighbor()
    if sizeof(NeighborPts) < MinPts
        mark P as NOISE
    else
        C = next cluster
        expandCluster(P, NeighborPts, C, Eps,
    MinPts)
  end for
End
```

شکل ۱. الگوریتم DBSCAN

ادامه این الگوریتم که فاز توسعه[۱۵] نامیده می‌شود توسط روال expandCluster انجام می‌شود. این روال اشیایی از NeighborPts که به هیچ خوشه‌ای تعلق ندارند را به C اضافه می‌کند. در این فرآیند، به ازای هر شیی 'P موجود در NeighborPts که برچسب ملاقات‌نشده را برخود دارد، DBSCAN آنرا ملاقات‌شده می‌کند و Eps-همسایه آنرا بررسی می‌کند. اگر Eps-همسایه 'P حداقل MinPts شیی داشت، اشیای Eps-همسایه مربوط به 'P به NeighborPts اضافه می‌شوند. این الگوریتم فرآیند اضافه کردن به C را تا جایی ادامه می‌دهد که C دیگر نتواند بزرگتر شود، یعنی NeighborPts خالی شود. شکل ۲ شبه کد روال expandCluster را نشان می‌دهد.

---

۹ threshold
۱۰ Core object
۱۱ Border object
۱۲ Noise object
۱۳ unvisited
۱۴ visited
۱۵ expansion

```
expandCluster()
Input: P, NeighborPts, C, Eps, MinPts
Output: results.
Begin
add P to cluster C
for each point P' in NeighborPts
    if P' is not visited
        mark P' as visited
        NeighborPts' = getNeighbor()
        if sizeof(NeighborPts') >= MinPts
        NeighborPts = NeighborPts
        joined with NeighborPts'
    end if
    if P' is not yet member of any cluster
        add P' to cluster C
  end for
End
```

شکل ۲. فاز توسعه الگوریتم DBSCAN

این الگوریتم برای پیدا کردن خوشه بعدی، به شکل تصادفی یک شیی ملاقات‌نشده را از اشیای باقی‌مانده انتخاب می‌کند و مراحل فوق تکرار می‌گردد. فرآیند خوشه‌بندی تا جایی که همه اشیا ملاقات شوند، ادامه می‌یابد.

الگوریتم Revised DBSCAN

بر خلاف ویژگی‌های مناسبی که الگوریتم اصلی BDSCAN دارد، زمانی که اشیای حاشیه‌ای خوشه‌ها نسبتا به یکدیگر نزدیک باشند، این الگوریتم موفق نخواهد بود [۹]. همچنین نتایج خوشه‌بندی به ترتیبی که داده‌ها در فاز توسعه پردازش می‌شوند، بستگی دارد [۱۱]. بنابراین الگوریتم اصلی، الگوریتم قوی‌ای نیست. هدف اصلی الگوریتم بهبود یافته، حفظ ویژگی‌های اصلی این الگوریتم در کنار افزایش دقت الگوریتم در نسبت دادن اشیای حاشیه‌ای به خوشه‌هاست.

در بدنه الگوریتم بهبود یافته، به جای زنجیره density-reachable از زنجیره core-density-reachable استفاده می‌شود. در زنجیره density-reachable اشیای X1,…Xn می‌توانند بصورت $[X_{core_1},...,X_{core_{n-1}},X_{core_n}]$ باشند که همگی اشیای مرکزی هستند یا بصورت $[X_{core_1},...,X_{core_{n-1}},X_{border_n}]$ باشند که فقط آخرین مورد آن شیی حاشیه‌ای است. در این زنجیره همانطور که از نام آن پیداست فقط اشیای مرکزی حضور دارند.

در این الگوریتم اشیای حاشیه‌ای که عمدتا در طی گام توسعه به خوشه‌ها تخصیص می‌یافتند، بطور موقت در خوشه‌بندی شرکت داده نمی‌شوند تا تمام نقاط مرکزی همه‌ی خوشه‌ها شناسایی شوند. این نقاط درون یک بردار نگهداری می‌شوند تا در آخرین مرحله به مناسب‌ترین خوشه اختصاص یابند.

در آخرین گام این الگوریتم، همه اشیای مرکزی به بهترین زنجیره density-reachable اختصاص یافته‌اند و فقط نقاط حاشیه‌ای باقی مانده‌اند. یک تصمیم منطقی این است که هر شیی حاشیه‌ای موجود در بردار به خوشه‌ای اختصاص یابد که نزدیکترین زنجیره core-density-reachable به آن تعلق دارد.

روش سلسله مراتبی

خوشه بندی سلسله مراتبی یک روش خوشه‌بندی شناخته شده‌ای است که کاربردهای فراوانی دارد. این روش شامل دو نوع کلی پایین به بالا یا بالا به پایین می‌شود. روش پایین به بالا دقیق‌تر است ولی نسبت به روش بالا به پایین پیچیدگی زمانی بیشتری دارد. با این حال، این افزایش پیچیدگی محاسباتی همزمان با افزایش پیچیدگی مفهومی و یا الگوریتمی نیست؛ زیرا فرآیند تشکیل سلسله مراتبی خوشه می‌تواند به شکل دنباله‌ای از ادغام خوشه‌های اولیه یا عملگرهای تقسیم-بندی سازماندهی شود [۲۳-۲۰].

در روش پایین به بالا، ابتدا هر داده به عنوان خوشه‌ای مجزا در نظر گرفته می‌شود. سپس طی فرایندی تکراری در هر مرحله، خوشه‌هایی که شباهت بیشتری با یکدیگر دارند ترکیب می‌شوند تا در نهایت یک خوشه و یا تعداد مشخصی خوشه حاصل شود [۸]. این الگوریتم برای N نمونه با N خوشه شروع می‌شود که هر خوشه شامل یک نمونه است. بعد از آن دو خوشه با نزدیک‌ترین شباهت ادغام می‌شوند تا اینکه تعداد خوشه‌ها به یک یا عددی که کاربر تعیین می‌کند برسد [۱۸, ۲۱, ۲۳, ۲۴]. شکل ۳ الگوریتم سلسله مراتبی پایین به بالا را نشان می‌دهد.

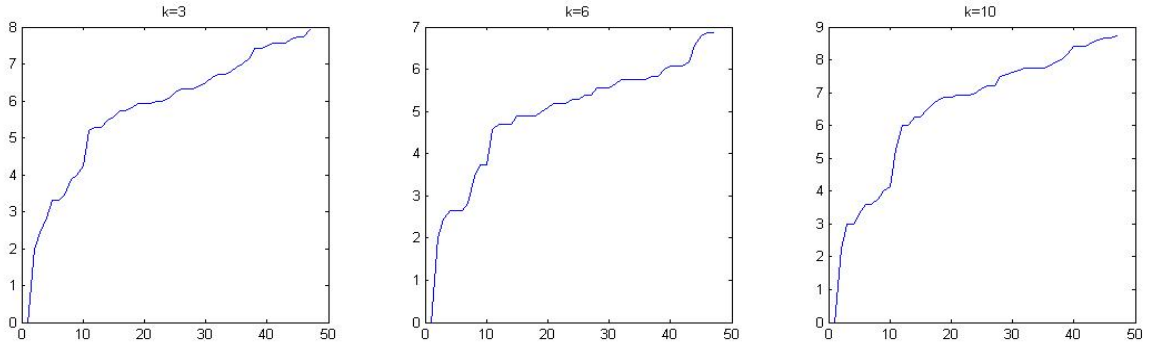
```
Bottom-up Hierarchical Algorithm.
Input: Data set D, Number of Clusters k
Output: Clusters.
Begin
   Initial Clustering, Put each data in a cluster
   While count of clusters greater than k
        Find two clusters Ci and Cj, in which
     Similarity(Ci,Cj)   is max.
     Merge Ci  and Cj
   end while
End
```

شکل ۳. الگوریتم سلسله مراتبی پایین به بالا

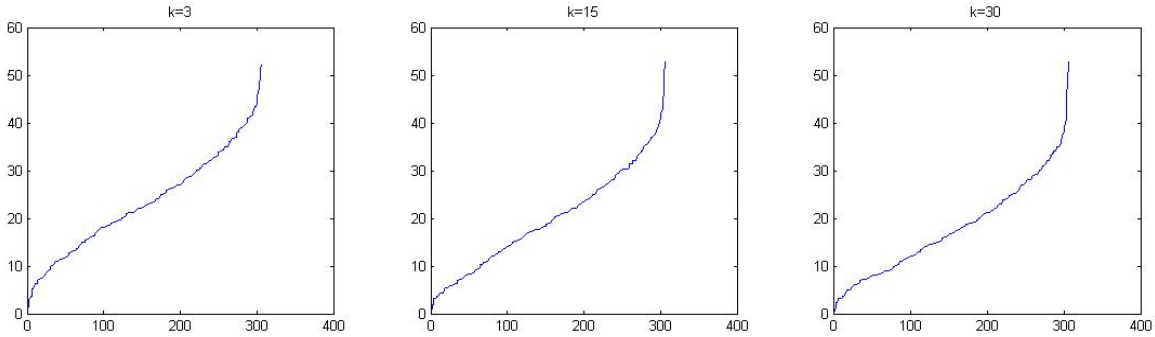
روش پیشنهادی

یکی از چالش‌های مهم الگوریتم DBSCAN، این است که این الگوریتم دو پارامتر ورودی Eps و minPts دارد و این پارامترها می‌بایست توسط کاربر مقداردهی شوند. این پارامترها به ترتیب مربوط به شعاع همسایگی و آستانه تعیین ناحیه با تراکم بالا هستند. مقدار این دو پارامتر به نحوه پراکندگی و فاصله داده‌ها وابسته است. بنابراین تعیین این پارامترها از ناحیه کاربر که شاید اطلاعی از این مفاهیم نداشته باشد بسیار مشکل است و تنها تعداد بسیار کمی الگوریتم باکیفیت برای تعیین این پارمترها وجود دارد[۱۰].

برای یافتن راهی که بتوان مقدار این پارامترها را تعیین کرد، مطالعه نحوه پراکندگی داده‌ها و فاصله‌ی بین آنها ضرروی است. یک ابزار مناسب برای این کار، گراف k-dist است. برای ترسیم این گراف، ابتدا فاصله بین هر نقطه تا k امین نزدیکترین همسایه‌ی آن نقطه محاسبه می‌شود. سپس این فواصل بدست آمده، بصورت صعودی مرتب می‌شوند. در نهایت یک گراف بر اساس این فواصل مرتب‌شده ترسیم می‌شود. شکل ۴ نمونه‌ای از گراف k-dist متعلق به داده واقعی Soybean را به ازای k های متفاوت نشان می‌دهد. شکل ۵ نمونه‌ای دیگر از آن‌را برای داده‌ی haberman نشان می‌دهد. ویژگی‌های این مجموعه داده‌ها در جدول ۱ واقع در بخش آزمایشات ذکر شده است. محور افقی نقاط مرتب شده بر اساس فاصله تا kامین نزدیکترین همسایه و محور عمودی، فاصله تا kامین نزدیکترین همسایه می‌باشد.



شکل ۴. گراف k-dist متعلق به داده واقعی Soybean به ازای k های متفاوت



شکل ۵. گراف k-dist متعلق به داده واقعی haberman به ازای k های متفاوت

همانطوریکه این گراف‌ها نشان می‌دهند، فاصله یک نقطه تا k امین نزدیکترین همسایه آن نه تنها برای همه نقاط یکسان نیست، بلکه تفاوت چشمگیری نیز دارد. بنابراین به وضوح مشاهده می‌شود که تعیین یک مقدار ثابت برای Eps اصلا منطقی نیست و به آرایه‌ای از Epsها نیاز است. در این‌صورت می‌بایست به‌ازای هر مقدار Eps یکبار الگوریتم DBSCAN اجرا شود. در هربار اجرای این الگوریتم تعدادی از نقاط خوشه‌بندی می‌شوند و تعدادی باقی می‌مانند تا با مقدار بعدی Eps در فرآیند خوشه‌بندی شرکت داده شوند.

Minpts، دومین پارامتر این الگوریتم است که آستانه تعیین ناحیه با تراکم بالا است و محدوده‌ای بین ۳ تا تعداد کل اشیا دارد؛ زیرا اگر مقدار این پارامتر ۱ در نظر گرفته شود، ممکن است خوشه‌ای با تراکم ۱ به وجود آید که کاملا غیر منطقی است. هم‌چنین اگر مقدار این پارامتر ۲ در نظر گرفته شود، نتیجه خوشه‌بندی الگوریتم DBSCAN مشابه نتیجه خوشه-

بندی الگوریتم‌های سلسله مراتبی خواهد شد [۲۵]. هر چه مقدار این پارامتر کوچکتر باشد، تعداد خوشه‌ها بیشتر می‌شود و هرچه مقدار این پارامتر بزرگ‌تر انتخاب شود، تعداد کمتری خوشه ایجاد می‌شود. در نهایت اگر مقدار این پارامتر برابر کل اشیا انتخاب شود، تنها یک خوشه ایجاد می‌شود که این هم غیر قابل قبول است.

در روش پیشنهادی ابتدا با استفاده از گراف k-dist آرایه‌ای از Eps ها تولید می‌شود. استفاده از آرایه‌ای از Eps ها و اجرای متعدد الگوریتم Revised DBSCAN ممکن است منجر به تولید تعداد زیادی خوشه شود. هم‌چنین در هر بار اجرای الگوریتم Revised DBSCAN، به جای اینکه با روش‌های پیچیده مقدار پارامتر minPts تعیین شود، بهتر است مقدار آن برابر با حداقل مقدار، یعنی ۳ در نظر گرفته شود؛ اما با انجام این کار نیز ممکن است خوشه‌های کوچک ولی با تعداد بیش از اندازه تولید شود. چنانچه مقدار این پارامترها باعث شد که تعداد خوشه‌ها زیاد شود، در فاز بعدی روش ترکیبی، الگوریتم سلسله مراتبی خوشه‌های کوچک را ادغام می‌کند تا تعداد خوشه‌ها به میزان مورد دلخواه کاربر برسد. خاتمه یافتن الگوریتم سلسه مراتبی نیاز به تعیین شرط اولیه دارد که اغلب تعداد خوشه‌هاست و از کاربر دریافت می‌گردد. این پارامتر برخلاف دو پارامتر DBSCAN به راحتی توسط کاربر قابل پیش‌بینی است.

بنابراین با الگوریتم پیشنهادی نیاز به تعیین پارامترهای الگوریتم DBSCAN توسط کاربر رفع شد. هم‌چنین با یک تغییر دید می‌توان الگوریتم پیشنهادی را نوعی الگوریتم سلسله مراتبی پایین به بالا نیز دانست که در آن سطوح اولیه خوشه‌بندی به الگوریتم DBSCAN سپرده شده است. مهمترین مزیت این کار این است که باعث می‌شود الگوریتم‌های سلسله‌مراتبی پایین به بالا افزایش سرعت چشمگیری داشته باشند. زیرا زمان بالای اجرای الگوریتم‌های سلسله مراتبی پایین به بالا مربوط به سطوح اولیه‌ی خوشه‌بندی است؛ به این دلیل که تعداد مقایسات در سطوح پایین سلسه مراتب بسیار زیاد است. به همین دلیل در الگوریتم پیشنهادی، خوشه بندی در سطوح پایین به الگوریتم Revised DBSCAN سپرده شد که از سرعت خوبی برخوردار است. به این ترتیب مشکل بالا بودن زمان اجرای الگوریتم سلسله مراتبی پایین به بالا نیز برطرف می‌شود. در بخش بعدی الگوریتم ترکیبی، با جزئیات بیشتر ارائه می‌شود.

الگوریتم پیشنهادی

در الگوریتم پیشنهادی در اولین گام می‌بایست آرایه‌ای از Epsها ایجاد شود تا به کمک این آرایه بتوان الگوریتم Revised DBSCAN را اجرا نمود. به این منظور مشابه روشی که در توضیح گراف k-dist بیان شد، ابتدا فاصله هر نقطه تا kامین نزدیکترین همسایه آن محاسبه می‌شود. سپس این فواصل بصورت صعودی مرتب می‌شود و درون آرایه x نگهداری می‌شود. تغییر و جهش یکباره نمودار می‌تواند به عنوان نقطه تغییر Eps در نظر گرفته شود. بنابراین می‌بایست در چنین محل‌هایی یک Eps در نظر گرفت. اما همانطور که در شکل‌های ۴ و ۵ مشاهده شد، ممکن است در مجموعه داده فاصله بین نقاط بازه‌ی وسیعی داشته باشد، اما نقطه‌ای که به یکباره جهش داشته باشد پیدا نشود. بنابراین در الگوریتم پیشنهادی، بدون در نظر گرفتن وجود نقاط زانو شکل، $\sqrt{m}$ عدد از عناصر آرایه x در فواصل منظم انتخاب می‌شود و به عنوان آرایه Eps در نظر گرفته می‌شود. شکل ۶ شبه کد مربوط به انتخاب آرایه‌ای از Epsها را نشان می‌دهد.

```
Eps Generation Algorithm.
Input: Data set D including m objects
Output: Array of Eps
Begin
   For i=1to number of point
      For j=1 to number of point
         dist(i,j)= distance between D(i) and D(j);
      End For
      MaxDist(i)=3 th minimum value from dist(i);
   End For
   sort MaxDist in  Ascending manner;
   sm= sqrt of m;
   j=sm;
   i=1;
   While j<m
      Eps(i)= MaxDist(j);
      j=j + sm;
      i=i + 1;
   End While
End
```

شکل ۶. روال مربوط به یافتن آرایه Eps ها

همانطوریکه در شبه کد نیز مشخص است، k برابر با ۳ در نظر گرفته شد. دلیل این انتخاب دقیقا مشابه با انتخاب عدد ۳ برای minPts است. زیرا بین k در بحث k-dist و minPts در الگوریتم DBSCAN تشابه بسیار زیادی وجود دارد. زیرا در هردو به دنبال kتا یا minPts تا عنصر در اطراف یک عنصر می‌گردیم. به طور مثال وقتی که گفته می‌شود که n امین نزدیکترین همسایه یک نقطه با آن نقطه فاصله‌ای برابر با ε دارد، این ε درواقع همان شعاع همسایگی است که می‌بایست

در نظر گرفته شود تا حداقل n عنصر در اطراف آن عنصر کشف شود. حداقل n عنصر اطراف یک عنصر همان تعریف minPts است. بنابراین می‌بایست هر مقداری که برای minPts در نظر گرفته می‌شود برای k نیز در نظر گرفته شود. به‌دلیل اینکه minPts برابر ۳ در نظر گرفته شد، مقدار k نیز برابر ۳ در نظر گرفته می‌شود. اما این ممکن است منجر به تولید خوشه‌های با تراکم کم ولی با تعداد زیاد گردد. در این‌صورت در مرحله بعدی روش پیشنهادی، خوشه‌بندی سلسله مراتبی با ادغام خوشه‌ها، تعداد خوشه‌ها را به عدد مورد نظر کاربر خواهد رساند..

پس از محاسبه آرایه‌ی Epsها، به تعداد عناصر موجود در این آرایه الگوریتم Revised DBSCAN اجرا می‌شود. این الگوریتم یک شیی P را بطور تصادفی انتخاب می‌کند. اگر Eps-همسایگی آن minPts تا عضو داشت، یک خوشه تشکیل می‌گردد و نقطه P داخل خوشه قرار می‌گیرد. در ضمن همسایه‌هایی که داخل شعاع Eps قرار دارند نیز درون یک مجموعه موقتی به‌نام Border_List قرار می‌گیرند. سپس کلیه نقاط بدست آمده در مرحله قبل دوباره مورد بررسی قرار می‌گیرند. هر عضوی از Eps-همسایگی که minPts تا عضو داشت نیز به خوشه اضافه می‌گردد و نقاط Eps-همسایگی آن به Border_List اضافه می‌شود و این عملیات به قدری ادامه می‌یابد تا کلیه نقاط بررسی شوند. در آخرین مرحله از این الگوریتم، کلیه نقاط درون Border_List بررسی می‌شوند تا به خوشه نزدیک‌ترین شیی مرکزی اضافه گردند.

در آخرین گام از الگوریتم ترکیبی، تعداد خوشه‌های به وجود آمده از مرحله قبل با تعداد مورد انتظار کاربر مقایسه می‌شود. با توجه به عملیات خوشه‌بندی ترکیبی تعداد خوشه‌های بدست آمده بیشتر از مقدار واقعی است. بنابراین نیاز است تا خوشه‌ها ادغام شوند. برای این منظور ابتدا مرکز هر خوشه محاسبه می‌شود. سپس فاصله دو به دوی همه‌ی خوشه‌ها محاسبه می‌شود و دو خوشه با کمترین فاصله با هم ادغام می‌شوند و یک خوشه‌ی جدید به وجود می‌آورند. الگوریتم‌های سلسله مراتبی نیازمند یک شرط برای خاتمه خوشه‌بندی هستند. در الگوریتم پیشنهادی شرط خاتمه برابر شدن تعداد خوشه‌ها با تعداد مورد نظر کاربر است. تا برقرار شدن این شرط روال ادغام دو نزدیکترین خوشه ادامه می‌یابد. تعداد خوشه‌های مورد انتظار کاربر که در الگوریتم پیشنهادی k نام‌گذاری شده است، بسیار آسان‌تر از minPts و Eps تخمین زده می‌شود. این پارامتر نه تنها عیب محسوب نمی‌شود، بلکه مزیت نیز به شمار می‌رود؛ زیرا چنانچه کاربر از سطح ادغام خوشه‌ها راضی نبود، می‌تواند عملیات ادغام را بدون صرف هزینه مجدد تا سطح مورد نظر خود ادامه دهد. شکل ۷ شبه کد مربوط به خوشه‌بندی ترکیبی است.

```
Combinational Algorithm.
Input: Data set D, Number of Clusters k, Array of Eps
Output: clusters
Begin
    For each value in Eps Array
        Execute Revised DBSCAN Algorithm
    End For
    Calculate the centroid of each cluster.
    Merge clusters until the count of clusters equals
    k.
End
```

شکل ۷. الگوریتم ترکیبی

روش پیشنهادی در این مقاله شامل دو الگوریتم است: الگوریتم مربوط به یافتن آرایه‌ای از Epsها (شکل ۶) و خود الگوریتم ترکیبی(شکل ۷)؛ بدین صورت که ابتدا الگوریتم مربوط به یافتن آرایه‌ای از Epsها اجرا شده و سپس خود الگوریتم ترکیبی اجرا می‌شود. بنابراین پیچیدگی محاسباتی روش پیشنهاد شده، برابر با مجموع پیچیدگی این دو الگوریتم است. پیچدگی الگوریتم مربوط به یافتن آرایه‌ای از Epsها به دلیل داشتن دو حلقه تو در تو، $O(n^2)$ خواهد بود. در خود الگوریتم ترکیبی نیز پرهزینه‌ترین عملیات، مربوط به حلقه For است که تعداد دفعات تکرار آن $\sqrt{n}$ می‌باشد و در هر بار تکرار، یکبار الگوریتم Revised DBSCAN اجرا می‌شود. از آنجایی که پیچیدگی الگوریتم Revised DBSCAN برابر $O(n\log n)$ می‌باشد، پیچیدگی الگوریتم ترکیبی برابر $O(\sqrt{n}.n\log n)$ است.

بنابراین پیچیدگی کل روش پیشنهادی ارائه شده برابر $O(\sqrt{n}.n\log n + n^2)$ است. از آنجایی که در این عبارت جمله n^2 بزرگتر است، پیچیدگی کل روش پیشنهادی $O(n^2)$ خواهد شد.

آزمایشات

در این بخش میزان کارایی الگوریتم ارائه شده مورد بررسی قرار می‌گیرد. کلیه آزمایشات در شرایطی یکسان و بر روی سیستم کامپیوتری با واحد پردازش مرکزی[۱۱] intel core i3 سری M370 چهار هسته‌ای با فرکانس ۲,۴ GHz و حافظه اصلی DDR3 با مقدار ظرفیت ۴ GB انجام شده است. همچنین سیستم عامل مورد استفاده، ویندوز ۷ نسخه Ultimate و کلیه کدها به زبان برنامه‌نویسی MATLAB نگاشته شده است. ابتدا لازم است ویژگی‌های مجموعه داده‌ها بیان شوند و سپس به بررسی سایر تنظیمات مورد نیاز و مقایسه کارایی پرداخته خواهد شد.

[۱۱] CPU

مجموعه داده‌ها

در بخش آزمایشات از داده‌های واقعی استفاده شده که همه‌ی آنها از سایت UCI [۲۶] گرفته شده‌اند. ویژگی‌های این مجموعه داده‌ها در جدول ۱ بیان شده است. از جمله‌ی ویژگی-های این داده‌ها، بعد بالای آنها می‌باشد. داده‌های با ابعاد بالا به دلیل برخی ویژگی‌هایشان نظیر پیچیدگی بیشتر، به معضل جدی در دنیای داده‌کاوی تبدیل شده‌اند و توجه زیادی را به خود جلب کرده‌اند.

جدول ۱. ویژگی های مجموعه داده ها

ردیف	مجموعه داده	تعداد خوشه‌ها	تعداد بعد	تعداد نمونه
۱	Soybean	۴	۳۵	۴۷
۲	Parkinsons	۲	۲۱	۱۹۷
۳	Ecoli	۸	۷	۳۳۶
۴	Haberman	۲	۳	۳۰۶
۵	zoo	۷	۱۷	۱۰۱
۶	Fertility	۲	۹	۱۰۰

کارایی روی مجموعه داده‌ها

در این قسمت الگوریتم ارائه شده در بخش قبل، روی مجموعه داده‌های معرفی شده آزمایش می‌شود. این مقایسات بر اساس دقت و زمان اجرا طراحی شده‌اند.

در این بخش برای مقایسه از الگوریتم‌های سلسله‌مراتبی پایین به بالا که در جداول بصورت AHC نامگذاری شده، k-means، Adaptive Method [۱۸] و G-DBSCAN [۱۲] استفاده شده است. زیرا AHC بطور مستقیم در روش پیشنهادی استفاده شده است. همچنین در مقایسات از الگوریتم k-means نیز استفاده شده است. یکی از مهمترین مزیت‌های الگوریتم‌های تفکیکی نظیر k-means این است که آن‌ها کیفیت خوشه‌بندی را به تدریج در یک فرآیند بهینه‌سازی بهبود می‌دهند [۲۷, ۲۸]. همچنین بسیاری از روش‌های جدید نیز روش پیشنهادی خود را با این دو الگوریتم مقایسه کرده‌اند [۲۷]. در این مقایسات از الگوریتم‌های Adaptive Method [۱۸] و G-DBSCAN [۱۲] نیز استفاده شده است. این دو الگوریتم از جمله روش‌های جدیدی هستند که برروی حل مشکل پارامترهای DBSCAN و سرعت آن تلاش‌هایی داشته‌اند. جدول ۲ مقایسه بر اساس دقت روش پیشنهادی را با این الگوریتم‌ها نشان می‌دهد.

در جداول مقایسات، الگوریتم پیشنهادی به‌صورت Combinational algorithm نام‌گذاری شده است. الگوریتم‌های ACH، k-means و Combinational algorithm به پارامتر k نیاز دارند. برای مقداردهی k از مقدار صحیح آن که در جدول ۱ آمده است استفاده شده است. الگوریتم G-DBSCAN نیز همچون الگوریتم DBSCAN، نیاز به دو پارامتر Eps و minPts دارد. مقادیری از این دو پارامتر که منجر به بالاترین دقت می-شود در آزمایشات مورد استفاده قرار گرفته و در ستون مربوط به الگوریتم G-DBSCAN در داخل پرانتز قید شده است. بر طبق جدول ۲ روش پیشنهادی دقت به مراتب بالاتری نسبت به سایر الگوریتم‌ها دارد.

جدول ۲. مقایسه دقت

ردیف	مجموعه داده	Combinational algorithm	AHC	k-means	Adaptive Method	G-DBSCAN
۱	Soybean	۸۹،۳۶۱۷	۸۰،۸۵۱۱	۷۲،۳۴۰۴	۶۰،۳۴۸۵	۷۸,۷۲۳۴ (Eps= ۰,۵ و minPts=۴)
۲	Parkinson	۷۴،۸۷۱۸	۷۲،۳۰۷۷	۷۳،۸۴۶۲	۵۹،۶۵۹	۷۱,۲۸۲۱ (Eps= ۰,۰۰۵ و minPts=۴۰)
۳	Ecoli	۷۷،۶۷۸۶	۷۲،۹۱۶۷	۵۹،۲۲۶۲	۶۳،۷۶۹۸	۴۲,۵۵۹۵ (Eps= ۰,۰۰۰۷ و minPts=۶)
۴	Haberman	۷۳،۵۲۹۴	۶۳،۳۹۸۷	۵۱،۹۶۰۸	۵۴،۸۷۶۶	۷۳,۲۰۲۶ (Eps= ۷ و minPts=۱۰)
۵	Zoo	۷۸،۲۱۷۸	۷۵،۲۴۷۵	۶۸،۳۱۶۸	۷۰،۶۵۴۵	۴۰ (Eps= ۱,۵۳ و minPts=۷)
۶	Fertility	۸۷	۶۶	۶۲،۲۰۲۴	۵۵،۸۷۶۷	۸۸ (Eps= ۰,۰۰۴ و minPts=۵)

البته افزایش دقت الگوریتم ترکیبی نسبت به سایر روش‌ها تاثیر زیادی روی میزان زمان اجرا نیز نگذاشته و الگوریتم پیشنهادی در بعضی موارد، مقدار ناچیزی زمان اجرای بیشتر نسبت به سایر الگوریتم‌ها به جز G-DBSCAN دارد. الگوریتم G-DBSCAN اگرچه از سرعت خوبی نسبت به الگوریتم ترکیبی پیشنهادی دارد، اما نتواسته است معضل دو پارامتر DBSCAN را مرتفع نماید. جدول ۳ مقایسه الگوریتم‌های فوق را بر اساس زمان اجرا نشان می‌دهد.

جدول ۳. مقایسه زمان اجرا برحسب ثانیه

ردیف	مجموعه داده	Combinational algorithm	AHC	k-means	Adaptive Method	G-DBSCAN
۱	Soybean	۰،۱۶۱۳۰۷	۰،۱۶۴۸۴۹	۰،۲۲۹۱۴۶	۰،۱۴۲۳۵۲	۰،۰۱۴۳۰۱
۲	Parkinson	۰،۲۶۲۸۶۴	۱۱،۱۳۳۳۹	۰،۲۳۹۸۱۶	۰،۶۸۳۰۱۲	۰،۰۲۴۲۰۳
۳	Ecoli	۰،۵۰۴۵۸۵	۵۷،۰۵۲۹	۰،۲۴۸۱	۱،۷۰۰۴۸۷	۰،۰۴۸۲۸۸
۴	Haberman	۰،۴۰۰۷۰۷	۴۳،۹۲۹۰۷	۰،۲۴۲۳۵۲	۱،۱۵۷۰۷۸	۰،۰۲۴۰۶
۵	Zoo	۰،۲۶۲۸۶۴	۱،۷۱۰۳۸	۰،۲۳۸۶۱۱	۰،۲۴۳۹۴۷	۰،۰۱۹۴۹۰
۶	Fertility	۰،۲۰۲۳۱۷	۱،۶۱۱۴۷۷	۰،۲۳۳۳۴۳	۰،۲۴۴۶۲۴	۰،۰۱۸۵

نتایج جدول ۳ نشان می‌دهد که روش ترکیبی علیرغم اینکه یک الگوریتم ترکیبی است ولی زمان اجرای بسیار بهتری نسبت به روش AHC دارد. دلیل سرعت بسیار مناسب روش پیشنهادی نسبت به AHC این است که زمان اجرای بالای الگوریتم‌های سلسله مراتبی پایین به بالا مربوط به سطوح اولیه‌ی خوشه‌بندی است و این خوشه‌بندی در سطوح پایین به الگوریتم Revised DBSCAN واگذار شد که از سرعت خوبی برخوردار است.

همانطوریکه آزمایشات نشان می‌دهد روش پیشنهادی دقت بالاتری نسبت به سایر الگوریتم‌های مطرح شده دارد و از لحاظ سرعت نیز با توجه به ترکیبی بودن الگوریتم، در مجموع عملکرد مناسبی نشان داده است.

نتیجه گیری

در این مقاله، یک الگوریتم خوشه‌بندی سلسله‌مراتبی بر پایه روش‌های مبتنی بر تراکم ارائه شد. الگوریتم مبتنی بر تراکم مورد استفاده در روش پیشنهادی، الگوریتم Revised DBSCAN است. برای مقدار دهی پارامتر Eps این الگوریتم در روش پیشنهادی، ابتدا برای Eps آرایه‌ای از مقادیر را تولید می‌کند و به تعداد عناصر آرایه، الگوریتم Revised DBSCAN اجرا می‌شود. در گام بعدی خوشه‌های اولیه بدست آمده در مرحله بعد ادغام می‌شوند تا تعداد خوشه‌ها به میزان مورد نظر کاربر برسد. این تنها پارامتر ورودی است که از کاربر دریافت می‌شود. برای سنجش کارایی این الگوریتم و مقایسه آن با سایر روش‌ها الگوریتم پیشنهادی برروی داده‌های واقعی آزمایش شد. آزمایشات نشان می‌دهد که روش پیشنهادی از لحاظ دقت عملکرد بهتری نسبت به دیگر الگوریتم‌ها دارد و از لحاظ سرعت نیز عملکرد مناسبی دارد.

مراجع

[1] M. Bouguessa, and S. Wang, “Mining projected clusters in high-dimensional spaces,” *IEEE Transactions on Knowledge and Data Engineering,* vol. 21, no. 4, pp. 507-522, 2009.

[2] M. Gagolewski, M. Bartoszuk, and A. Cena, “Genie: A new, fast, and outlier-resistant hierarchical clustering algorithm,” *Information Sciences,* vol. 363, pp. 8-23, 2016.

[3] E. Akbari, H. M. Dahlan, R. Ibrahim, and H. Alizadeh, “Hierarchical cluster ensemble selection,” *Engineering Applications of Artificial Intelligence,* vol. 39, pp. 146-156, 2015.

[4] F. J. Quintana, G. Getz, G. Hed, E. Domany, and I. R. Cohen, “Cluster analysis of human autoantibody reactivities in health and in type 1 diabetes mellitus: a bio-informatic approach to immune complexity,” *Journal of autoimmunity,* vol. 21, no. 1, pp. 65-75, 2003.

[5] L. De Angelis, and J. G. Dias, “Mining categorical sequences from data using a hybrid clustering method,” *European Journal of Operational Research,* vol. 234, no. 3, pp. 720-730, 2014.

[6] J. Sun, W. Chen, W. Fang, X. Wun, and W. Xu, “Gene expression data analysis with the clustering method based on an improved quantum-behaved Particle Swarm Optimization,” *Engineering Applications of Artificial Intelligence,* vol. 25, no. 2, pp. 376-391, 2012.

[7] H.-P. Kriegel, P. Kröger, and A. Zimek, “Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering,” *ACM Transactions on Knowledge Discovery from Data (TKDD),* vol. 3, no. 1, pp. 1, 2009.

[8] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*: Elsevier, 2011.

[9] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise." pp. 226-231.

[10] S. Jahirabadkar, and P. Kulkarni, “Algorithm to determine ε-distance parameter in density based clustering,” *Expert Systems with Applications,* vol. 41, no. 6, pp. 2939-2946, 2014.

[11] T. N. Tran, K. Drab, and M. Daszykowski, “Revised DBSCAN algorithm to cluster data with dense adjacent clusters,” *Chemometrics and Intelligent Laboratory Systems,* vol. 120, pp. 92-96, 2013.

[12] K. M. Kumar, and A. R. M. Reddy, “A fast DBSCAN clustering algorithm by accelerating neighbor searching using Groups method,” *Pattern Recognition,* vol. 58, pp. 39-48, 2016.

[13] O. Uncu, W. A. Gruver, D. B. Kotak, D. Sabaz, Z. Alibhai, and C. Ng, "Gridbscan: Grid density-based spatial clustering of applications with noise." pp. 2976-2981.

[14] H. Darong, and W. Peng, “Grid-based DBSCAN algorithm with referential parameters,” *Physics Procedia,* vol. 24, pp. 1166-1170, 2012.

evolution," *International Journal of Computer Applications,* vol. 91, no. 7, 2014.
[20] O. Maimon, and L. Rokach, *Data mining and knowledge discovery handbook*: Springer, 2005.
[21] E. Alpaydin, *Introduction to machine learning*: MIT press, 2014.
[22] E. Masciari, G. M. Mazzeo, and C. Zaniolo, "A new, fast and accurate algorithm for hierarchical clustering on euclidean distances." pp. 111-122.
[23] Z. Nazari, D. Kang, M. R. Asharif, Y. Sung, and S. Ogawa, "A new hierarchical clustering algorithm." pp. 148-152.
[24] P. Cichosz, *Data Mining Algorithms: Explained Using R*: John Wiley & Sons, 2014.
[25] C. Braune, S. Besecke, and R. Kruse, "Density Based Clustering: Alternatives to DBSCAN," *Partitional Clustering Algorithms*, pp. 193-213: Springer, 2015.
[26] K. Bache and M. Lichman, "UCI machine learning repository," 2013.
[27] C. C. Aggarwal and C. K. Reddy, Data clustering: algorithms and applications: CRC Press, 2013.
[28] P. Berkhin, "A survey of clustering data mining techniques," in Grouping multidimensional data, ed: Springer, 2006, pp. 25-71.
[15] J. Esmaelnejad, J. Habibi, and S. H. Yeganeh, "A novel method to find appropriate ε for DBSCAN." pp. 93-102.
[16] M. T. Elbatta, and W. M. Ashour, "A dynamic method for discovering density varied clusters," *International Journal of Signal Processing, Image Processing and Pattern Recognition,* vol. 6, no. 1, pp. 14, 2013.
[17] M. N. Gaonkar, and K. Sawant, "AutoEpsDBSCAN: DBSCAN with Eps automatic for large dataset," *International Journal on Advanced Computer Theory and Engineering,* vol. 2, no. 2, pp. 11-16, 2013.
[18] K. Sawant, "Adaptive Methods for Determining DBSCAN Parameters," *International Journal of Innovative Science, Engineering & Technology,* vol. 1, no. 4, 2014.
[19] A. Karami, and R. Johansson, "Choosing dbscan parameters automatically using differential