ارائه یک الگوریتم خوشه‌بندی مبتنی بر چگالی با قابلیت کشف خوشه‌های با چگالی متفاوت در پایگاه داده‌های مکانی

علی زاده ده بالایی، علیرضا باقری و حامد افشار

چکیده: خوشه‌بندی یکی از تکنیک‌های مهم کشف دانش در پایگاه داده‌های مکانی است. الگوریتم‌های خوشه‌بندی مبتنی بر چگالی یکی از روش‌های اصلی برای خوشه‌بندی در داده‌کاوی هستند. DBSCAN الگوریتم پایه روش‌های خوشه‌بندی مبتنی بر چگالی است که علی‌رغم مزایایی که دارد دارای مشکلاتی نظیر سخت‌بودن تعیین پارامترهای ورودی و عدم توانایی کشف خوشه‌های با چگالی متفاوت نیز است.

در این مقاله الگوریتمی ارائه شده که برخلاف الگوریتم DBSCAN، قابلیت تشخیص خوشه‌های با چگالی متفاوت را دارد. این الگوریتم همچنین خوشه‌های تودرتو و چسبیده به هم را نیز به خوبی تشخیص می‌دهد. ایده الگوریتم پیشنهادی به این صورت است که ابتدا با استفاده از تکنیکی چگالی‌های مختلف مجموعه داده را تشخیص داده و برای هر چگالی یک شعاع *Eps* تعیین می‌کند. سپس الگوریتم DBSCAN جهت اعمال بر روی مجموعه داده، با پارامترهای به دست آمده تطبیق داده می‌شود. الگوریتم پیشنهادی بر روی مجموعه داده‌های استاندارد و مصنوعی تست شده است و نتایج به دست آمده با نتایج حاصل از الگوریتم DBSCAN و پنج بهبود الگوریتم DBSCAN شامل: VDBSCAN، VMDBSCAN، LDBSCAN، DVBSCAN و MDDBSCAN که همگی برای رفع مشکل تغییرات چگالی الگوریتم DBSCAN ارائه شده‌اند، بر اساس معیارهای ارزیابی روش‌های خوشه‌بندی مقایسه شده‌اند. نتایج ارزیابی‌ها نشان می‌دهد که الگوریتم پیشنهادی از دقت بالا و درصد خطای پایینی برخوردار بوده و نتایج بهتری نسبت به سایر الگوریتم‌ها داشته است.

کلیدواژه: **چگالی متفاوت، خوشه‌بندی مبتنی بر چگالی، داده‌کاوی مکانی، DBSCAN.**

۱- مقدمه

امروزه استفاده از سیستم پایگاه داده مکانی [۱] به عنوان یک سیستم جهت مدیریت داده‌های مکانی و غیر مکانی در زمینه‌های مختلف از جمله در سیستم‌های اطلاعات جغرافیایی و پزشکی روز به روز در حال افزایش است. حجم داده‌های مکانی جمع‌آوری شده به دلایل مختلف از جمله تولید رو به رشد نقشه‌ها به شدت در حال افزایش است. این حجم عظیم داده‌ها بیشتر از آن است که بتواند به صورت دستی تجزیه و تحلیل شود و بنابراین نیاز است که روش‌های کشف دانش از جمله داده‌کاوی روی داده‌های مکانی اعمال شود. داده‌کاوی مکانی [۲] عبارت است از استخراج دانش و روابط مکانی و هر خصوصیت دیگری که به صورت ضمنی در پایگاه داده ذخیره شده است. داده‌کاوی مکانی برای پیداکردن قواعد و روابط ضمنی بین داده‌های مکانی مورد استفاده قرار می‌گیرد. تلاش‌های تحقیقاتی که در زمینه داده‌کاوی مکانی صورت گرفته است در زیر چترهای مختلفی مانند آمار مکانی و محاسبات جغرافیایی قرار گرفته‌اند. خوشه‌بندی [۳] یعنی گروه‌بندی اشیای پایگاه داده در زیرگروه‌های معنادار، یکی از تکنیک‌های مهم کشف دانش در پایگاه داده مکانی است. تجزیه و تحلیل خوشه به طور گسترده‌ای در تجزیه و تحلیل داده‌ها مورد استفاده قرار می‌گیرد، به این صورت که یک مجموعه از اقلام داده‌ای را به داخل گروه‌ها یا خوشه‌هایی سازمان‌دهی می‌کند به طوری که اقلامی که در داخل یک خوشه هستند دارای خصوصیات مشابه به هم و با اقلام موجود در سایر خوشه‌ها متفاوت هستند [۴]. خوشه‌بندی به عنوان یک روش یادگیری بدون نظارت است که با چالش‌های بسیار زیادی مانند بعد مجموعه داده‌ها، اشکال اختیاری از خوشه‌ها، مقیاس‌پذیری، پارامترهای ورودی، دانش دامنه و داده‌های دارای نویز مواجه است. تعداد زیادی از الگوریتم‌های خوشه‌بندی تا این زمان معرفی شده‌اند تا با این چالش‌ها برخورد کنند. تا به امروز هیچ گونه الگوریتم واحدی که قابلیت رسیدگی به همه چالش‌های گفته‌شده را داشته باشد ارائه نشده است. الگوریتم‌های خوشه‌بندی کاربردهای متعددی در زمینه‌های مختلف دارند. از جمله این کاربردها می‌توان به خوشه‌بندی اطلاعات در موتورهای جستجو [۵]، کاوش تصاویر پزشکی [۶]، تشخیص ناهنجاری در اطلاعات دما [۷]، خوشه‌بندی گره‌ها در شبکه‌های حسگر بی‌سیم [۸] و همچنین کاربردهایی نظیر بازاریابی، کتابداری، زیست‌شناسی، نقشه‌برداری شهری، مطالعات زلزله‌نگاری، وب، تشخیص گفتار و تقسیم‌بندی تصاویر اشاره کرد.

روش‌های بسیاری برای انجام خوشه‌بندی پیشنهاد شده است که این روش‌ها را می‌توان به ۵ نوع اصلی دسته‌بندی کرد [۹]: مبتنی بر پارتیشن، مبتنی بر سلسله‌مراتب، مبتنی بر چگالی، مبتنی بر مدل و مبتنی بر شبکه. الگوریتم‌های خوشه‌بندی مبتنی بر چگالی یکی از روش‌های اصلی برای خوشه‌بندی در داده‌کاوی هستند. از جمله مزایای این الگوریتم‌ها این است که این الگوریتم‌ها خودشان را به شکل خوشه‌ها محدود نمی‌کنند و همچنین فهم و درک آنها نیز ساده است. علاوه بر این، این الگوریتم‌ها نیاز ندارند که تعداد خوشه‌ها را از قبل مشخص کنیم. الگوریتم‌های خوشه‌بندی مبتنی بر چگالی بسیاری ارائه شده است. DBSCAN [۱۰] الگوریتم پایه روش‌های خوشه‌بندی مبتنی بر چگالی است. این الگوریتم قابلیت کشف خوشه‌های با اندازه و اشکال متفاوت را از حجم زیادی از داده‌ها دارد و در مقابل نویز نیز مقاوم است. علی‌رغم وجود این مزایا، این الگوریتم چندین مشکل اساسی نیز دارد. اول این که نیاز به دو پارامتر ورودی *Minpts* و *Eps* دارد که تعیین مقدار دقیق این پارامترها به خصوص در پایگاه داده‌های با حجم بالا بسیار سخت است. دوم این که این الگوریتم قابلیت کشف خوشه‌های با چگالی متفاوت را ندارد.

این مقاله در تاریخ ۲۱ آذر ماه ۱۳۹۵ دریافت و در تاریخ ۳۱ مرداد ماه ۱۳۹۶ بازنگری شد.

علی زاده ده بالایی، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران، (email: alizadehei@aut.ac.ir).

علیرضا باقری، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران، (email: ar_bagheri@aut.ac.ir).

حامد افشار، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران، (email: hafshar@aut.ac.ir).

همچنین عدم تشخیص صحیح خوشه‌های نزدیک به هم و پیچیدگی زمانی بالای الگوریتم از جمله مشکلات دیگر الگوریتم DBSCAN محسوب می‌شوند. به دلیل وجود این مشکلات، بهبودهای بسیاری برای الگوریتم DBSCAN ارائه شده است.

هدف این مقاله رفع مشکل تغییرات چگالی الگوریتم DBSCAN است. اولین بهبود برای رفع مشکل تغییرات چگالی الگوریتم DBSCAN در سال ۱۹۹۹ توسط Ankerst و همکارانش تحت عنوان الگوریتم OPTICS [۱۱] ارائه شد. بعد از آن نیز بهبودهای بسیاری برای رفع مشکل تغییرات چگالی الگوریتم DBSCAN ارائه شد. هدف تمامی بهبودهای ارائه‌شده عبارت است از افزایش دقت الگوریتم و به طور هم‌زمان کاهش حساسیت به پارامترهای ورودی و قابلیت تشخیص هر نوع از خوشه‌ها. تعداد زیاد این الگوریتم‌ها موجب سر در گمی کاربران در انتخاب الگوریتم مناسب شده است. به عبارت دیگر الگوریتم مناسبی که مشکل تغییرات چگالی الگوریتم DBSCAN را حل کرده باشد به گونه‌ای که همه جنبه‌های ذکرشده در بالا را در نظر گرفته باشد هنوز ارائه نشده است. در الگوریتم‌های ارائه‌شده برای حل مشکل تغییرات چگالی DBSCAN توازن خوبی بین دقت الگوریتم‌ها و تعداد پارامترهای ورودی وجود ندارد. به عنوان مثال، الگوریتمی مانند الگوریتم VDBSCAN [۱۲] هر چند که تعداد پارامترهای ورودی را کاهش داده است اما از طرفی دقت این الگوریتم بسیار پایین است به گونه‌ای که تنها بر روی برخی مجموعه داده‌های خاص عملکرد مناسبی دارد. از طرفی الگوریتمی مانند DVBSCAN [۱۳] اگرچه دقت مناسبی دارد اما تعداد زیاد پارامترهای این الگوریتم باعث شده که انتخاب دقیق این پارامترها برای کاربران مشکل باشد. از این رو ارائه یک الگوریتم واحد که به حل همه مشکلات ذکرشده در بالا بپردازد بسیار ضروری است.

با توجه به مطالبی که در بالا گفته شد در این مقاله برای رفع مشکل تغییرات چگالی DBSCAN، الگوریتمی ارائه شده که ضمن ساده‌بودن و قابل فهم بودن، تمامی جنبه‌های ذکرشده را در نظر داشته است. الگوریتم ارائه‌شده ضمن تشخیص خوشه‌های با چگالی متفاوت، حساسیت پایینی نسبت به پارامترهای ورودی دارد و همچنین قابلیت تشخیص خوشه‌های با اندازه و اشکال متفاوت و خوشه‌های چسبیده به هم و تو در تو را نیز دارد. ایده الگوریتم پیشنهادی به این صورت است که ابتدا با استفاده از تکنیکی مقادیر مختلف پارامتر *Eps* را به دست می‌آورد. سپس الگوریتم DBSCAN جهت اعمال بر روی مجموعه داده با پارامترهای به دست آمده تطبیق داده می‌شود. الگوریتم پیشنهادی بر روی مجموعه داده‌های استاندارد تست شده و نتایج به دست آمده با نتایج حاصل از الگوریتم پایه DBSCAN و همچنین پنج الگوریتم VDBSCAN، VMDBSCAN، LDBSCAN، DVBSCAN و MDDBSCAN که همگی برای رفع مشکل تغییرات چگالی الگوریتم DBSCAN ارائه شده‌اند، مقایسه شده است. نتایج ارزیابی نشان می‌دهد که الگوریتم پیشنهادی از شاخص شباهت بالاتر و درصد خطای پایین‌تری نسبت به سایر الگوریتم‌های مورد ارزیابی برخوردار است.

بقیه مقاله در بخش‌های زیر سازمان‌دهی شده است: در بخش ۲ برخی از بهبودهای ارائه‌شده برای الگوریتم DBSCAN را معرفی می‌کنیم. با توجه به این که ما قصد بهبود الگوریتم DBSCAN را داریم در بخش ۳ به معرفی الگوریتم DBSCAN می‌پردازیم و مشکلات این الگوریتم را نیز بیان می‌کنیم. سپس در بخش ۴ الگوریتم پیشنهادی را ارائه می‌دهیم. در بخش ۵ نیز نتایج ارزیابی‌ها نشان داده شده و در پایان نیز در بخش ۶ نتیجه‌گیری و کارهای آتی ارائه شده است.

۲- مروری بر کارهای گذشته

الگوریتم‌های خوشه‌بندی مبتنی بر چگالی بسیار محبوب هستند. DBSCAN پایه الگوریتم‌های خوشه‌بندی مبتنی بر چگالی است. این الگوریتم علی‌رغم مزایایی که دارد دارای مشکلاتی نیز است. از سال ۱۹۶۶ که این الگوریتم ارائه شده است تا به امروز بهبودهای بسیاری برای این الگوریتم ارائه شده است.

الگوریتم OPTICS [۱۱] الگوریتم DBSCAN را جهت حل مسئله تغییر چگالی تطبیق داده است. این الگوریتم برای حل مسئله تغییر چگالی دو فیلد اضافی فاصله دسترسی‌پذیری[1] و فاصله مرکز[2] را ذخیره می‌کند. فاصله مرکز یک شیء مانند p عبارت است از فاصله شیء p از $Minpts$امین همسایه‌اش به شرطی که p یک شیء مرکزی باشد. همچنین فاصله دسترسی‌پذیری شیء p از شیء o، کوچک‌ترین فاصله‌ای است که در آن فاصله، شیء p دسترسی‌پذیر چگالی مستقیم از o باشد به شرطی که o یک شیء مرکزی باشد. با استفاده از این فاصله‌ها، OPTICS یک مرتب‌سازی که نشان‌دهنده ساختار خوشه‌بندی است بر روی داده‌ها می‌سازد. در این الگوریتم پارامتر *Eps* برای تشخیص گودی‌ها در Reachabiliy Plot که نشان‌دهنده خوشه‌ها هستند ضروری است. الگوریتم OPTICS به جای تولید خوشه‌های با چگالی محلی مشابه، تنها خوشه‌های با چگالی محلی بیش از یک حد آستانه را تولید می‌کند. در پایگاه داده‌های با اندازه متوسط OPTICS زمان اجرا تقریباً ۱/۶ برابر DBSCAN را دارد.

الگوریتم VDBSCAN [۱۲] به منظور رفع مشکل الگوریتم DBSCAN در تجزیه و تحلیل خوشه‌های با چگالی متفاوت ارائه شده است. ایده این الگوریتم به این صورت است که قبل از اعمال الگوریتم DBSCAN با استفاده از مفهوم k-dist plot چگالی‌های مختلف را شناسایی کرده و برای هر چگالی یک مقدار *Eps* متناسب را برمی‌گزیند. بعد از تعیین مقادیر مختلف *Eps*، الگوریتم DBSCAN را به تعداد چگالی‌های به دست آمده با استفاده از مقادیر مختلف *Eps* به دست آمده بر روی مجموعه داده اعمال می‌کند. منحنی k-dist plot از مرتب‌سازی نقاط مجموعه داده بر اساس فاصله هر نقطه از kامین نزدیک‌ترین همسایه‌اش ساخته می‌شود. بعد از ساخت منحنی k-dist plot، هر تغییر شدید در این منحنی یک چگالی متفاوت را مشخص می‌کند. الگوریتم VDBSCAN وابستگی به تعیین دقیق مقدار پارامتر k دارد به گونه‌ای که انتخاب نادرست آن باعث کاهش دقت نتایج می‌شود.

یکی دیگر از بهبودهای الگوریتم DBSCAN که در سال ۲۰۰۶ معرفی شد، الگوریتم DCBRD [۱۴] است. این الگوریتم قابلیت تشخیص خوشه‌های با اشکال و اندازه‌های مختلف از مجموعه داده‌های با ابعاد بالا را دارد اما بر خلاف الگوریتم DBSCAN وابستگی به مقادیر پارامتر تعیین‌شده توسط کاربر ندارد. در این الگوریتم از دانش به دست آمده از مجموعه داده‌های مبنایی استفاده شده و سپس عمل خوشه‌بندی انجام شده است. به طور کلی در این الگوریتم عمل خوشه‌بندی در دو مرحله انجام می‌شود. در ابتدا فضای داده به نواحی مدور همپوشان تقسیم‌بندی می‌شود. در این مرحله عمل تقسیم‌بندی به گونه‌ای انجام می‌شود که شعاع هر ناحیه (بر اساس فرمول ابداعی در مقاله) بزرگ‌تر از یک حد آستانه چگالی مورد انتظار *Eps* باشد. بعد از تقسیم‌بندی فضای داده، الگوریتم DBSCAN با استفاده از یک مقدار بهینه از *Eps* که از

1. Reachability Distance
2. Core Distance

دایره‌ای که همه فضای داده را پوشش می‌دهد محاسبه می‌شود، بر روی هر ناحیه اعمال می‌گردد. بر اساس آزمایش‌های انجام‌شده، این الگوریتم از لحاظ دقت بهبود چندانی نسبت به الگوریتم DBSCAN به دست نیاورده است و تقریباً نتایج مشابه با الگوریتم DBSCAN را تولید می‌کند. از طرفی نقطه مثبت این الگوریتم نسبت به الگوریتم DBSCAN سرعت بهتر آن در یافتن خوشه‌ها است.

الگوریتم LDBSCAN [۱۵] یکی دیگر از بهبودهای الگوریتم DBSCAN است که از مفهوم فاکتور دورافتادگی محلی (LOF) [۱۶] و چگالی دسترسی‌پذیری محلی (LRD) جهت تشخیص خوشه‌های با چگالی متفاوت استفاده می‌کند. در این الگوریتم جهت تشخیص نویز از فاکتور دورافتادگی محلی استفاده شده است به گونه‌ای که اگر فاکتور دورافتادگی محلی یک نقطه کمتر از یک حد آستانه باشد، آن نقطه یک نقطه مرکزی است و در غیر این صورت نویز محسوب می‌شود. در هنگام بسط یک خوشه، یک نقطه در صورتی بسط داده می‌شود که چگالی دسترسی‌پذیری محلی آن نقطه نزدیک به چگالی دسترسی‌پذیری محلی نقطه مرکزی خوشه متعلق به آن باشد. در غیر این صورت آن نقطه به طور ساده به خوشه افزوده شده و بسط داده نمی‌شود. انتخاب پارامترهای مناسب برای الگوریتم LDBSCAN نسبت به الگوریتم DBSCAN ساده‌تر است اما با توجه به این که این الگوریتم نیاز به ۴ پارامتر ورودی دارد، برای پایگاه داده‌های حجیم، این تعداد زیاد پارامترهای ورودی ممکن است مشکل‌ساز شود.

الگوریتم DVBSCAN [۱۳] از مفهوم واریانس چگالی خوشه (CDV) و شاخص شباهت خوشه (CSI) به منظور جلوگیری از بسط خوشه از ناحیه متراکم به ناحیه متراکم‌تر و برعکس استفاده می‌کند. الگوریتم با انتخاب یک نقطه مرکزی شروع به شکل‌دهی خوشه‌ها می‌کند و سپس همه نقاطی که در همسایگی Eps نقطه مرکزی انتخابی باشند را به یک صف وارد می‌کند. این نقاط در صورتی اجازه بسط پیدا می‌کنند که واریانس چگالی خوشه کمتر یا مساوی از حد آستانه α باشد و شاخص شباهت خوشه یعنی اختلاف بین حداقل و حداکثر شیء قرارگرفته در خوشه نیز کمتر از حد آستانه γ باشد. در غیر این صورت نقطه به طور ساده به خوشه افزوده شده و دیگر بسط داده نمی‌شود. این الگوریتم علاوه بر دو پارامتر استفاده‌شده در DBSCAN نیاز به تعیین دو پارامتر α و γ که به منظور محدودکردن مقدار تغییر چگالی محلی اجازه داده شده در داخل خوشه استفاده می‌شوند، نیز دارد. الگوریتم DVBSCAN قابلیت تشخیص خوشه‌های با اندازه، اشکال و چگالی متفاوت را دارد و در مقابل نویز نیز مقاوم است. با این حال، این الگوریتم نیاز به تعیین ۴ پارامتر ورودی دارد که تعیین ۴ پارامتر به مراتب سخت‌تر از تعیین ۲ پارامتر نسبت به الگوریتم DBSCAN است. این در حالی است که نتایج این الگوریتم بسیار وابسته به تعیین دقیق این پارامترها است.

در سال ۲۰۱۳ بهبود دیگری از الگوریتم DBSCAN با عنوان AutoEpsDBSCAN [۱۷] ارائه شد که هدف آن کاهش پارامترهای ورودی و در نتیجه کاهش خطاهای ایجادشده به دلیل دخالت کاربر است. این الگوریتم بر خلاف الگوریتم DBSCAN توانایی تشخیص خوشه‌های با چگالی متفاوت را دارد. این الگوریتم ابتدا میانگین فاصله هر نقطه از همه k نزدیک‌ترین همسایه‌هایش را به دست می‌آورد و سپس این k-Distanceها را به ترتیب صعودی Plot می‌کند. برای برآورد مجموعه پارامترهای Eps نیاز به تعیین "زانوها[1]" است. "زانو" معادل با یک تغییر شدید در امتداد منحنی k-distance است. اگر دقت کرده باشید متوجه می‌شوید که روال کار این الگوریتم شبیه با روال کار الگوریتم VDBSCAN است. اما این الگوریتم بر خلاف الگوریتم VDBSCAN که تنها kامین نزدیک‌ترین همسایه را در محاسبه فاصله در نظر می‌گیرد، میانگین فاصله از همه k نزدیک‌ترین همسایه‌ها را لحاظ می‌کند. این کار به تشخیص نویز کمک می‌کند و باعث می‌شود تا تشخیص خودکار حد آستانه چگالی‌ها ساده‌تر شود اما این روش نیز همان مشکلات اساسی الگوریتم VDBSCAN را دارد. در واقع مشکل اصلی الگوریتم VDBSCAN تشخیص چگالی‌های متفاوت از روی تغییرات شدید منحنی k-dist plot بود که این مشکل در الگوریتم ارائه‌شده در [۱۷] همچنان پابرجا است. یعنی این الگوریتم در مجموعه داده‌هایی که تفاوت چگالی خوشه‌های آن کم باشد قادر به تشخیص صحیح پارامترها نیست و تنها بر روی مجموعه داده‌هایی قادر به تشخیص پارامترها است که فاقد منحنی ملایم در k-dist plot مربوطه‌شان باشند. این الگوریتم نتایج بهتری نسبت به الگوریتم DBSCAN تولید می‌کند منتها هنوز نیاز به یک پارامتر k دارد که باید توسط کاربر تعیین شود.

الگوریتم DDSC [۱۸] الگوریتمی دیگر جهت یافتن خوشه‌های با چگالی متفاوت است و در این الگوریتم از مفهوم شیء مرکزی هم‌جنس استفاده شده است. شیء مرکزی هم‌جنس به شیئی گفته می‌شود که اولاً یک شیء مرکزی باشد و ثانیاً اختلاف چگالی با همسایه‌هایش در حد α باشد. الگوریتم با انتخاب یک شیء مرکزی هم‌جنس کار خودش را آغاز می‌کند و تا زمانی خوشه را بسط می‌دهد که به یک شیء مرکزی غیر هم‌جنس که نشان‌دهنده تغییر وسیع در چگالی است برسد. پیچیدگی زمانی این الگوریتم همانند الگوریتم DBSCAN است و یکی از مزایای آن کاهش وابستگی به پارامتر Eps است و این در حالی است که الگوریتم پیشنهادی برای رسیدن به این هدف پارامتر سومی (پارامتر α) را نیز به الگوریتم افزوده است.

الگوریتم MDDBSCAN [۱۹] توسعه دیگری از الگوریتم DBSCAN است که قابلیت تشخیص خوشه‌های با چگالی متفاوت را دارد. الگوریتم در ابتدا چگالی هر نقطه (DST_P) را محاسبه می‌کند و سپس متراکم‌ترین نقطه را به عنوان شیء مرکزی در نظر می‌گیرد و شروع به بسط خوشه به وسیله نقاط همسایه با چگالی مشابه می‌کند. در هنگام بسط خوشه از بین نقاطی که جزء k نزدیک‌ترین همسایه شیء مرکزی هستند، تنها نقاطی که چگالیشان کمتر از میانگین چگالی خوشه باشد به خوشه افزوده می‌شوند و بسط پیدا می‌کنند. در واقع در این الگوریتم، میانگین چگالی خوشه ($AVGDST_C$) و چگالی هر نقطه (DST_P) تصمیم می‌گیرند که نقطه‌ای متعلق به خوشه‌ای باشد یا خیر. در این مقاله به مسایل پل نویز و شکاف خوشه‌ها نیز اشاره شده و این دو مسئله به دلیل انتخاب نامناسب مقدار k رخ می‌دهند. بنابراین با انتخاب یک مقدار مناسب و دقیق برای k می‌توانیم بر این مسایل غلبه کنیم.

الگوریتم VMDBSCAN [۲۰] الگوریتمی دیگر جهت غلبه بر مسئله تغییرات چگالی خوشه‌ها است. این الگوریتم ابتدا تابع چگالی هر نقطه را به دست می‌آورد و سپس با اعمال DBSCAN بر روی مجموعه داده، مرکز هر خوشه را به دست می‌آورد. سپس به ازای هر شیء اگر تابع چگالی کلی آن شیء با توجه به مرکز خوشه‌ای که متعلق به آن است بیشتر از تابع چگالی کلی با توجه به مرکز خوشه‌های دیگر باشد، آن نقطه را به سمت خوشه‌ای که شیء مرکزی آن حداکثر تأثیر را بر روی آن شیء دارد حرکت می‌دهد. این الگوریتم نسبت به DBSCAN تعداد صحیح‌تری از خوشه‌ها را تشخیص می‌دهد، هر چند که برای داده‌های با بعد بالا

1. Knees

همچنان تعداد خوشه‌های ناصحیح قابل توجه است.

KDDClus [۲۱] الگوریتمی با قابلیت تشخیص خوشه‌های با چگالی متفاوت است که از ساختار داده KD-Tree برای پردازش کارای داده‌های با ابعاد بالا استفاده می‌کند. در واقع استفاده از ساختار داده KD-Tree محاسبه کارای kامین نزدیک‌ترین همسایه‌ها را خصوصاً برای مجموعه داده‌های بزرگ ممکن می‌سازد. روال کار این الگوریتم به این صورت است که برای هر نقطه فاصله تا kامین نزدیک‌ترین همسایه را با استفاده از ساختار داده KD-Tree محاسبه کرده و سپس با مشخص‌کردن زانوها از روی منحنی k-dist، مجموعه پارامترهای *Eps* را تخمین می‌زند. یکی از مشکلات این الگوریتم نیاز به پارامتر ورودی k است. با بررسی روش توضیح داده شده در این مقاله به راحتی می‌توان گفت که این الگوریتم نیز نسخه‌ای از الگوریتم VDBSCAN است که با داشتن پرس و جوهای ناحیه‌ای بهینه، به سبب استفاده از ساختار شاخص KD-Tree برای مجموعه داده‌های بزرگ نیز قابل استفاده است. بنابراین مشکلات اساسی الگوریتم VDBSCAN برای این الگوریتم نیز مطرح است.

الگوریتم VDSC [۲۲] یک روش خوشه‌بندی مؤثر است که قابلیت تشخیص خوشه‌های تو در تو در فضای با چگالی متفاوت را دارد. در این الگوریتم ابتدا فاصله از مرکز هر نقطه محاسبه می‌شود و سپس شیء با حداقل فاصله از مرکز به عنوان شیء مرکزی در نظر گرفته می‌شود و فرایند بسط خوشه آغاز می‌گردد. فاصله از مرکز یک شیء مثل p عبارت است از حداکثر فاصله‌ای که شرط *Minpts* را ارضا کند. در این الگوریتم از مفاهیم دسترسی‌پذیر مستقیم سطح ۱، دسترسی‌پذیر مستقیم سطح ۲ و فاصله از مرکز بسط‌یافته استفاده شده است. پیچیدگی زمانی این الگوریتم مشابه با پیچیدگی زمانی الگوریتم DBSCAN است اما برخلاف الگوریتم DBSCAN، این الگوریتم قابلیت تشخیص خوشه‌های تو در تو در فضای با چگالی متفاوت را دارد. این الگوریتم برای تشخیص خوشه‌ها از پارامتر *Eps* صرف نظر کرده است اما به منظور تشخیص خوشه‌های با چگالی متفاوت، پارامتر دیگری (پارامتر α) به الگوریتم اضافه شده است. انتخاب این پارامتر برای مجموعه داده‌های بزرگ زمان‌بر و مشکل است و از این رو نیاز است که این پارامتر به صورت پویا تولید شود.

الگوریتم VDBSCAN قابلیت تشخیص خوشه‌های با چگالی متفاوت را دارد و در مقال نویز نیز مقاوم است اما این الگوریتم تا حد زیادی به پارامترهای *Eps* و *Minpts* بستگی دارد و نیاز دارد که این دو پارامتر با دقت بسیار بالایی انتخاب شوند. در [۲۳] روشی برای انتخاب خودکار این دو پارامتر ارائه شده است.

در سال ۲۰۱۶ الگوریتمی با عنوان MDBSCAN [۲۴] جهت تشخیص خوشه‌های با چگالی متفاوت ارائه شده است که در آن با استفاده از روش‌های آماری دو پارامتر *Eps* و *Minpts* استخراج می‌شوند. یکی از ایرادات اساسی این مقاله، عدم مقایسه الگوریتم ارائه‌شده با سایر الگوریتم‌های هم‌رده آن می‌باشد به گونه‌ای که الگوریتم ارائه‌شده تنها با الگوریتم پایه DBSCAN مقایسه شده و عملکرد آن در برابر سایر الگوریتم‌هایی که برای حل مشکل تغییرات چگالی ارائه شده‌اند، مشخص نیست.

اکثر روش‌های ارائه‌شده برای حل مشکل تغییرات چگالی الگوریتم DBSCAN روش‌های بدون نظارت[1] هستند که در آنها از دانش قبلی برای بهبود نتایج خوشه‌بندی استفاده نمی‌شود. در [۲۵] نشان داده شده است که چگونه می‌توان با استفاده از دانش زمینه‌ای در الگوریتم‌های خوشه‌بندی مبتنی بر چگالی برای تشخیص خوشه‌های با چگالی متفاوت استفاده کرد. روش ارائه‌شده در این مقاله به این صورت است که در مرحله اول مجموعه داده را به سطوح چگالی متفاوت تقسیم می‌کند و برای هر سطح چگالی پارامترهای چگالی مناسب مربوط به آن سطح را تعیین می‌کند. در ادامه الگوریتم مربوطه با استفاده از محدودیت‌های دو به دو فرایند خوشه‌بندی را بر مبنای پارامترهای به دست آمده بسط می‌دهد. نتایج ارزیابی این الگوریتم نشان می‌دهد که الگوریتم پیشنهادی نتایج بهتری نسبت به برخی از الگوریتم‌های خوشه‌بندی نیمه‌نظارتی و بدون نظارت ارائه می‌دهد.

الگوریتم MDCUT [۲۶] بهبود دیگری از الگوریتم DBSCAN است که قابلیت تشخیص خوشه‌های با چگالی متفاوت را دارد. این الگوریتم در دو فاز کار می‌کند. در فاز اول با استفاده از فرایند ریاضی Spline (یک تابع هموار چندضابطه‌ای- چندجمله‌ای) بر روی فواصل k نزدیک‌ترین همسایه، تعداد سطوح چگالی مشخص می‌شوند. در مرحله بعدی از سطوح چگالی به دست آمده در مرحله اول به عنوان آستانه‌های چگالی محلی برای تشخیص خوشه‌های با چگالی و اشکال مختلف استفاده می‌شود. این الگوریتم در مقایسه با الگوریتم DBSCAN از دقت بالاتری برخوردار است.

الگوریتم‌های خوشه‌بندی مبتنی بر چگالی روش‌های بسیار ارزشمندی برای خوشه‌بندی جریان‌های داده هستند. اخیراً تعدادی الگوریتم خوشه‌بندی مبتنی بر چگالی برای خوشه‌بندی جریان داده‌ها ارائه شده که یکی از ایرادات اصلی این الگوریتم‌ها کاهش کیفیت خوشه‌بندی به دلیل وجود خوشه‌های با چگالی متفاوت است. در [۲۷] الگوریتمی ارائه شده که توانایی خوشه‌بندی جریانات داده‌ای با چگالی‌های متفاوت را دارد. الگوریتم ارائه‌شده از روش Grid_base برای مدیریت نویز و داده‌های با چگالی متفاوت و همچنین برای کاهش زمان ادغام خوشه‌ها استفاده کرده است. همچنین در [۲۸] نیز الگوریتم دیگری برای خوشه‌بندی جریانات داده‌ای غیر ایستا ارائه شده که این الگوریتم توانایی تشخیص خوشه‌های با چگالی متفاوت از جریانات داده‌ای را دارد. یکی از مزایای الگوریتم ارائه‌شده، کاهش وابستگی به پارامترهای ورودی است.

با توجه به این که الگوریتم DBSCAN زمانی که خوشه‌ها نزدیک به هم باشند ممکن است با شکست مواجه شود، در [۲۹] الگوریتمی جهت رفع مشکل خوشه‌های مجاور ارائه شده است. در این الگوریتم به جای استفاده از مفهوم دسترسی‌پذیری چگالی[2] از مفهوم دسترسی‌پذیری چگالی مرکزی[3] استفاده شده است. در واقع این الگوریتم در زنجیره دسترسی‌پذیری بهبود انجام داده است به گونه‌ای که این زنجیره تنها شامل اشیای مرکزی است. روال کار به این صورت است که ابتدا خوشه‌های متشکل از اشیای مرکزی پیدا شده و سپس اشیای حاشیه‌ای به نزدیک‌ترین شیء مرکزی تخصیص داده می‌شوند. این الگوریتم بر روی هر دوی داده‌های مکانی و غیر مکانی قابل اعمال است. با توجه به هدف اصلی الگوریتم ارائه‌شده، این الگوریتم به خصوص در مجموعه داده‌های متراکم شامل خوشه‌های نزدیک به هم به خوبی عمل می‌کند و در سایر موارد نتایج نزدیک به الگوریتم DBSCAN را تولید می‌کند.

یکی از مشکلات الگوریتم DBSCAN بحث پیچیدگی زمانی بالای این الگوریتم در مجموعه داده‌های با حجم و بُعد بالا است. FDBSCAN [۳۰] یکی از الگوریتم‌هایی است که با هدف بهبود زمان الگوریتم

1. Unsupervised

2. Density Reachable

3. Core Density Reachable

DBSCAN ارائه شده است. در این الگوریتم اشیایی که توسط الگوریتم به عنوان بذر برای بسط‌دادن انتخاب می‌شوند، بهبود داده شده‌اند تا به کارایی بهتری برسیم. یعنی با توجه به این که بسیاری از پرس و جوهای ناحیه‌ای برای یافتن همسایه‌های اشیاء، قابل چشم‌پوشی هستند، الگوریتم FDBSCAN تعدادی شیء را به نمایندگی از تمامی اشیا انتخاب می‌کند و پرس و جوی ناحیه‌ای را تنها بر روی آن اشیا انجام می‌دهد. هر چند که این الگوریتم کارایی بهتری نسبت به DBSCAN دارد و پیچیدگی محاسباتی و زمانی کمتری نیز دارد اما نسبت به DBSCAN دقت پایین‌تری دارد و اشیای بیشتری در آن از دست می‌روند. علاوه بر این الگوریتم در [۳۱] تا [۳۳] نیز روش‌هایی برای بهبود پیچیدگی زمانی الگوریتم DBSCAN ارائه شده است.

الگوریتم DBSCAN متکی بر مفهوم چگالی خوشه‌ها است و به منظور کشف خوشه‌های با اشکال مختلف به همراه نویز است. در [۳۴] الگوریتمی ارائه شده که قادر است اشکال هندسی غیر از نقطه مانند چندضلعی‌های دوبعدی را نیز خوشه‌بندی کند. علاوه بر این، این الگوریتم توانایی این را دارد تا اشیای نقطه را به خوبی اشیای بسط‌یافته مکانی بر طبق هر دو خصوصیت مکانی و غیر مکانی آن اشیا خوشه‌بندی کند. علاوه بر این در این مقاله کاربردهایی از دنیای واقعی مانند علوم زمین، زیست‌شناسی، نجوم و جغرافیا برای این الگوریتم ارائه شده است. الگوریتم ارائه‌شده به خصوص در پایگاه داده‌های بسیار بزرگ خوب عمل می‌کند. این الگوریتم خوشه‌بندی یک‌سطحی را ایجاد می‌کند و این در حالی است که ممکن است خوشه‌بندی سلسله‌مراتبی مفیدتر باشد، به خصوص زمانی که پارامترهای ورودی مناسب را نتوان با دقت برآورد کرد.

ST-DBSCAN [۳۵] یکی دیگر از توسعه‌های الگوریتم DBSCAN است که بر خلاف الگوریتم DBSCAN قابلیت کشف خوشه‌ها مطابق با مقادیر مکانی، غیر مکانی و زمانی اشیا را دارد. این الگوریتم از سه جهت نسبت به الگوریتم DBSCAN تفاوت دارد. اول این که بر خلاف الگوریتم DBSCAN و دیگر الگوریتم‌های مبتنی بر چگالی موجود، قابلیت خوشه‌بندی داده‌های مکانی- زمانی را بر طبق خصوصیات مکانی، غیر مکانی و زمانی اشیا دارد. دوم این که بر خلاف DBSCAN در مواقعی که خوشه‌های با چگالی متفاوت در مجموعه داده وجود داشته باشند نیز قابلیت تشخیص نویز را دارد و در نهایت اگر مقادیر غیر مکانی اشیای همسایه تفاوت اندکی داشته باشند و خوشه‌ها مجاور یکدیگر باشند، مقادیر اشیای حاشیه‌ای در یک طرف خوشه ممکن است بسیار متفاوت با مقادیر اشیای حاشیه‌ای در طرف مقابل باشند. الگوریتم ST-DBSCAN این مشکل را با مقایسه مقدار میانگین یک خوشه با مقادیر اشیای جدید افزوده‌شده به خوشه، حل می‌کند. پیچیدگی زمانی این الگوریتم همانند DBSCAN است. الگوریتم ST-DBSCAN کاربردهای متعددی دارد که از جمله این کاربردها می‌توان به سیستم اطلاعات جغرافیا، تصاویر پزشکی و پیش‌بینی وضع هوا اشاره کرد. یکی از نقاط ضعف این الگوریتم عدم توانایی کشف خوشه‌های با چگالی متفاوت است. همچنین پارامترهای ورودی آن به صورت خودکار تولید نمی‌شوند.

انباره داده یک منبع داده بسیار مناسب برای روش‌های داده‌کاوی از جمله خوشه‌بندی است. در محیط‌های انباره داده، به صورت دوره‌ای حجمی از داده‌ها به داده‌های موجود در انباره اضافه می‌شود. از این رو نیاز داریم تا قبل از این که انباره داده در دسترس کاربران قرار بگیرد، خوشه‌هایی که از قبل کشف شده‌اند را با توجه به داده‌های جدید افزوده‌شده به روز کنیم. در [۳۶] یک الگوریتم خوشه‌بندی مبتنی بر چگالی افزایشی ارائه شده که توانایی استفاده در محیط‌های انباره داده شامل داده‌های با چگالی متفاوت را دارد. این الگوریتم علاوه بر افزودن افزایشی نقاط، توانایی افزودن افزایشی خوشه‌ها را نیز دارد. الگوریتم ارائه‌شده در این مقاله با عنوان IMD_DBSCAN نسخه افزایشی الگوریتم MDDBSCAN [۱۹] است و روش کار آن به این صورت است که ابتدا نقاط جدید اضافه‌شده به انباره داده را با استفاده از الگوریتم MDDBSCAN خوشه‌بندی می‌کند و سپس خوشه‌های حاصل را به خوشه‌های موجود در انباره داده اضافه می‌کند. الگوریتم ارائه‌شده نسبت به الگوریتم MDDBSCAN از دقت بالاتری برخوردار است و همچنین نیاز به پرس و جوی ناحیه‌ای کمتر و در نتیجه سرعت بالاتری نیز دارد.

۳- الگوریتم DBSCAN

۳-۱ معرفی الگوریتم DBSCAN

با توجه به این که الگوریتم پیشنهادشده، یک نسخه بهبودیافته از الگوریتم DBSCAN است ضروری است که ابتدا الگوریتم DBSCAN شرح داده شود. الگوریتم DBSCAN نیاز به تعیین ۲ پارامتر $Minpts$ و Eps دارد. این دو پارامتر برای تعیین حداقل چگالی یک خوشه مورد استفاده قرار می‌گیرند.

به منظور درک الگوریتم DBSCAN لازم است که ابتدا برخی از تعاریف مورد استفاده در این الگوریتم معرفی شوند [۱۰]:

تعریف ۱: همسایه‌های شعاع Eps یک نقطه: همسایه‌های موجود در شعاع Eps یک نقطه مثل p که با $NEps(p)$ نشان داده می‌شوند مجموعه‌ای از نقاط هستند که فاصله‌شان از p کمتر از شعاع Eps باشد یعنی

$$NEps(p) = \{q \in D \,|\, Dist(p,q) \le Eps\} \qquad (۱)$$

تعریف ۲: شیء مرکزی: به شیئی که حداقل تعداد $Minpts$ شیء در همسایگی شعاع Eps خود را داشته باشد شیء مرکزی گفته می‌شود.

تعریف ۳: دسترسی‌پذیر چگالی مستقیم: نقطه p دسترسی‌پذیر چگالی مستقیم از نقطه q است اگر اولاً p جزء همسایه‌های شعاع Eps شیء q باشد و ثانیاً شیء q یک شیء مرکزی باشد.

تعریف ٤: دسترسی‌پذیر چگالی: نقطه p دسترسی‌پذیر چگالی از نقطه q است اگر یک زنجیره از نقاط $p_1, p_2, p_3, \ldots, p_n$ که $p_1 = q$ و $p_n = p$ وجود داشته باشد به گونه‌ای که p_{i+1} دسترسی‌پذیر چگالی مستقیم از p_i باشد.

تعریف ۵: متصل چگالی: نقطه p متصل چگالی از نقطه q است اگر یک نقطه مثل o وجود داشته باشد به گونه‌ای که هر دوی p و q دسترسی‌پذیر چگالی از o باشند.

تعریف ٦: خوشه: فرض کنید که D یک پایگاه داده از نقاط باشد. خوشه C یک زیرمجموعه غیر تهی از D است به گونه‌ای که شرط‌های زیر را ارضا کند:

- به ازای همه جفت نقاط p و q اگر $p \in C$، یعنی p یکی از اعضای خوشه C باشد و همچنین q نیز دسترسی‌پذیر چگالی از p باشد آن گاه q نیز باید متعلق به خوشه C باشد (شرط حداکثر بودن).
- به ازای همه جفت نقاط p و q متعلق به خوشه C، p باید متصل چگالی از q باشد (شرط اتصال).

تعریف ۷: نویز: فرض کنید که $C_1, C_2, \ldots, C_k$ خوشه‌های یافت‌شده از پایگاه داده D باشند. به مجموعه‌ای از نقاط که در پایگاه داده D وجود دارند ولی متعلق به هیچ یک از خوشه‌های یافت‌شده $C_{i=1 \text{ to } k}$

Algorithm DBSCAN
Input: $D, Minpts, Eps$
Output: Set of clusters
1: **begin**
2: $C = 0$
3: **for** (each) unvisited point p in the dataset D **do**
4: Mark p as visited
5: N = regionQuery (p, Eps)
6: **if** sizeof $(N) < Minpts$ **then**
7: Mark p as Noise
8: **else**
9: $C = C + 1$
10: Enlargecluster $(Eps, Minpts, C, p, N)$
11: **end if**
12: **end for**
13: **end**

Function *Enlargecluster* $(Eps, Minpts, C, p, N)$
1: **begin**
2: Add p to cluster C
3: **for** (each) point p' in N **do**
4: **if** p' is unvisited **then**
5: Mark p' as visited
6: N' = regionQuery (p', Eps)
7: **if** sizeof $(N') >= Minpts$ **then**
8: $N = N'$ combine to N
9: **end if**
10: **if** p' is not in any cluster **then**
11: Add p' to cluster C
12: **end if**
13: **end if**
14: **end for**
15: **end function**

شکل ۱: الگوریتم DBSCAN [۱۰].

نباشند نویز می‌گویند

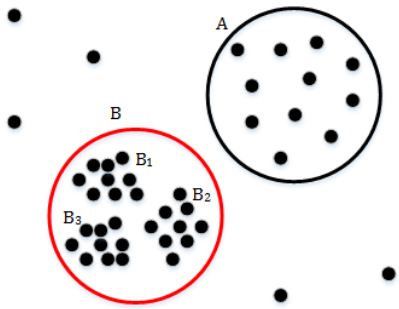
$$Noise = \{p \in D \mid \forall i : p \notin C_i\} \qquad (۲)$$

تعریف ۸: شیء حاشیه‌ای: شیء حاشیه‌ای به شیئی گفته می‌شود که شیء مرکزی نباشد منتها از یک شیء مرکزی دیگر دسترسی‌پذیر چگالی باشد.

اکنون با توجه به تعاریف ارائه‌شده می‌توانیم الگوریتم DBSCAN را شرح دهیم. الگوریتم با یک نقطه اختیاری p از مجموعه داده شروع می‌کند و همه اشیای دسترسی‌پذیر چگالی از آن نقطه را بازیابی می‌کند. اگر p یک شیء مرکزی باشد، یک خوشه شکل‌دهی می‌شود و در غیر این صورت الگوریتم نقطه بعدی از مجموعه داده را ملاقات می‌کند. این فرایند تا زمانی ادامه دارد که همه نقاط مجموعه داده پردازش شوند. پیچیدگی زمانی این الگوریتم در صورت استفاده از ساختارهای شاخص مکانی مانند $R*Tree$ از مرتبه $O(n \log n)$ است. شبه‌کد این الگوریتم در شکل ۱ نشان داده شده است.

۳-۲ بررسی مشکلات الگوریتم DBSCAN

اکثر روش‌های خوشه‌بندی موجود از جمله DBSCAN نیاز به پارامترهای ورودی دارند و انتخاب دقیق مقادیر این پارامترها بر روی خروجی الگوریتم بسیار تأثیرگذار است. هر چند که برخی از الگوریتم‌ها کاربر را در انتخاب پارامتر صحیح کمک می‌کنند اما در مجموعه داده‌های با حجم و ابعاد بالا، انتخاب دقیق این پارامترها مشکل است. الگوریتم VDBSCAN و بهبودهای صورت‌گرفته بر روی آن تلاش کرده‌اند که مقادیر این پارامترها را به صورت خودکار تعیین کنند و تا حدی هم در این



شکل ۲: خوشه‌های با چگالی متفاوت.

کار موفق بوده‌اند اما با پیاده‌سازی و انجام آزمایش‌های متعدد بر روی مجموعه داده‌های مختلف، دریافتیم که الگوریتم VDBSCAN تنها بر روی مجموعه داده‌هایی قادر به تشخیص پارامترها است که فاقد منحنی ملایم در k-dist plot مربوطه‌شان باشند.

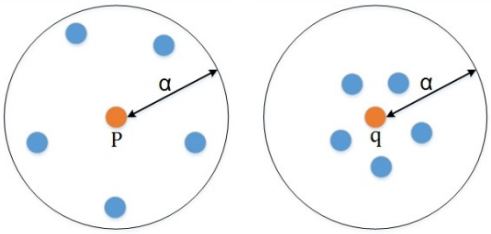
یکی از خصوصیات مهم مجموعه داده‌های دنیای واقعی این است که خوشه‌های موجود در این مجموعه داده‌ها به دلیل وجود چگالی‌های محلی متفاوت، تنها با یک تنظیم پارامتر سراسری قابل تشخیص نیستند و بنابراین ما نیاز به بیش از یک چگالی محلی برای تشخیص خوشه‌ها داریم. به عنوان نمونه در مجموعه داده‌ای که در شکل ۲ نشان داده شده است، خوشه‌های A، B_1، B_2 و B_3 تنها با یک تنظیم پارامتر سراسری قابل تشخیص نیستند. اگر پارامترها را مطابق با چگالی محلی خوشه‌های B_1، B_2 و B_3 تنظیم کنیم خوشه A به عنوان نویز محسوب می‌شود. اگر پارامترها را مطابق با چگالی محلی خوشه A تنظیم کنیم خوشه‌های B_1، B_2 و B_3 به اشتباه با هم ترکیب می‌شوند. بنابراین با یک تنظیم پارامتر سراسری نمی‌توان خوشه‌ها را به درستی تشخیص داد.

یکی از مشکلاتی که الگوریتم DBSCAN در مواجهه با مجموعه داده‌های بزرگ ممکن است با آن رو به رو شود بحث پیچیدگی زمانی بالای این الگوریتم است. الگوریتم DBSCAN به ازای همه نقاط موجود در پایگاه داده عمل پرس و جوی ناحیه‌ای را انجام می‌دهد. در پایگاه داده‌های بزرگ، زمان انجام این عمل قابل توجه خواهد بود و در نتیجه کارایی الگوریتم تنزل می‌یابد. همچنین الگوریتم DBSCAN زمانی که خوشه‌ها نزدیک به هم باشند ممکن است در تشخیص صحیح خوشه‌ها دچار مشکل شود.

۴- الگوریتم پیشنهادی

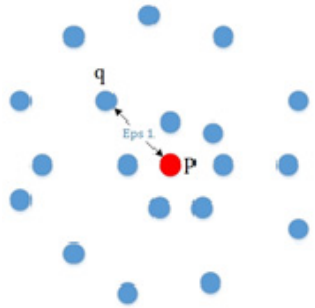
۴-۱ معرفی الگوریتم پیشنهادی

همان گونه که در بخش قبلی بیان شد، یکی از مشکلات الگوریتم DBSCAN عدم پشتیبانی از تغییرات چگالی داخل خوشه‌ها است. برای غلبه بر این مسئله، الگوریتم پیشنهادی ابتدا با استفاده از تکنیکی مقادیر مختلف پارامتر Eps را محاسبه می‌کند و سپس الگوریتم DBSCAN جهت اعمال بر روی مجموعه داده با پارامترهای به دست آمده تطبیق داده می‌شود. الگوریتم پیشنهادی بر مبنای مفهوم چگالی محلی نقاط کار می‌کند. چگالی یک نقطه می‌تواند از طریق شمردن تعداد نقاط موجود در یک شعاع مشخص از آن نقطه محاسبه شود اما این روش تقریب خوبی از چگالی نقاط را به ما نمی‌دهد. شکل ۳ را در نظر بگیرید. همان گونه که در شکل مشاهده می‌کنید در صورتی که ما تعداد نقاط موجود در شعاع α را به عنوان چگالی نقاط در نظر بگیریم، دو نقطه p و q دارای چگالی مشابه هستند اما همان طور که ملاحظه می‌کنید نقطه q دارای تراکم بالاتری است. بنابراین استفاده از این روش تقریب خوبی از چگالی

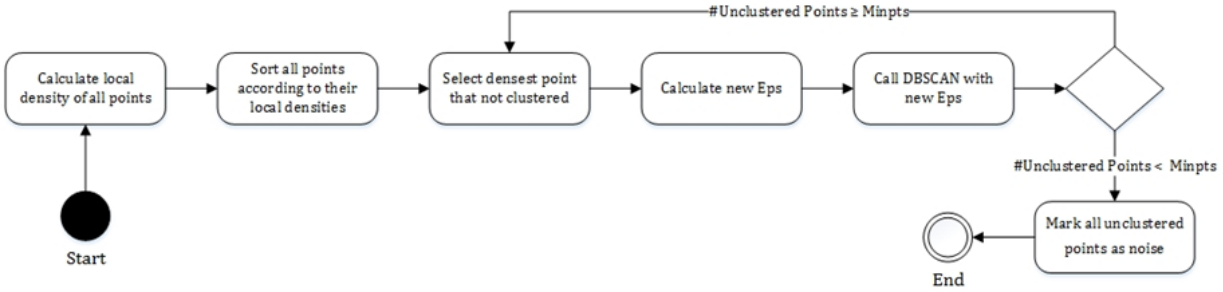


(ب) (الف)

شکل ۳: چگالی مبتنی بر شمردن نقاط موجود در شعاع α.



شکل ۴: فاصله شیء P تا iامین نزدیک‌ترین همسایه‌اش به عنوان Eps_i در نظر گرفته می‌شود.



شکل ۵: مراحل اصلی الگوریتم پیشنهادی.

نقاط را به ما نمی‌دهد. این در حالی است که روش دقیق‌تر، روشی است که چگالی نقاط را بر اساس فاصله نقاط از همسایه‌هایشان محاسبه کند.

تعریف ۹: تابع چگالی محلی: با فرض داشتن مجموعه داده D، چگالی محلی شیء $x \in D$ از طریق محاسبه مجموع فاصله اقلیدسی شیء x از L نزدیک‌ترین همسایه آن به دست می‌آید یعنی

$$Local\_Density(x) = \sum_{i=1}^{L} d(x, x_i) \qquad (۳)$$

که در آن x_i معادل با iامین نزدیک‌ترین همسایه از شیء x است و همچنین $d(x, x_i)$ فاصله اقلیدسی بین دو شیء را برمی‌گرداند. در این تعریف، تصمیم‌گیری در مورد مقدار L بسیار مهم است به گونه‌ای که انتخاب نادرست آن منجر به تنزل دقت نتایج خوشه‌بندی می‌شود. به دو دلیل مقدار L را نمی‌توانیم بزرگ در نظر بگیریم. اول این که همان گونه که در قسمت ارزیابی نشان داده شده است در نظر گرفتن مقادیر بزرگ برای L تقریب مناسبی از چگالی نقاط را به ما نمی‌دهد. همچنین با توجه به این که بخش غالب پیچیدگی زمانی الگوریتم پیشنهادی مربوط به محاسبه چگالی محلی نقاط است، هرچه مقدار L بزرگ باشد پیچیدگی زمانی الگوریتم نیز افزایش می‌یابد به طوری که به ازای $L = n$ پیچیدگی زمانی الگوریتم از مرتبه $O(n^2 \log n)$ می‌شود.

علاوه بر پارامتر L، الگوریتم پیشنهادی نیاز به دو پارامتر ورودی $Minpts$ و k نیز دارد. پارامتر $Minpts$ حداقل تعداد نقاط موجود در یک خوشه را مشخص می‌کند و پارامتر k نیز برای محاسبه مقادیر Eps مورد استفاده قرار می‌گیرد. روال کار الگوریتم پیشنهادی به این صورت است که در ابتدا چگالی محلی همه نقاط را طبق تعریف ۹ محاسبه و سپس نقاط را بر اساس چگالی محلیشان به صورت نزولی مرتب می‌کنیم. دقت داشته باشید که نقطه با مقدار چگالی محلی کمتر از چگالی بیشتری برخوردار است. سپس از بین مجموعه نقاطی که هنوز خوشه‌بندی نشده‌اند متراکم‌ترین نقطه (مانند p) را انتخاب می‌کنیم. هدف ما این است که خوشه با تراکم بالاتر را زودتر تشخیص دهیم.

در واقع با توجه به این که در مجموعه داده‌های با چگالی متفاوت برای هر چگالی یک مقدار Eps متفاوت وجود دارد، در صورتی که ما کار را از خوشه‌های با چگالی کمتر آغاز کنیم، یعنی خوشه‌هایی که مقدار Eps آنها بزرگ است، خوشه‌های با چگالی بالا نیز ممکن است به اشتباه با خوشه‌های چگالی پایین ترکیب شوند. به عبارت دیگر خوشه‌های با چگالی کمتر خوشه‌های با چگالی بالاتر از خودشان را نیز شامل می‌شوند. بنابراین ما همیشه از متراکم‌ترین نقاط کار را آغاز می‌کنیم تا خوشه‌های با چگالی بالاتر زودتر تشخیص داده شوند. سپس با صرف نظر کردن از این نقاط که خوشه‌بندی شده‌اند از چند بار خوشه‌بندی شدن یک نقطه طی تکرارهای بعدی جلوگیری به عمل می‌آوریم. در تکرار iام بعد از انتخاب متراکم‌ترین نقطه p از بین نقاطی که هنوز خوشه‌بندی نشده‌اند، فاصله تا kامین نزدیک‌ترین همسایه از p را به عنوان Eps_i در نظر می‌گیریم (شکل ۴).

بعد از آن الگوریتم DBSCAN را با پارامتر $Minpts$ که از ورودی گرفته‌ایم و پارامتر Eps_i که در مرحله قبل محاسبه شده است فراخوانی می‌کنیم. بعد از پایان کار الگوریتم DBSCAN و صرف نظر کردن از نقاطی که خوشه‌بندی شده‌اند از طریق برچسب خوشه‌ای که به نقاط می‌دهیم، طی یک فرایند تکراری، از بین مجموعه نقاطی که هنوز خوشه‌بندی نشده‌اند متراکم‌ترین نقطه را انتخاب می‌کنیم و فاصله آن نقطه تا kامین نزدیک‌ترین همسایه‌اش را به عنوان پارامتر Eps_{i+1} در نظر می‌گیریم و الگوریتم DBSCAN را با پارامتر جدید Eps_{i+1} فراخوانی می‌کنیم. این فرایند تا زمانی ادامه دارد که همه نقاط خوشه‌بندی شوند یا تعداد نقاط خوشه‌بندی نشده کمتر از مقدار $Minpts$ باشند. اگر تعداد نقاط خوشه‌بندی نشده باقیمانده کمتر از $Minpts$ شود، این نقاط برچسب نویز خواهند گرفت چرا که در این حالت دیگر امکان تشکیل خوشه جدید نیست. مراحل اصلی الگوریتم پیشنهادی به طور خلاصه در شکل ۵ نشان داده شده است.

فرض کنید r تعداد نقاط باقیمانده از مجموعه داده برای خوشه‌بندی بعد از فراخوانی چندین باره الگوریتم DBSCAN باشد. زمانی که $Minpts \le r < k$ باشد ما فاصله تا rامین نزدیک‌ترین همسایه از

Algorithm *Proposed algorithm*
Input: $D, Minpts, k, L$
Output: Set of clusters
1: **begin**
2: $i = 0$
3: **for** (each) point p in the dataset D **do**
4: Calculate local density for p
5: **end for**
6: Sort all points according to their local densities
7: **While** there is an unclustered Point and NumOfUnclusteredPoints ($Result$) >= $Minpts$ **do**
8: Select the point p that has the highest density value and that is not yet clustered
9: Eps_i = Distance ($p, KNN(p)$)
10: Result = Call DBSCAN ($D, Minpts, Eps_i$)
11: **for** each point p in Result **do**
12: **if** p.noise = True **then**
13: Mark p as Unvisited
14: **end if**
15: **end for**
16: $i = i + 1$
17: **end while**
18: **end**

شکل ۶ شبه‌کد الگوریتم پیشنهادی.

متراکم‌ترین نقطه خوشه‌بندی نشده را به عنوان Eps جدید در نظر می‌گیریم. حال اگر $r < Minpts$ باشد نقاط باقیمانده به عنوان نویز برچسب می‌خورند. شبه‌کد الگوریتم پیشنهادی در شکل ۶ آمده است.

در شبه‌کد الگوریتم پیشنهادی، بعد از فراخوانی الگوریتم DBSCAN نقاط با چگالی پایین‌تر برچسب نویز می‌گیرند و این در حالی است که ممکن است این نقاط طی فراخوانی‌های بعدی الگوریتم DBSCAN برچسب خوشه بگیرند. از این رو برای این که نتایج خوشه‌بندی دقیقی داشته باشیم نقاطی که بعد از هر فراخوانی الگوریتم DBSCAN برچسب نویز می‌گیرند برای این که مجدداً طی فراخوانی‌های بعدی الگوریتم DBSCAN مورد ارزیابی قرار بگیرند، برچسب "Uuvisited" می‌گیرند و بنابراین نقاط نویز ممکن است طی فراخوانی‌های بعدی الگوریتم DBSCAN برچسب خوشه بگیرند.

الگوریتم پیشنهادی با استفاده از شاخص مکانی KD-Tree پیاده‌سازی شده است. تابع KNN با گرفتن یک نقطه به عنوان ورودی، kامین نزدیک‌ترین همسایه آن نقطه را برمی‌گرداند. تابع Distance نیز فاصله بین دو شیء ورودی‌اش را محاسبه می‌کند. همان طور که قبلاً نیز اشاره کردیم تعیین پارامترهای الگوریتم پیشنهادی نسبت به سایر الگوریتم‌ها ساده‌تر است. پارامتر $Minpts$ که حداقل تعداد نقاط یک خوشه را مشخص می‌کند، بسته به مجموعه داده مورد بررسی قابل تعیین است. بدیهی است که پارامتر k حداقل باید به اندازه $Minpts$ باشد چرا که در غیر این صورت ممکن است خوشه‌ای یافت نشود. به عبارت دیگر، پارامتر k باید به گونه‌ای باشد که در فاصله بین نقطه مورد بررسی و kامین نزدیک‌ترین همسایه‌اش یک خوشه تشکیل شود یعنی حداقل $Minpts$ نقطه در این فاصله قرار گرفته باشد. بنابراین اگر پارامتر k حداقل به اندازه $Minpts$ باشد این شرط برقرار خواهد بود. در رابطه با پارامتر L نیز همان گونه که در آزمایش‌ها نشان داده شده است می‌توانیم این پارامتر را با مقدار ثابت $Minpts$ تنظیم کنیم.

۴-۲ ویژگی‌ها و کاربرد الگوریتم پیشنهادی

به طور کلی الگوریتم پیشنهادی دارای ویژگی‌های زیر است:

۱) قابلیت تشخیص خوشه‌های با چگالی متفاوت

۲) قابلیت کشف خوشه‌های با اندازه و اشکال متفاوت

۳) مقاوم‌بودن در مقابل نویز

۴) ساده‌تر بودن تعیین مقدار دقیق پارامترهای ورودی (پارامتر Eps به صورت خودکار تعیین می‌شود): در رابطه با پارامترهای ورودی الگوریتم ارائه‌شده چندین نکته وجود دارد. به طور کلی الگوریتم ارائه‌شده با تعیین خودکار پارامتر Eps، حساسیت به پارامترهای ورودی را کاهش داده است. اما چگونه؟ در نگاه اول شاید به نظر برسد که الگوریتم ۳ پارامتر دارد و این چندان مناسب نباشد اما همان گونه که در بخش ارزیابی نشان داده شده است بهترین مقدار برای پارامتر L همان مقدار $Minpts$ است. یعنی برای داشتن نتایج ایده‌آل، مقدار پارامتر L را برابر با مقدار $Minpts$ قرار داده و از این پارامتر صرف نظر می‌کنیم. بنابراین برای این الگوریتم تنها تعیین مقدار دو پارامتر $Minpts$ و k کافی است و تنها برای فهم بهتر و دقیق الگوریتم است که پارامتر L به صورت یک پارامتر جداگانه در نظر گرفته شده است. در رابطه با پارامتر k نیز باید بگوییم که اولاً همان گونه که قبلاً نیز اشاره کردیم، حد پایین این پارامتر مشخص است و این پارامتر باید حداقل به اندازه $Minpts$ باشد، ثانیاً مقدار این پارامتر در یک طیف وسیع قابل تغییر است (بر خلاف پارامتر Eps) بدون این که در نتایج الگوریتم تغییری حاصل شود و این یعنی کاهش حساسیت به پارامتر ورودی. این کاهش حساسیت باعث انتخاب سریع‌تر و ساده‌تر مقدار دقیق پارامترها می‌شود به گونه‌ای که ما در ارزیابی‌های خود تنها با تعداد معدودی تست بر روی مجموعه داده‌ها به نتایج ایده‌آل خود رسیده‌ایم. بنابراین انتخاب مقدار پارامترها مشکل نیست.

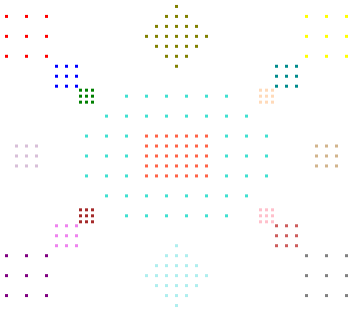
۵) سادگی و قابل فهم بودن: یکی از ویژگی‌های بارز الگوریتم DBSCAN سادگی و قابل فهم بودن آن است. در الگوریتم جدید ارائه‌شده این سادگی و قابل فهم بودن به خوبی رعایت شده است به گونه‌ای که ساختار کلی الگوریتم ارائه‌شده دقیقاً مشابه ساختار الگوریتم DBSCAN است.

۶) قابلیت تشخیص خوشه‌های تو در تو و چسبیده به هم

۷) دقت بالا و درصد خطای پایین نسبت به سایر الگوریتم‌های هم‌رده الگوریتم پیشنهادی

یکی از کاربردهای الگوریتم پیشنهادی، خوشه‌بندی تصاویر دریافتی از ماهواره‌ها است. هر روز حجم بسیار عظیمی از داده‌ها در قالب تصاویر از ماهواره‌ها دریافت می‌شود که این داده‌ها بایستی به اطلاعات قابل فهم تبدیل شوند. خوشه‌بندی نواحی موجود در تصاویر بر اساس رودخانه‌ها، کوه‌ها، جنگل‌ها و جاده‌ها نمونه‌هایی از این تبدیل اطلاعات هستند. توجه داشته باشید که تصاویر دریافتی از ماهواره مستقیماً نمی‌توانند خوشه‌بندی شوند. در واقع، قبل از خوشه‌بندی این اطلاعات توسط الگوریتم‌های خوشه‌بندی، لازم است که عمل پردازش تصویر روی این تصاویر انجام شود. بعد از انجام عمل پردازش تصویر، داده‌ها در قالب داده‌های مکانی نمایش داده می‌شوند و سپس الگوریتم خوشه‌بندی می‌تواند بر روی این داده‌ها اعمال شود. از جمله کاربردهای دیگر الگوریتم پیشنهادی در تشخیص ناهنجاری‌ها و داده‌های دورافتاده[1] در مجموعه داده‌ها است که در بخش ۵ بیشتر در این مورد توضیح داده خواهد شد.

1. Outliers



شکل ۷: نتیجه اعمال الگوریتم پیشنهادی بر روی مجموعه داده مصنوعی DS۳ با پارامترهای $Minpts = ۶$ و $k = ۷$.

۴-۳ پیچیدگی زمانی الگوریتم پیشنهادی

قبل از شروع کار الگوریتم لازم است که یک ساختار شاخص بر روی مجموعه داده برای داشتن پرس و جوهای بهینه ساخته شود و ما از ساختار شاخص KD-Tree استفاده کرده‌ایم. زمان ساخت این شاخص مکانی از مرتبه $O(d \times n \log n)$ است که در آن d تعداد ابعاد داده‌ها است.

الگوریتم ابتدا چگالی محلی همه نقاط را محاسبه می‌کند و برای این منظور لازم است که فاصله هر نقطه از L نزدیک‌ترین همسایه‌اش را با هم جمع کنیم. توجه داشته باشید که ما برای پاسخ پرس و جوهای ناحیه‌ای الگوریتم DBSCAN از ساختار شاخص KD-Tree استفاده کرده‌ایم. حال اگر برای به دست آوردن L نزدیک‌ترین همسایه هر نقطه نیز از ساختار شاخص KD-Tree استفاده کنیم، زمان مورد نیاز برای یافتن L نزدیک‌ترین همسایه همه نقاط (برای ابعاد پایین) $O(L \times n \log n)$ است. در اینجا لازم است به این نکته اشاره کنیم که یافتن نزدیک‌ترین همسایه برای هر نقطه با استفاده از ساختار شاخص KD-Tree می‌تواند در زمان $O(\log n)$ انجام شود [۳۷]. بعد از محاسبه چگالی محلی نقاط مجموعه داده باید نقاط را بر اساس چگالی محلیشان مرتب کرد. برای مرتب‌سازی از الگوریتم Merge Sort استفاده می‌کنیم که نیاز به زمان $O(n \log n)$ دارد. بعد از مرتب‌سازی نقاط، نوبت به محاسبه مقدار Eps است. برای به دست آوردن شعاع Eps مربوط به هر چگالی نیاز به محاسبه فاصله متراکم‌ترین نقطه از kامین نزدیک‌ترین همسایه‌اش داریم. یافتن kامین نزدیک‌ترین همسایه یک نقطه می‌تواند در زمان $O(n \log n)$ انجام شود. حالا نوبت به انجام عمل خوشه‌بندی با استفاده از الگوریتم DBSCAN است. الگوریتم DBSCAN به تعداد چگالی‌های موجود در مجموعه داده فراخوانی می‌شود. در هر فراخوانی الگوریتم DBSCAN، این الگوریتم برای همه نقاط باقیمانده از مجموعه داده پرس و جوی ناحیه‌ای انجام می‌دهد، از این رو نیاز به زمان $O(n \log n)$ دارد. به طور کلی عمل خوشه‌بندی توسط فراخوانی‌های چندین باره الگوریتم DBSCAN حداکثر از مرتبه $O(m \times n \log n)$ است که در آن m معادل با تعداد چگالی‌های موجود در مجموعه داده است و بنابراین در کل زمان انجام عمل خوشه‌بندی توسط الگوریتم پیشنهادی عبارت است از

$$O(dn \log n) + O(L \times n \log n) + O(n \log n) + O(n \log n) + O(mn \log n) = O(\max < d, m, L > n \log n) \quad (۴)$$

که در آن m معادل با تعداد چگالی‌های موجود در مجموعه داده و L نیز پارامتر مورد استفاده برای به دست آوردن چگالی محلی نقاط است. اگر فرض کنیم که d، m و L ثابت باشند آن گاه می‌توانیم بگوییم که

جدول ۱: اطلاعات مجموعه داده‌های مورد استفاده.

نام مجموعه داده	نوع مجموعه داده	تعداد اشیا	تعداد خوشه	چند چگالی
Aggregation	Standard	۷۸۸	۷	خیر
Jain	Standard	۳۷۳	۲	بله
Can۴۷۳	Standard	۴۷۳	۸	بله
Spiral	Standard	۳۱۲	۳	خیر
Path-Based	Standard	۳۰۰	۳	خیر
Compound	Standard	۳۹۹	۶	بله
D۳۱	Standard	۳۱۰۰	۳۱	خیر
Flame	Standard	۲۴۰	۲	خیر
R۱۵	Standard	۶۰۰	۱۵	خیر
DS۳	Artificial	۲۵۷	۱۸	بله
DS۲	Artificial	۲۶۷۱	۱۰	بله
DS۱	Artificial	۱۰۲۰۸	۹	بله

پیچیدگی الگوریتم پیشنهادی از مرتبه $O(n \log n)$ است.

۵- ارزیابی

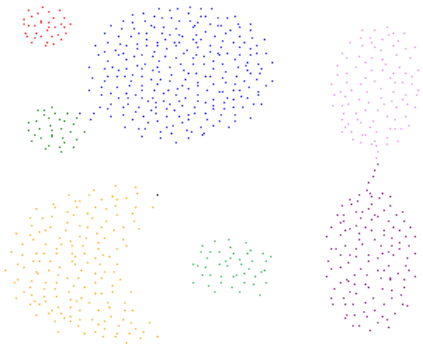
برای ارزیابی، ما علاوه بر الگوریتم پیشنهادی، الگوریتم DBSCAN و پنج الگوریتم VDBSCAN، LDBSCAN، DVBSCAN، MDDBSCAN و VMDBSCAN را که جهت رفع مشکل تغییرات چگالی DBSCAN ارائه شده‌اند نیز با استفاده از زبان برنامه‌نویسی سی شارپ در محیط ویندوز پیاده‌سازی کرده‌ایم. کامپیوتر مورد استفاده برای ارزیابی دارای پردازنده Core i۵ و ۴ گیگابایت حافظه داخلی است و همه این الگوریتم‌ها با استفاده از شاخص مکانی KD-Tree پیاده‌سازی شده‌اند و همچنین در همه ارزیابی‌ها پارامتر L با مقدار $Minpts$ تنظیم شده است. جهت ارزیابی از مجموعه داده‌های استاندارد [۳۸] Aggregation، Can۴۷۳، Jain، Spiral، Path-Based، Compound، D۳۱، Flame، R۱۵ و سه مجموعه داده مصنوعی DS۱، DS۲ و DS۳ استفاده گردیده و در جدول ۱ جزئیات مربوط به مجموعه داده‌های مورد استفاده نشان داده شده است.

در شکل ۷ خروجی الگوریتم پیشنهادی بر روی مجموعه داده مصنوعی DS۳ را مشاهده می‌کنید. همان طور که می‌بینید الگوریتم پیشنهادی به خوبی توانسته خوشه‌های چسبیده به هم، تو در تو و خوشه‌های با چگالی متفاوت را تشخیص دهد. خروجی الگوریتم پیشنهادی بر روی سایر مجموعه داده‌ها نیز در شکل‌های ۸ تا ۱۵ آمده است. دقت داشته باشید که خوشه‌ها با رنگ‌های مختلف از هم جدا شده‌اند.

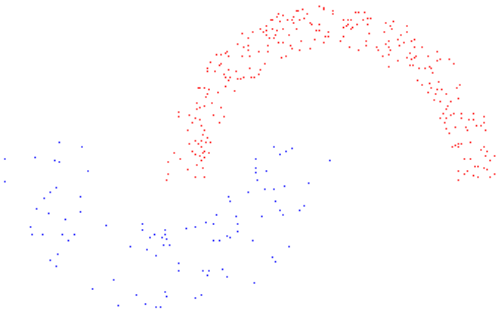
به منظور مقایسه الگوریتم پیشنهادی با الگوریتم‌های دیگر از ۳ معیار Rand [۳۹]، Jaccard [۴۰] و Fowlkes-Mallows [۴۱] که از جمله معیارهای ارزیابی روش‌های خوشه‌بندی هستند و معیار درصد خطای خوشه‌بندی استفاده کرده‌ایم. سه معیار اول میزان شباهت خوشه‌های حاصل از الگوریتم‌های مورد ارزیابی با خوشه‌های برچسب‌دار را محاسبه می‌کنند. این معیارها از فرمول‌های زیر به دست می‌آیند

$$Rand\_Measure = \frac{TP + TN}{TP + FP + FN + TN} \quad (۵)$$

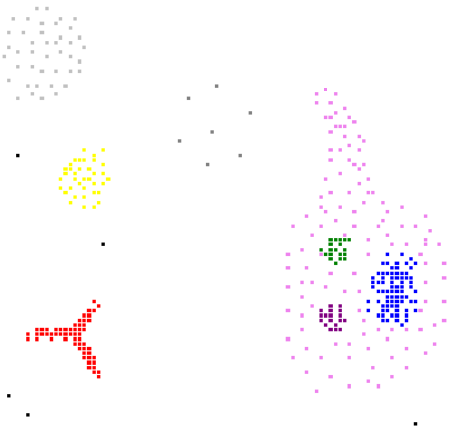
$$Jaccard\_Index = \frac{|A \cap B|}{|A \cup B|} = \frac{TP}{TP + FP + FN} \quad (۶)$$



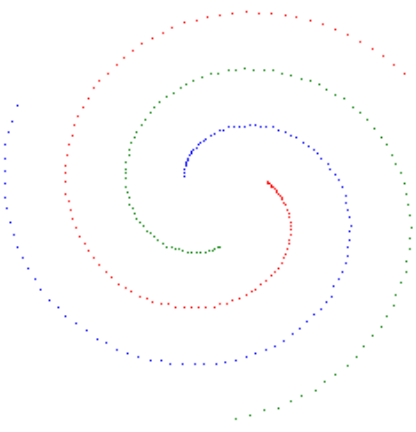
شکل ۸: نتیجه اعمال الگوریتم پیشنهادی بر روی مجموعه داده Aggregation با پارامترهای $Minpts = ۱۲$ و $k = ۳۱$.



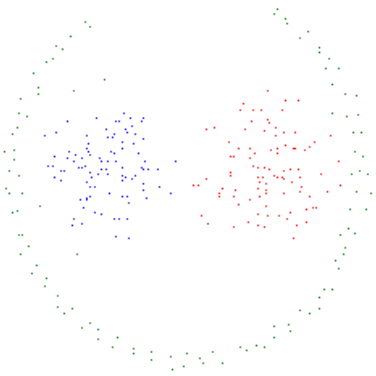
شکل ۹: نتیجه اعمال الگوریتم پیشنهادی بر روی مجموعه داده Jain با پارامترهای $Minpts = ۱۴$ و $k = ۲۶$.



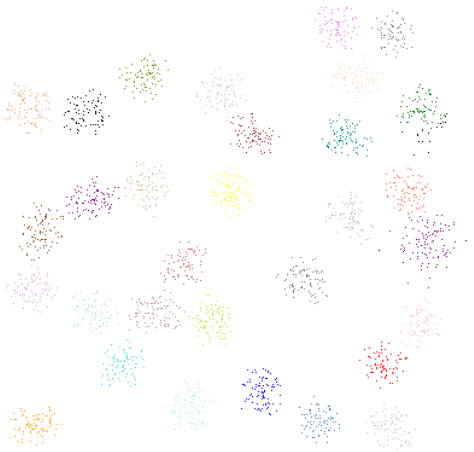
شکل ۱۰: نتیجه اعمال الگوریتم پیشنهادی بر روی مجموعه داده Can۴۷۳ با پارامترهای $Minpts = ۱۱$ و $k = ۳۴$.



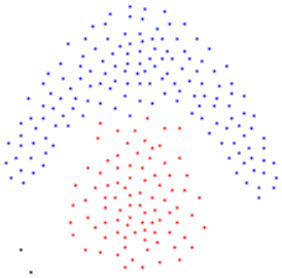
شکل ۱۱: نتیجه اعمال الگوریتم پیشنهادی بر روی مجموعه داده Spiral با پارامترهای $Minpts = ۷$ و $k = ۱۹$.



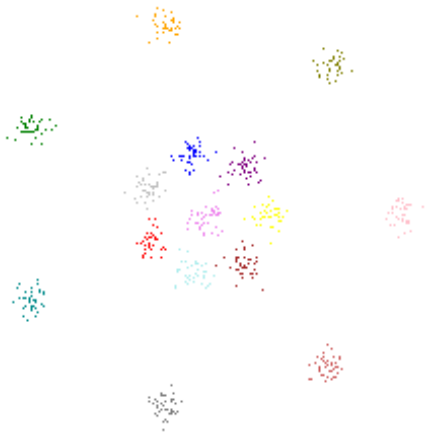
شکل ۱۲: نتیجه اعمال الگوریتم پیشنهادی بر روی مجموعه داده PathBased با پارامترهای $Minpts = ۱۰$ و $k = ۲۴$.



شکل ۱۳: نتیجه اعمال الگوریتم پیشنهادی بر روی مجموعه داده D۳۱ با پارامترهای $Minpts = ۱۷$ و $k = ۲۰$.



شکل ۱۴: نتیجه اعمال الگوریتم پیشنهادی بر روی مجموعه داده Flame با پارامترهای $Minpts = ۱۰$ و $k = ۱۹$.



شکل ۱۵: نتیجه اعمال الگوریتم پیشنهادی بر روی مجموعه داده R۱۵ با پارامترهای $Minpts = ۱۸$ و $k = ۲۹$.

جدول ۲: نتایج ارزیابی با توجه به معیار RAND.

Algorithms \ Datasets	Can۴۷۳	Jain	Agg	Path-B	DS۱	DS۲
LDBSCAN	۰٫۸۷۶	۰٫۹۶۷	۰٫۹۶۴	۰٫۹۲۵	۰٫۹۷	۰٫۹۸
VDBSCAN	۰٫۷۹۲	۰٫۶۱۴	۰٫۹۲۶	۰٫۷۵۲	۰٫۶۳	۰٫۸۱
DVBSCAN	۰٫۹۰	۰٫۹۵۳	۰٫۹۹۴	۰٫۷۵۹	۰٫۸۵	۰٫۹۷
VMDBSCAN	۰٫۷۹	۰٫۶۲۴	۰٫۷۰۸	۰٫۸۷۵	۰٫۷۹	۰٫۹۴
MDDBSCAN	۰٫۹۸	۱	۰٫۹۳۸	۰٫۹۱۲	۰٫۹۸	۱
Proposed Alg	۰٫۹۸۸	۱	۰٫۹۹۸	۰٫۹۶	۰٫۹۶	۱

جدول ۳: نتایج ارزیابی با توجه به معیار JACCARD.

Algorithms \ Datasets	Can۴۷۳	Jain	Agg	Path-B	DS۱	DS۲
LDBSCAN	۰٫۵۹۸	۰٫۹۴۷	۰٫۸۵۷	۰٫۷۸۱	۰٫۹۱	۰٫۹۲
VDBSCAN	۰٫۴۷۹	۰٫۶۱۴	۰٫۶۴۶	۰٫۴۵۶	۰٫۵۴	۰٫۶۸
DVBSCAN	۰٫۵۴	۰٫۹۲۴	۰٫۹۷۵	۰٫۵۷۲	۰٫۷۵۲	۰٫۸۵
VMDBSCAN	۰٫۴۶	۰٫۶۱۸	۰٫۴۲۵	۰٫۶۸۶	۰٫۷۳	۰٫۸۱
MDDBSCAN	۰٫۹۱	۱	۰٫۷۲	۰٫۷۳۷	۰٫۹۴	۱
Proposed Alg	۰٫۹۴	۱	۰٫۹۹۲	۰٫۸۸۸	۰٫۹۲	۱

جدول ۴: نتایج ارزیابی با توجه به معیار FOWLKES MALLOWS.

Algorithms \ Datasets	Can۴۷۳	Jain	Agg	Path-B	DS۱	DS۲
LDBSCAN	۰٫۷۶۷	۰٫۹۷۳	۰٫۹۲۴	۰٫۸۸	۰٫۹۴	۰٫۹۸
VDBSCAN	۰٫۶۹۲	۰٫۷۸۳	۰٫۸۶۳	۰٫۶۲۷	۰٫۶۲۵	۰٫۷۹
DVBSCAN	۰٫۷۱	۰٫۹۶۱	۰٫۹۸۷	۰٫۷۵۱	۰٫۸۴	۰٫۹۵
VMDBSCAN	۰٫۶۶	۰٫۷۸۵	۰٫۶۵۲	۰٫۸۱۳	۰٫۷۶	۰٫۹۱
MDDBSCAN	۰٫۹۵	۱	۰٫۸۴۷	۰٫۸۵۷	۰٫۹۶	۱
Proposed Alg	۰٫۹۶۹	۱	۰٫۹۹۶	۰٫۹۱	۰٫۹۴	۱

$$Fowlkes\_Mallows\_Index = \sqrt{\frac{TP}{TP+FP}\cdot\frac{TP}{TP+FN}} \quad (۷)$$

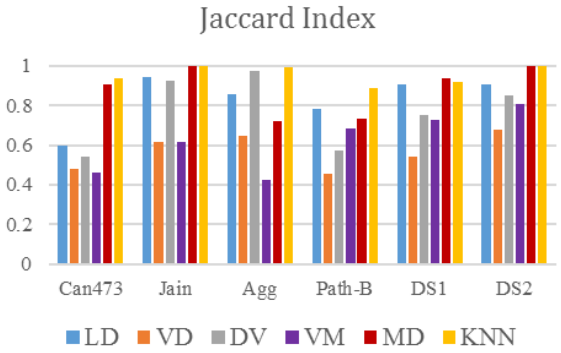
همه این فرمول‌ها بر مبنای روش‌های بیان خطای FP[1]، FN[2] و TP[3] و TN[4] بیان شده‌اند. با برداشتن ۲ شیء از مجموعه داده، TP زمانی برقرار است که این ۲ شیء واقعاً متعلق به یک خوشه باشند (بر طبق مجموعه داده برچسب خورده) و الگوریتم مورد ارزیابی نیز آن دو شیء را به یک خوشه تخصیص داده باشد. اگر این دو شیء واقعاً متعلق به یک خوشه نباشند و الگوریتم مورد ارزیابی هم آنها را به یک خوشه تخصیص نداده باشد TN رخ داده است. در صورتی که این ۲ شیء واقعاً متعلق به یک خوشه باشند و الگوریتم مورد ارزیابی آنها را به یک خوشه تخصیص نداده باشد، FN رخ داده و نهایتاً در صورتی که این دو شیء واقعاً متعلق به یک خوشه نباشند و الگوریتم مورد ارزیابی آنها را به یک خوشه تخصیص داده باشد FP رخ داده است. پس با چک‌کردن تمامی جفت نقاط مجموعه داده، این ۴ معیار تعیین خطا مشخص می‌شوند.

1. False Positive
2. False Negative
3. True Positive
4. True Negative

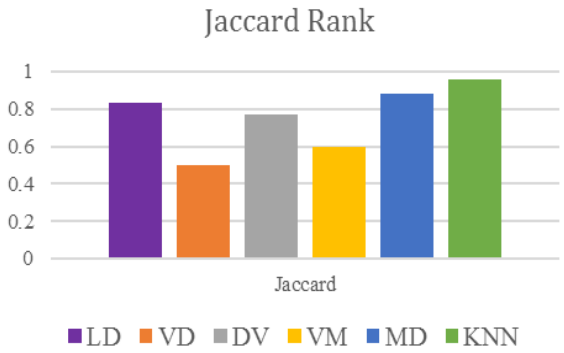
نتایج حاصل از اعمال الگوریتم پیشنهادی و پنج الگوریتم دیگر بر روی ۶ مجموعه داده مورد نظر بر اساس معیارهای Rand، Jaccard و Fowlkes Mallows در جداول ۲ تا ۴ نشان داده شده است. توجه داشته باشید که برای الگوریتم‌های مورد ارزیابی با انجام تست‌های مختلف بر روی مجموعه داده‌ها، بهترین پارامترها تنظیم شده است.

در شکل ۱۶ نتایج به دست آمده با توجه به معیار Jaccard در قالب نمودار نشان داده شده است. همچنین در شکل ۱۷ نیز معیار Jaccard به طور میانگین برای همه الگوریتم‌ها آمده است. همان گونه که در شکل ۱۷ می‌بینید الگوریتم پیشنهادی از میانگین بهتری نسبت به سایر الگوریتم‌ها برخوردار است به این معنی که خوشه‌های حاصل از این الگوریتم شباهت بیشتری به خوشه‌های برچسب‌گذاری شده داشته است.

به منظور بررسی صحت نتایج ارزیابی الگوریتم پیشنهادی و اثبات معنادار بودن آماری بهبود عملکرد به دست آمده، ما از معیار بازه اطمینان (CI) استفاده کرده‌ایم. برای این منظور، ۱۰ زیرمجموعه تصادفی از مجموعه داده‌های اولیه انتخاب نموده و هر بار دقت الگوریتم‌های مورد نظر را به دست می‌آوریم. این زیرمجموعه‌ها برای مجموعه داده‌های Path-Based، Jain، Aggregation و Can۴۷۳ با انتخاب تصادفی به ترتیب ۲۵۰، ۳۰۰، ۶۵۰ و ۳۸۰ نمونه از ۳۰۰، ۳۷۳، ۷۸۸ و ۴۷۳ نمونه موجود به دست می‌آید. بازه اطمینان دقت به دست آمده با درجه اطمینان ۹۵ درصد برای الگوریتم‌ها در جدول ۵ ارائه شده است.



شکل ۱۶: نتایج ارزیابی با توجه به معیار Jaccard



شکل ۱۷: میانگین معیار Jaccard

جدول ۵: بازه اطمینان ۹۵٪ الگوریتم‌های مورد ارزیابی.

Datasets \ Algorithms	LDBSCAN	VDBSCAN	DVBSCAN	VMDBSCAN	MDDBSCAN	Proposed Algorithm
Path-Based	[۰٫۸۸ −۰٫۹۳۷] ۰٫۹۰۸	[۰٫۶۱ −۰٫۷۹] ۰٫۷	[۰٫۶۹ −۰٫۷۸] ۰٫۷۳۵	[۰٫۸۰۵ −۰٫۸۹۲] ۰٫۸۴۸	[۰٫۸۷ −۰٫۹۳] ۰٫۹	[۰٫۹۲ −۰٫۹۷۳] ۰٫۹۴۶
Jain	[۰٫۹۲ −۰٫۹۷۵] ۰٫۹۴۷	[۰٫۵۲ −۰٫۶۹] ۰٫۶۰۵	[۰٫۸۹۷ −۰٫۹۷] ۰٫۹۳۳	[۰٫۶۱ −۰٫۷۰۳] ۰٫۶۵۶	[۰٫۹۵۹ −۱] ۰٫۹۷۹	[۰٫۹۶۷ −۱] ۰٫۹۸۳
Aggregation	[۰٫۹۱ −۰٫۹۸۱] ۰٫۹۴۵	[۰٫۷۲ −۰٫۸۹۳] ۰٫۸۰۶	[۰٫۹۳۴ −۱] ۰٫۹۶۷	[۰٫۶۸۵ −۰٫۸۴] ۰٫۷۶۲	[۰٫۸۵ −۰٫۹۶۴] ۰٫۹۰۷	[۰٫۹۷۱ −۱] ۰٫۹۸۵
Can۴۷۳	[۰٫۸۰ −۰٫۸۹۳] ۰٫۸۴۶	[۰٫۶۶ −۰٫۷۷۴] ۰٫۷۱۷	[۰٫۸۴ −۰٫۹۳] ۰٫۸۸۵	[۰٫۷۴ −۰٫۸۳۵] ۰٫۷۸۷	[۰٫۹۵ −۰٫۹۸۳] ۰٫۹۶۶	[۰٫۹۴ −۰٫۹۹۱] ۰٫۹۶۵

جدول ۶: نتایج ارزیابی با توجه به شاخص درصد خطای خوشه‌بندی.

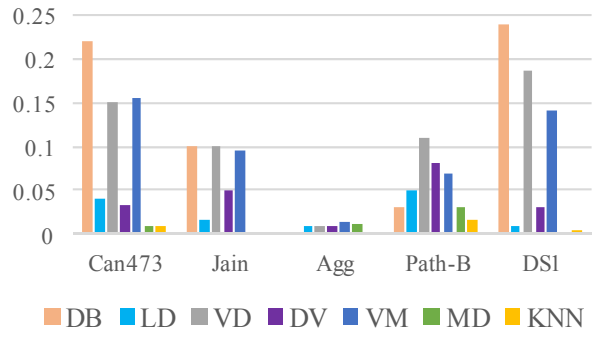
Datasets	Can۴۷۳(۸)		Jain(۲)		Aggregation(۷)		Path-Based(۳)	
Algorithms	#CTR	%err	#CTR	%err	#CTR	%err	#CTR	%err
Proposed Alg	۸	۰٫۰۱	۲	۰	۷	۰٫۰۰۳	۳	۰٫۱۶
DBSCAN	۶	۰٫۲۲	۲	۰٫۱	۷	۰٫۰۰۳	۳	۰٫۰۳۳
LDBSCAN	۸	۰٫۰۴	۲	۰٫۰۱۶	۷	۰٫۰۰۹	۳	۰٫۰۵
VDBSCAN	۶	۰٫۱۵	۳	۰٫۱	۶	۰٫۰۱	۳	۰٫۱۱
DVBSCAN	۸	۰٫۰۳۳	۳	۰٫۰۵	۷	۰٫۰۰۸	۴	۰٫۰۸
VMDBSCAN	۶	۰٫۱۵۶	۲	۰٫۰۹۶	۶	۰٫۰۱۵	۳	۰٫۰۷
MDDBSCAN	۸	۰٫۰۱	۲	۰	۷	۰٫۰۱۲	۳	۰٫۰۳

همان طور که مشخص است دقت الگوریتم VDBSCAN نسبت به الگوریتم‌های دیگر بسیار کمتر بوده و بازه اطمینان بزرگی دارد که نشان از عدم پایداری این الگوریتم می‌باشد. همچنین میانگین دقت روش پیشنهادی خوب بوده و بازه اطمینان آن به نسبت کم است.

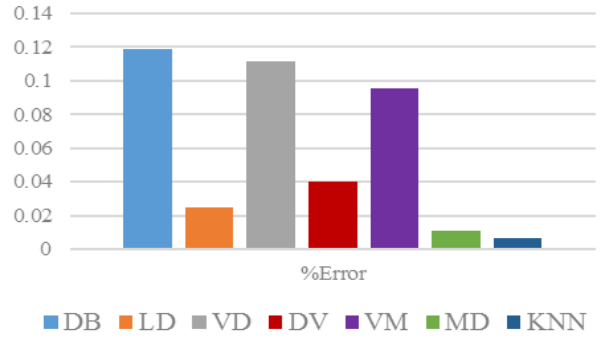
شاخص درصد خطای خوشه‌بندی از تقسیم تعداد نمونه‌های به اشتباه خوشه‌بندی شده بر تعداد کل نمونه‌های مجموعه داده حاصل می‌شود. در جدول ۶ درصد خطا و تعداد خوشه‌های یافت‌شده توسط الگوریتم‌ها و همچنین درصد خطای مربوط به الگوریتم DBSCAN قرار داده شده است. توجه داشته باشید که تعداد خوشه‌های صحیح هر مجموعه داده در داخل پرانتز، مقابل اسم مجموعه داده مشخص شده است.

در شکل ۱۸ درصد خطای مربوط به الگوریتم‌های مختلف در قالب نمودار بیان شده است. همچنین در شکل ۱۹ نمودار میانگین درصد خطای مربوط به الگوریتم‌های مورد ارزیابی را مشاهده می‌کنید. همان طور که در شکل ۱۹ می‌بینید الگوریتم پیشنهادی به طور میانگین از دقت بالاتری نسبت به سایر الگوریتم‌ها برخوردار است. همان گونه که قبلاً نیز اشاره کردیم الگوریتم VDBSCAN تنها در مجموعه داده‌هایی که دارای شیب تند در منحنی k-dist plot متناظرشان هستند پاسخ مناسب تولید می‌کند و از این رو الگوریتم VDBSCAN درصد خطای بالاتری نسبت به سایر الگوریتم‌ها دارد. با توجه به این که الگوریتم DBSCAN قابلیت تشخیص خوشه‌های با چگالی متفاوت را ندارد همان گونه که در شکل ۱۹ نیز مشاهده می‌کنید، درصد خطای این الگوریتم در مجموعه داده‌های با چگالی متفاوت بالا است.

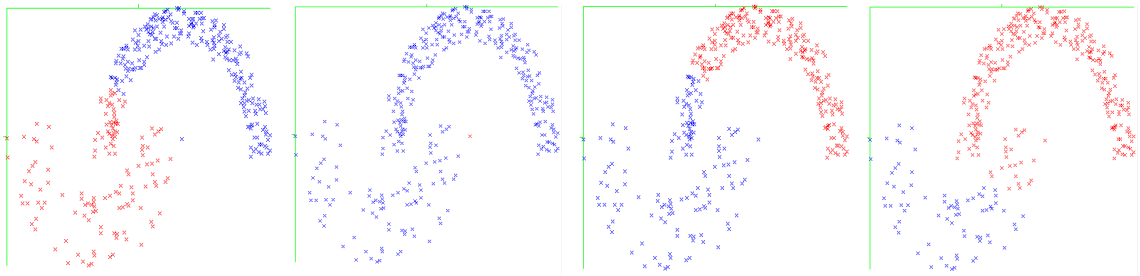
همان طور که دیدیم الگوریتم پیشنهادی که یک الگوریتم خوشه‌بندی مبتنی بر چگالی است بر روی مجموعه داده‌های مورد نظر به خوبی عمل کرده است. حال می‌خواهیم کارایی برخی از الگوریتم‌هایی را که در دسته‌های دیگر از روش‌های خوشه‌بندی قرار دارند بر روی مجموعه داده‌های مورد استفاده برای ارزیابی الگوریتم پیشنهادی ارزیابی کنیم. برای این منظور ۴ الگوریتم K-Means، Single Link، Average Link و Complete Link را انتخاب کرده‌ایم. K-Means [۴۲] یک الگوریتم خوشه‌بندی مبتنی بر پارتیشن است و سه الگوریتم دیگر نیز جزء الگوریتم‌های خوشه‌بندی سلسله‌مراتبی [۴۳] هستند. برای ارزیابی این الگوریتم‌ها از ابزار Weka استفاده شده و نتایج به دست آمده از ارزیابی



شکل ۱۸: درصد خطای الگوریتم‌های مورد ارزیابی.

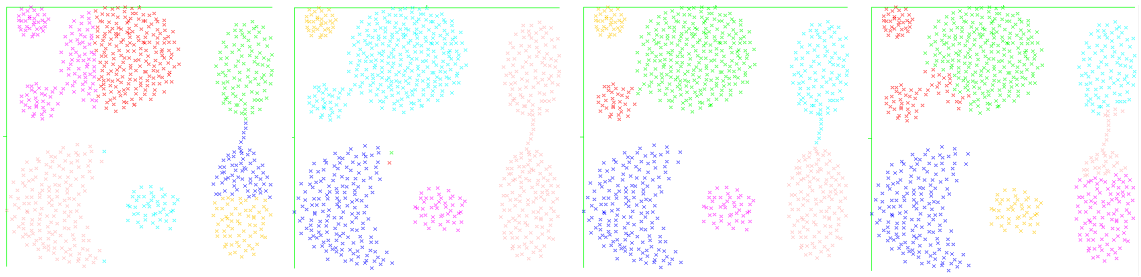


شکل ۱۹: میانگین درصد خطای الگوریتم‌های مورد ارزیابی.



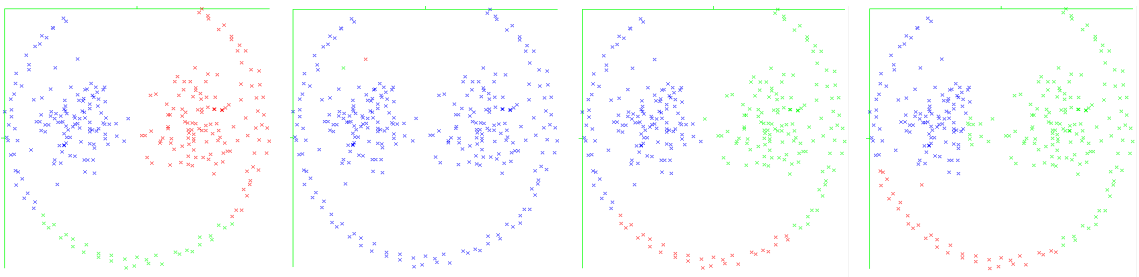
(الف) (ب) (ج) (د)

شکل ۲۰: نتایج اعمال الگوریتم‌های مورد نظر بر روی مجموعه داده Jain، (الف) Complete Link، (ب) Average Link، (ج) Single Link و (د) K-Means، Number of iterations: ۷.



(الف) (ب) (ج) (د)

شکل ۲۱: نتایج اعمال الگوریتم‌های مورد نظر بر روی مجموعه داده Aggregation، (الف) Complete Link، (ب) Average Link، (ج) Single Link و (د) K-Means، Number of iterations: ۱۸.



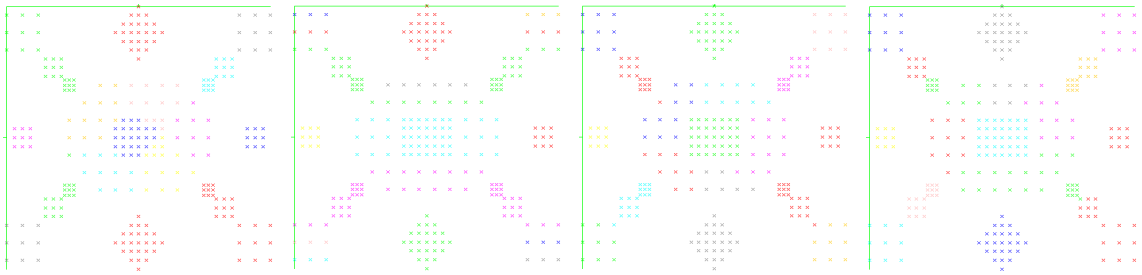
(الف) (ب) (ج) (د)

شکل ۲۲: نتایج اعمال الگوریتم‌های مورد نظر بر روی مجموعه داده Path-Based، (الف) Complete Link، (ب) Average Link، (ج) Single Link و (د) K-Means، Number of iterations: ۹.

این الگوریتم‌ها در ادامه نشان داده خواهد شد (شکل‌های ۲۰ تا ۲۴).

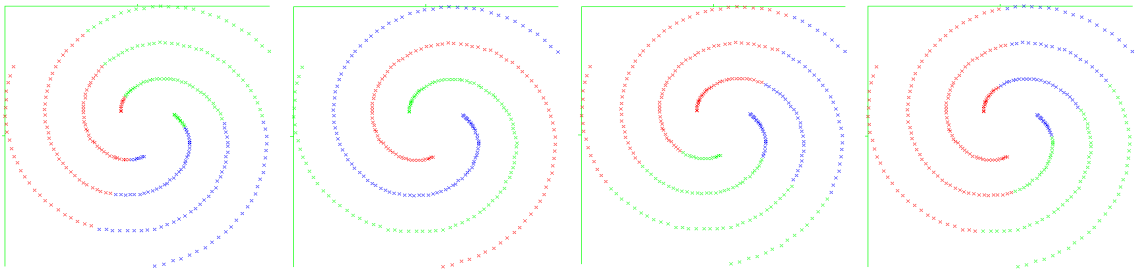
همان طور که دیدید الگوریتم‌های K-Means، Single Link، Average Link و Complete Link نتایج مطلوبی را بر روی مجموعه داده‌های مورد نظر ایجاد نکردند و در اکثر موارد خطای بالایی داشتند. در واقع این الگوریتم‌ها که مبتنی بر پارتیشن و سلسله‌مراتبی هستند در ایجاد نتایج مطلوب بر روی داده‌های مکانی با شکست مواجه شدند و این در حالی است که این الگوریتم‌ها برای کاربردهای اصلی مربوط به خودشان به عنوان الگوریتم مناسب در نظر گرفته می‌شوند.

همان طور که قبلاً نیز اشاره کردیم یکی از کاربردهای الگوریتم پیشنهادی در تشخیص ناهنجاری‌ها یا نقاط دورافتاده در مجموعه داده‌ها است، جایی که نقاط در یک الگوی مورد انتظار شبیه به دیگر موارد موجود در مجموعه داده، مطابقت نمی‌کنند. در مجموعه داده‌های واقعی معمولاً بسیاری از این گونه نقاط غیر قابل استفاده بوده و حاوی اطلاعات مهمی نمی‌باشند. الگوریتم‌های خوشه‌بندی می‌توانند برای فیلترکردن چنین نقاطی مورد استفاده قرار بگیرند. از این رو ما الگوریتم پیشنهادی را بر روی یک مجموعه داده واقعی با بیش از ۳ میلیون نقطه GPS از منطقه سیاتل آمریکا [۴۴] ارزیابی کرده‌ایم. نتایج ارزیابی بر روی این مجموعه داده در شکل ۲۵ آمده است. توجه داشته باشید که ناهنجاری‌های موجود



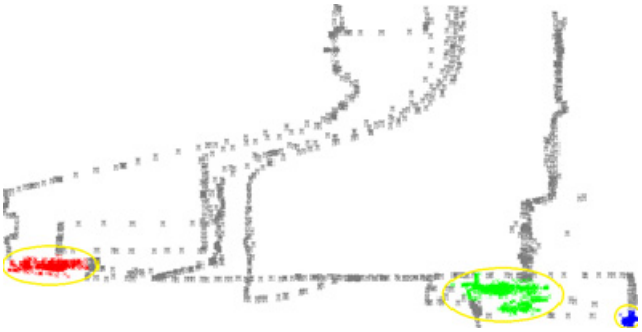
(الف) (ب) (ج) (د)

شکل ۲۳: نتایج اعمال الگوریتم‌های مورد نظر بر روی مجموعه داده DS۳، (الف) Complete Link، (ب) Average Link، (ج) Single Link و (د) K-Means، Number of iterations: ۱۹.



(الف) (ب) (ج) (د)

شکل ۲۴: نتایج اعمال الگوریتم‌های مورد نظر بر روی مجموعه داده Spiral، (الف) Complete Link، (ب) Average Link، (ج) Single Link و (د) K-Means، Number of iterations: ۹.



شکل ۲۵: نتایج اعمال الگوریتم پیشنهادی بر روی مجموعه داده GPS_Volunteer_۰۱.

در این مجموعه داده در قالب خوشه‌های متفاوت نشان داده شده‌اند.

تا اینجا دیدیم که الگوریتم پیشنهادی دارای دقت بالا و درصد خطای پایین بود. حال می‌خواهیم در مورد پارامترهای الگوریتم پیشنهادی بحث کنیم. ما با اضافه‌کردن دو پارامتر L و k توانستیم مقادیر مختلف پارامتر Eps را به طور خودکار تعیین کنیم و قبلاً گفتیم که پارامتر k باید حداقل به اندازه $Minpts$ باشد. اما در رابطه با پارامتر L باید بگوییم که تعیین یک مقدار مناسب برای این پارامتر چندان مشکل نیست. در شکل ۲۶ به ازای مقادیر مختلف پارامتر L خروجی الگوریتم پیشنهادی با توجه به معیار درصد خطای خوشه‌بندی بر روی سه مجموعه داده نشان داده شده است. در واقع در این شکل خروجی الگوریتم با توجه به تغییر مقدار پارامتر L آمده است در حالی که مقدار پارامترهای K و $Minpts$ مطابق با جدول ۷ ثابت نگه داشته شده است.

همان گونه که در شکل ۲۶ مشاهده می‌کنید نتایج نشان می‌دهد که به ازای هر مجموعه داده، پارامتر L به ازای مقادیر نزدیک به $Minpts$ بهترین نتایج را تولید می‌کند. از این رو تعیین مقدار دقیق پارامتر L چندان چالشی نیست و می‌توانیم این پارامتر را برابر با $Minpts$ تنظیم کنیم. در واقع با توجه به این که هر خوشه باید حداقل $Minpts$ عضو داشته باشد، از این رو جمع فاصله از $Minpts$ همسایه هر نقطه تقریب

جدول ۷: مقادیر پارامتر تنظیم‌شده برای مجموعه داده‌های مورد ارزیابی.

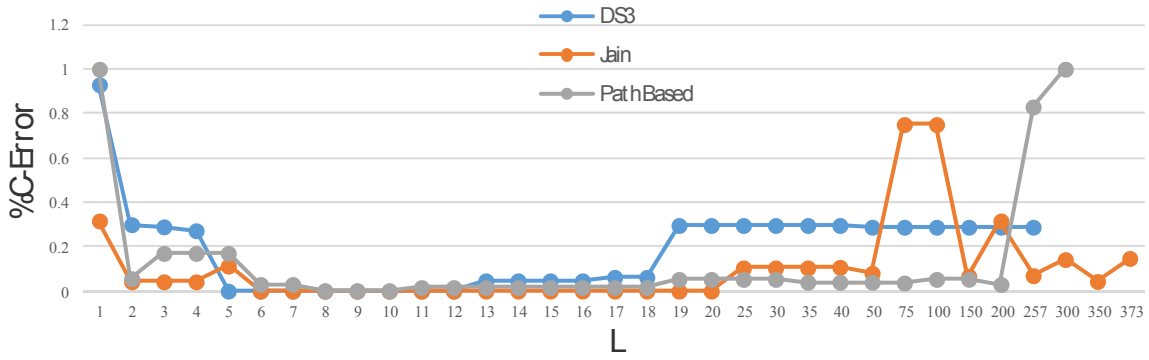
Parameters / Data sets	*MinPts*	*K*
DS۳	۶	۷
Jain	۱۴	۲۶
Path Based	۱۰	۲۴

خوبی برای چگالی محلی نقاط است. ناگفته نماند که در اعمال الگوریتم پیشنهادی بر روی بقیه مجموعه داده‌های ارائه‌شده در جدول ۱ نیز مقدار L برابر با $Minpts$ تنظیم شده و بنابراین می‌توان از پارامتر L به عنوان یک پارامتر ورودی صرف نظر کرد.

۶- نتیجه‌گیری و کارهای آتی

خوشه‌بندی یکی از تکنیک‌های مهم در داده‌کاوی محسوب می‌شود. الگوریتم‌های خوشه‌بندی مبتنی بر چگالی با توجه به این که فهمشان ساده است و قابلیت تشخیص نویز و اشکال اختیاری از خوشه‌ها را دارند، بسیار محبوب هستند. الگوریتم DBSCAN به عنوان الگوریتم پایه روش‌های خوشه‌بندی مبتنی بر چگالی، علی‌رغم مزایایی که دارد در تشخیص خوشه‌های با چگالی متفاوت ناتوان است. در این مقاله الگوریتمی با مرتبه زمانی $O(n \log n)$ برای بهبود الگوریتم DBSCAN در تشخیص خوشه‌های با چگالی متفاوت ارائه شد. الگوریتم پیشنهادی علاوه بر تشخیص خوشه‌های با چگالی متفاوت در تشخیص خوشه‌های تو در تو و چسبیده به هم با اندازه و اشکال اختیاری نیز به خوبی عمل می‌کند.

برای ارزیابی الگوریتم پیشنهادی از معیارهای استاندارد ارزیابی روش‌های خوشه‌بندی استفاده گردیده و الگوریتم پیشنهادی بر روی مجموعه داده‌های استاندارد و مصنوعی با سایر الگوریتم‌های برتر هم رده خود مقایسه شده است. به منظور بررسی صحت نتایج ارزیابی الگوریتم پیشنهادی و اثبات معنادار بودن آماری بهبود عملکرد به دست آمده از معیار بازه اطمینان (CI) استفاده کرده‌ایم. همچنین الگوریتم پیشنهادی با



شکل ۲۶: خروجی الگوریتم پیشنهادی بر روی سه مجموعه داده DS۳، Jain و PathBased با توجه به مقادیر مختلف L.

سایر الگوریتم‌های برتر موجود در دیگر دسته‌های خوشه‌بندی نیز مقایسه شده است. بر اساس نتایج به دست آمده از آزمایش‌ها، الگوریتم پیشنهادی از دقت بالایی برخوردار بوده و نتایج به مراتب بهتری نسبت به الگوریتم‌های دیگر دارد. الگوریتم پیشنهادی به خصوص بر روی مجموعه داده‌های بزرگ که درصد نویز پایینی داشته باشند کارایی بالایی دارد. به طور کلی، الگوریتم پیشنهادی نیاز به ۳ پارامتر ورودی دارد ولی همان طور که در قسمت ارزیابی نشان دادیم، پارامتر L را می‌توان با مقدار ثابت $Minpts$ تنظیم کرد. بنابراین برای این الگوریتم تنها تعیین مقدار دو پارامتر $Minpts$ و k کافی است. تعیین مقدار دیگر پارامترها نیز با توجه به کاهش حساسیت الگوریتم به پارامترهای ورودی، کار مشکلی نیست.

به طور کلی در پایگاه داده‌های بزرگ، لازم است که به منظور صرفه‌جویی در زمان و افزایش دقت، پارامترهای الگوریتم‌های خوشه‌بندی تا حد ممکن به صورت خودکار انتخاب شوند. از این رو به عنوان کارهای آتی می‌توان بر روی تعیین خودکار پارامترهای ورودی این الگوریتم کار کرد. همچنین بهبود پیچیدگی زمانی الگوریتم پیشنهادی می‌تواند در آینده مورد توجه قرار گیرد. این کار می‌تواند از طریق بهبود یا کاهش پرس و جوهای ناحیه‌ای انجام‌گرفته، صورت بگیرد. همچنین با توجه به این که پایگاه داده‌های بسیار بزرگ نیاز به قدرت محاسباتی بالایی دارند می‌توان به منظور بهبود کارایی الگوریتم پیشنهادی در این پایگاه داده‌ها، نسخه موازی این الگوریتم را ارائه کرد.

مراجع

[1] R. H. Guting, "An introduction to spatial database systems," *The International J. on Very Large Data Bases*, vol. 3, no. 4, pp. 357-399, Oct. 1994.

[2] J. Mennis and D. Guo, "Spatial data mining and geographic knowledge discovery-an introduction," *Computers, Environment and Urban Systems*, vol. 33, no. 6, pp. 403-408, Nov. 2009.

[3] C. J. Matheus, P. K. Chan, and G. Piatetsky-Shapiro, "Systems for knowledge discovery in databases," *IEEE Trans. on Knowledge and Data Engineering,* vol. 5, no. 6, pp. 903-913, Dec. 1993.

[4] N. Soni and A. Ganatra, "Comparative study of several clustering algorithms," *International J. of Advanced Computer Research*, vol. 2, no. 6, pp. 37-42, Dec. 2012.

[5] T. Liu, C. Rosenberg, and H. A. Rowley, "Clustering billions of images with large scale nearest neighbor search," in *Proc. IEEE Workshop on Applications of Computer Vision, WACV'07,* pp. 28-28, Austin, TX, USA, 21-22 Feb. 2007.

[6] M. E. Celebi, Y. A. Aslandogan, and P. R. Bergstresser, "Mining biomedical images with density-based clustering," in *Proc. Int. Conf. on Information Technology: Coding and Computing, ITCC'05*, pp. 163-168, Las Vegas, NV, USA, 4-6 Apr. 2005.

[7] M. Celik, F. Dadaser-Celik, and A. Dokuz, "Anomaly detection in temperature data using DBSCAN algorithm," in *Proc. Int. Symp. on Innovations in Intelligent Systems and Applications, INISTA'11,* pp. 91-95, Istanbul, Turkey, 15-18 Jun. 2011.

[8] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Computer Communications*, vol. 30, no. 14, pp. 2826-2841, Oct. 2007.

[9] N. Soni and A. Ganatra, "Categorization of several clustering algorithms from different perspective: a review," *International J. of Advanced Research in Computer Science and Software Engineering*, vol. 2, no. 8, pp. 63-68, Aug. 2012.

[10] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. of the 2nd Int. Conf. on Knowledge Discovery and Data Mining, KDD'06*, pp. 226-231, Portland, OR, USA, 2-4 Aug. 1996.

[11] M. Ankerst, M. M. Breunig, H. P. Kriegel, and J. Sander, "Optics: ordering points to identify the clustering structure," in *Proc. of the 1999 ACM Int. Conf. on Management of data*, *SIGMOD'99,* pp. 49-60, 31 May-3 Jun. 1999.

[12] P. Liu, D. Zhou, and N. Wu, "VDBSCAN: varied density based spatial clustering of applications with noise," in *Proc. Int. Conf. on Service Systems and Service Management,* 4 pp., Chengdu, China, 9-11 Jun. 2007.

[13] A. Ram, S. Jalal, A. S. Jalal, and M. Kumar, "A density based algorithm for discovering density varied clusters in large spatial databases," *Int. J. of Computer Applications*, vol. 3, Article 1, 4 pp., 2010.

[14] A. Fahim, A. Salem, F. Torkey, and M. Ramadan, "Density clustering based on radius of data (DCBRD)," *International Scholarly and Scientific Research & Innovation*, vol. 2, no. 10, pp. 3463-3469, 2008.

[15] L. Duan, L. Xu, F. Guo, J. Lee, and B. Yan, "A local-density based spatial clustering algorithm with noise," *Information Systems*, vol. 32, no. 7, pp. 978-986, Nov. 2007.

[16] M. M. Breunig, H. P. Kriegel, R. T. Ng, and J. Sander, "LOF: identifying density-based local outliers," in *Proc. of the 1999 ACM Int. Conf. on Management of data, SIGMOD'99*, pp. 93-104, 15-18 May 2000.

[17] M. N. Gaonkar and K. Sawant, "AutoEpsDBSCAN: DBSCAN with Eps automatic for large dataset," *International J. on Advanced Computer Theory and Engineering,* vol. 2, no. 2, pp. 2319-2526, Aug. 2013.

[18] B. Borah and D. K. Bhattacharyya, "DDSC: a density differentiated spatial clustering technique," *J. of Computers*, vol. 3, no. 2, pp. 72-79, Feb. 2008.

[19] W. Ashour and S. Sunoallah, "Multi density DBSCAN," in *Intelligent Data Engineering and Automated Learning-IDEAL*, Ed: Springer, pp. 446-453, 2011.

[20] M. T. Elbatta, R. M. Bolbol, and W. M. Ashour, "A vibration method for discovering density varied clusters," *International Scholarly Research Notices*, vol. 2012, Article ID 723516, 8 pp., 2011.

[21] S. Mitra and J. Nandy, "KDDClus: a simple method for multi-density clustering," in *Proc. of the Int. Workshop on Soft Computing Applications and Knowledge Discovery*, *SKAD'11,* pp. 72-76, Moscow, Russia, 25 Jun. 2011.

[22] R. K. Prasad and R. Sarmah, "Variable density spatial data clustering," in *Proc. 2nd Int. Conf. on Computer and Communication Technology, ICCCT'11*, vol. ???, pp. 53-58, 25 Jun. 2011.

[23] S. Vijayalakshmi and M. Punithavalli, "Improved varied density based spatial clustering algorithm with noise," in *Proc. IEEE Int. Conf. on Computational Intelligence and Computing Research, ICCIC'10,* 4 pp., Coimbatore, India, 28-29 Dec. 2010.

[24] S. Wang, Y. Liu, and B. Shen, "MDBSCAN: multi-level density based spatial clustering of applications with noise," in Proc. *of the the 11th Int. Knowledge Management in Organizations Conf. on the*

Changing Face of Knowledge Management Impacting Society, p. 24, Hagen, Germany, 25-28 Jul. 2016.
[25] W. Atwa and K. Li, "Constraint-based clustering algorithm for multi-density data and arbitrary shapes," in *Proc. Industrial Conf. on Data Mining*, *ICDM'17,* pp. 78-92, 2017.
[26] S. Louhichi, M. Gzara, and H. Ben-Abdallah, "Unsupervised varied density based clustering algorithm using spline," *Pattern Recognition Letters*, vol. 93, no. 8, pp. 48-57, Jul. 2017.
[27] A. Amini, H. Saboohi, T. Herawan, and T. Ying Wah, "MuDi-Stream: a multi density clustering algorithm for evolving data stream," *J. of Network and Computer Applications*, vol. 59, no. 3, pp. 370-385, Jan. 2016.
[28] C. Fahy, S. Yang, and M. Augusto Gongora, "Finding multi-density clusters in non-stationary data streams using an ant colony with adaptive parameters," in *Proc. IEEE Congress on Evolutionary Computation*, pp. 673-680, San Sebastian, Spain, 5-8 Jun. 2017.
[29] 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, no. 1, pp. 92-96, Jan. 2013.
[30] S. Zhou, A. Zhou, W. Jin, Y. Fan, and W. Qian, "FDBSCAN: a fast DBSCAN algorithm," *Journal of Software*, vol. 11, no. 6, pp. 735-744, Jun. 2000.
[31] J. H. Peter and A. Antonysamy, "An optimised density based clustering algorithm," *International J. of Computer Applications*, vol. 6, no. 9, pp. 20-25, Sept. 2010.
[32] C. F. Tsai, C. T. Wu, and S. Chen, "GF-DBSCAN; a new efficient and effective data clustering technique for large databases," in *Proc. WSEAS Int. Conf. Mathematics and Computers in Science and Engineering*, pp. 231-236, Hangzhou, China, 20-22 May 2009.
[33] X. Wang and H. J. Hamilton, "DBRS: a density-based spatial clustering method with random sampling," in *Proc. Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, *PAKDD'03*, pp. 563-575, 2003.
[34] J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "Density-based clustering in spatial databases: the algorithm gdbscan and its applications," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 169-194, Jun. 1998.
[35] D. Birant and A. Kut, "ST-DBSCAN: an algorithm for clustering spatial-temporal data," *Data & Knowledge Engineering*, vol. 60, no. 1, pp. 208-221, Jan. 2007.
[۳۶] ع. زاده ده بالایی، ع. ر. باقری و ح. افشار، "IMD-DBSCAN: یک الگوریتم خوشه‌بندی مبتنی بر چگالی افزایشی با قابلیت افزودن خوشه‌های با چگالی متفاوت در محیط‌های انباره داده،" *مجموعه مقالات بیستمین کنفرانس ملی سالانه انجمن کامپیوتر ایران*، صص. ۳۴-۲۸، مشهد، ۱۴-۱۲ اسفند ۱۳۹۳.
[37] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Trans. on Mathematical Software*, vol. 3, no. 3, pp. 209-226, Sept. 1977.
[38] —, *Clustering Datasets*, [Online]. http://cs.joensuu.fi/sipu/datasets/
[39] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *J. of the American Statistical Association*, vol. 66, no. 336, pp. 846-850, Dec. 1971.
[40] P. Jaccard, "Distribution of the alpine flora in the dranse's basin and some neighbouring regions," *Bulletin de la Societe Vaudoise des Sciences Naturelles*, vol. 37, no. 1, pp. 241-272, Jan. 1901.
[41] E. B. Fowlkes and C. L. Mallows, "A method for comparing two hierarchical clusterings," *J. of the American Statistical Association*, vol. 78, 383, pp. 553-569, Sept. 1983.
[42] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. of the Fifth Berkeley Symp. on Mathematical Statistics and Probability*, vol. 1, no. 14, pp. 281-297, Jun. 1967.
[43] P. Berkhin, "A survey of clustering data mining techniques," in Grouping Multidimensional Data, pp. 25-71, Springer Berlin Heidelberg, 2006.
[44] J. Krumm and A. J. Brush, *MSR GPS Privacy Dataset 2009*, http://research.microsoft.com /~jckrumm/GPSData2009.

علی زاده ده بالایی در سال ۱۳۹۰ مدرک کارشناسی مهندسی نرم‌افزار خود را از دانشگاه شهید باهنر کرمان و در سال ۱۳۹۳ مدرک کارشناسی ارشد مهندسی نرم‌افزار خود را از دانشگاه صنعتی امیرکبیردر تهران دریافت نمود. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: داده‌کاوی، الگوریتم‌های خوشه‌بندی، پایگاه‌داده مکانی و الگوریتم‌های موازی.

علیرضا باقری تحصیلات خود را در مقاطع کارشناسی و کارشناسی ارشد مهندسی نرم افزار به‌ترتیب در سال‌های ۱۳۷۵ و ۱۳۷۷ از دانشگاه صنعتی شریف و در مقطع دکتری علوم کامپیوتر در سال ۱۳۸۴ از دانشگاه صنعتی امیرکبیر به پایان رسانده است و هم‌اکنون استادیار دانشکده مهندسی کامپیوتر دانشگاه صنعتی امیرکبیر می‌باشد. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: هندسه محاسباتی، رسم گراف و الگوریتم‌های گراف، آنالیز داده‌های عظیم و آنالیز شبکه‌های اجتماعی.

حامد افشار در سال ۱۳۹۱ مدرک کارشناسی مهندسی فن‌آوری اطلاعات خود را از دانشگاه آزاد واحد جنوب تهران و در سال ۱۳۹۳ مدرک کارشناسی ارشد مهندسی نرم‌افزار خود را از دانشگاه صنعتی امیرکبیر در تهران دریافت نمود. نام‌برده از سال ۱۳۸۹ در شرکت دانش‌بنیان پروا سیستم (پارس پک) در زمینه رایانش ابری مشغول به فعالیت است. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: داده‌کاوی، رایانش ابری و مهندسی نرم‌افزار.