ارائه یک نسخه جدید از الگوریتم مورچگان باینری به منظور حل مسأله انتخاب ویژگی

شیما کاشف و حسین نظام‌آبادی‌پور

چکیده: استفاده از الگوریتم‌های ابتکاری یک انتخاب مناسب برای حل مسایل بهینه‌سازی است. در این مقاله نسخه بهبودیافته‌ای از الگوریتم بهینه‌ساز مورچگان باینری برای حل مسأله انتخاب ویژگی ارائه شده است. نسخه پیشنهادی خصوصیات الگوریتم جمعیت مورچه گسسته و الگوریتم مورچه باینری را به صورت توأمان در خود دارد. کارایی روش پیشنهادی روی ۱۲ پایگاه داده استاندارد در موضوع طبقه‌بندی بررسی و نتایج با چند الگوریتم مطرح در این زمینه شامل بهینه‌ساز جمعیت مورچگان گسسته و باینری مقایسه شده است. نتایج بیانگر کارایی مناسب الگوریتم پیشنهادی است.

کلید واژه: **انتخاب ویژگی، الگوریتم مورچگان باینری، طبقه‌بندی، کاهش بعد ویژگی.**

۱- مقدمه

در بازشناسی الگو، شناسایی ویژگی‌هایی که بیشترین قدرت تمایزدهی بین گروه‌ها را دارند، قدم مهمی است. به علت فراوانی نویز، اطلاعات جعلی و نیز وجود ویژگی‌های تکراری و نامرتبط در بین مجموعه ویژگی اولیه، مسأله انتخاب ویژگی ضروری به نظر می‌رسد. از طرف دیگر افزایش تعداد ویژگی‌ها نه تنها هزینه محاسباتی سیستم را افزایش می‌دهد، بلکه باعث کاهش دقت طبقه‌بندی نیز می‌شود [۱]. همه طبقه‌بندها در ابعاد پایین، نتایج قابل قبولی ارائه می‌کنند در حالی که در ابعاد بالا با مشکل شناخته‌شده‌ای به نام بلای بعد[1] روبه‌رو هستند. بنابراین انتخاب ویژگی می‌تواند تأثیر قابل توجهی بر نرخ بازشناسی درست الگوریتم طبقه‌بندی داشته باشد. این موضوع ما را به سمتی هدایت می‌کند که از روش‌های جستجو برای پیداکردن زیرمجموعه‌ای بهینه از ویژگی‌ها استفاده کنیم.

در حالت عمومی مشخص نیست که کدام زیرمجموعه از ویژگی‌ها بیشترین تمایز را برای تفکیک گروه‌های مختلف الگوهای مورد مطالعه ایجاد می‌کند و از طرف دیگر امکان بررسی تمام زیرمجموعه‌های موجود، از نظر صرف وقت، امکان‌پذیر نبوده و مقرون به صرفه نیست. هدف اصلی از انتخاب ویژگی، کاهش بعد بردار ویژگی در طبقه‌بندی الگو است به طوری که نرخ طبقه‌بندی قابل قبولی نیز حاصل شود. در این شرایط ویژگی‌هایی که قدرت تمایز کمتری دارند، حذف شده و مجموعه‌ای از ویژگی‌ها که شامل اطلاعات مناسبی برای تمایز دسته‌های الگو هستند، باقی می‌مانند [۲].

این مقاله در تاریخ ۳۱ مرداد ماه ۱۳۹۲ دریافت و در تاریخ ۲۸ اسفند ماه ۱۳۹۲ بازنگری شد.

شیما کاشف، بخش مهندسی برق، دانشگاه شهید باهنر کرمان، کرمان، (email: shkashef.1988@yahoo.com).

حسین نظام‌آبادی‌پور، بخش مهندسی برق، دانشگاه شهید باهنر کرمان، کرمان، (email: nezam@uk.ac.ir).

1. Curse of Dimensionality

تا کنون روش‌های متنوعی برای انتخاب ویژگی استفاده شده که بسیاری از آنها در [۳] تا [۵] مرور و بررسی شده‌اند. به طور کلی این روش‌ها را می‌توان در سه گروه اصلی دسته‌بندی کرد: روش‌های فیلتر، روش‌های پیچشی[2] و روش‌های ترکیبی [۵]. روش‌های فیلتر، مستقل از الگوریتم‌های یادگیری عمل می‌کنند و به عبارت دیگر، بازخوردی از الگوریتم یادگیری اعمال‌شده (طبقه‌بند) به سامانه انتخاب ویژگی اعمال نمی‌شود. در این روش‌ها با استفاده از معیارهایی خاص، برای هر ویژگی یک اعتبار محاسبه می‌شود، سپس ویژگی‌ها بر اساس رتبه‌هایشان مرتب شده و ویژگی‌هایی که رتبه آنها کمتر از یک مقدار آستانه از پیش تعیین شده است، حذف می‌گردند. این روش‌ها از سرعت بالایی برخوردارند و برای داده‌های با ابعاد زیاد مناسبند. تا کنون روش‌های فیلتر زیادی نظیر F-score [۶]، اطلاعات متقابل[3] [۷]، بهره اطلاعاتی[4] [۸]، همبستگی[5] [۹] و غیره در زمینه انتخاب ویژگی مطرح شده است.

در روش پیچشی [۹] و [۱۰]، معیار انتخاب ویژگی، کارایی الگوریتم طبقه‌بندی است و بنابراین دقت این روش بسیار بالا است اما از آنجا که برای بررسی میزان شایستگی هر زیرمجموعه، باید الگوریتم طبقه‌بند آموزش دیده و اجرا شود، سرعت کار این روش، پایین است و پیچیدگی محاسباتی بالایی دارد و این امر استفاده از این روش را در داده‌های با ابعاد بالا محدود می‌سازد. روش‌های فیلتر و پیچشی، دو روش مکمل هم هستند و بنابراین روش‌های ترکیبی با به کارگیری مزیت‌های هر دو روش، قادرند جواب‌های بهتری تولید کند. این ویژگی مهم، توجه بسیاری از محققان را در دهه گذشته به خود جلب کرده است [۱۱] تا [۱۳].

آزمایش‌ها نشان داده که در مجموعه‌های بزرگ، استفاده از روش‌های ابتکاری اثربخش‌تر از سایر روش‌ها است. الگوریتم‌های جستجوی ابتکاری الگوریتم‌هایی هستند که با الهام از فرایندهای فیزیکی و بیولوژیکی در طبیعت به دست آمده‌اند و غالب آنها به صورت جمعیتی عمل می‌کنند. این روش‌ها تنها از تابع شایستگی برای هدایت جستجو استفاده می‌کنند اما به علت دارابودن هوشمندی جمعی، قادر به کشف جواب هستند. از جمله این روش‌ها می‌توان به الگوریتم وراثتی [۱۴] و [۱۵]، الگوریتم جمعیت ذرات [۱۶] و [۱۷]، الگوریتم جستجوی گرانشی [۱۸] تا [۲۱]، روش‌های مبتنی بر فازی و غیره اشاره کرد.

بهینه‌ساز جمعیت مورچگان[6] (ACO) از دسته الگوریتم‌های جستجوی ابتکاری است که از رفتار مورچه‌های واقعی در طبیعت الهام گرفته است [۲۲] تا [۲۴]. این روش مانند اکثر روش‌های جستجوی تکاملی با یک جمعیت از جواب‌ها، جستجو را به شکل موازی شروع می‌کند، سپس

2. Wrapper
3. Mutual Information
4. Information Gain
5. Correlation
6. Ant Colony Optimization

شایستگی هر یک از افراد جمعیت را بر اساس یک تابع هزینه تعیین کرده و با استفاده از مقادیر شایستگی، اطلاعات جمعیت را به روز می‌کند. این الگوریتم توانایی زیادی در حل مسایلی دارد که به شکل گراف قابل عرضه و نمایش هستند. برای حل مسأله انتخاب ویژگی با استفاده از این الگوریتم، ویژگی‌ها در جایگاه گره‌های گراف قرار می‌گیرند. مورچه‌ها با استفاده از قوانین احتمالاتی حرکت، با عبور از یال‌ها ویژگی‌های خاصی را انتخاب می‌کنند [۲۵]. در به کارگیری این روش، تعداد ویژگی‌ها از قبل توسط کاربر تعیین می‌شود، بنابراین در این روش مورچه‌ها تنها می‌توانند تعداد گام‌های محدودی بردارند و به آنها اجازه داده نمی‌شود که از تمام گره‌های گراف عبور کنند، زیرا این به معنی انتخاب تمام ویژگی‌ها است. به عبارت دیگر هر گره‌ای که مورچه از آن بگذرد، ویژگی متناظر آن انتخاب و در حافظه مورچه ثبت می‌شود. بنابراین ایراد اصلی این روش، تعیین تعداد ویژگی‌ها پیش از اجرای الگوریتم است که ممکن است منجر به یک زیرمجموعه نزدیک بهینه نشود.

برای حل این مشکل از نسخه باینری این الگوریتم در حل مسأله انتخاب ویژگی استفاده شده است. نسخه اولیه الگوریتم باینری مورچه، اولین بار در سال ۲۰۰۰ در [۲۶] با نام TACO[1] ارائه شد. حل مسأله انتخاب ویژگی با استفاده از یک نسخه بهبودیافته از الگوریتم جمعیت مورچه باینری برای اولین بار توسط توحیدی و همکاران انجام شد [۲۷]. اگرچه این روش، مشکل مربوط به الگوریتم مورچه گسسته را در انتخاب ویژگی مرتفع کرد اما هنوز یک مشکل اساسی برای این الگوریتم وجود دارد و آن این که هر مورچه، ترتیب یکسانی از ویژگی‌ها را می‌بیند و در هر لحظه تنها می‌تواند بین انتخاب و عدم انتخاب ویژگی بعد از خودش (صرفاً یک گره خاص که متناظر با یک ویژگی خاص است) تصمیم بگیرد و قادر به دیدن سایر ویژگی‌ها (سایر گره‌های گراف) نیست. این محدودیت، توان کاوش الگوریتم را به شدت کاهش داده و منجر به تولید زیرمجموعه‌هایی غیر بهینه خواهد شد، اگرچه پاسخ‌ها به مراتب بهتر از روش الگوریتم مورچه گسسته هستند.

برای رفع مشکلات الگوریتم‌های مورچه گسسته و باینری در حل مسأله انتخاب ویژگی، نسخه جدیدی از این الگوریتم ارائه می‌شود که ترکیبی از این دو الگوریتم است [۲۸] یعنی مانند الگوریتم مورچه گسسته، در هر گره مورچه‌ها اجازه دیدن تمام ویژگی‌ها (گره‌های دیگر) را دارند و هم‌زمان اختیار انتخاب یا عدم انتخاب ویژگی‌ها به آنها داده می‌شود که این منجر به انعطاف‌پذیری در انتخاب تعداد ویژگی‌ها می‌شود.

ادامه این مقاله این گونه سازمان‌دهی شده که در بخش دوم الگوریتم جمعیت مورچگان گسسته و باینری معرفی خواهند شد. در بخش سوم نسخه پیشنهادی الگوریتم مورچه برای حل مسأله انتخاب ویژگی آمده است. در بخش چهارم، کارایی روش پیشنهادی روی چند مجموعه داده استاندارد بررسی شده و نتایج مقایسه با سایر روش‌ها گزارش می‌شود و در نهایت در بخش پنجم، مقاله جمع‌بندی می‌شود.

۲- الگوریتم بهینه‌ساز جمعیت مورچگان

الگوریتم ACO با الهام از رفتار مورچه‌ها در طبیعت ایجاد شده است. مورچه‌ها قادرند با وجود کور و کم هوش بودن، کوتاه‌ترین مسیر رفت و برگشت از خانه تا غذا را پیدا کنند. زیست‌شناسان دریافتند که این توانایی در نتیجه رد پای فرومونی است که مورچه‌ها از آن برای برقراری ارتباط با یکدیگر و انتقال اطلاعات مسیر استفاده می‌کنند. این ردپا با ایجاد یک مسیر پس‌خورد مثبت، مورچه‌ها را به سمت مسیرهای هرچه کوتاه‌تر هدایت می‌کند. از معایب بزرگ الگوریتم مورچه وابستگی شدید آن به پارامترهای الگوریتم است به گونه‌ای که اگر پارامترهای الگوریتم به خوبی انتخاب نشوند، احتمال رکود و همگرایی به بهینه‌های محلی در آن زیاد خواهد بود. در این بخش مروری بر الگوریتم مورچه گسسته و باینری خواهد شد.

۲-۱ الگوریتم جمعیت مورچه گسسته در انتخاب ویژگی

برای استفاده از الگوریتم بهینه‌سازی جمعیت مورچگان در حل مسایل مختلف، مسأله باید به صورت گراف قابل تعریف باشد. در مسأله انتخاب ویژگی، ویژگی‌ها به جای گره‌های گراف قرار می‌گیرند. ابتدا مورچه‌ها به صورت تصادفی روی گراف پخش می‌شوند و سپس هر مورچه برای انتخاب گره بعدی مسیر خود از (۱) استفاده می‌کنند که به قانون گذار معروف است

$$P_{ij}^{k}(t)=\begin{cases}\dfrac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{l}\tau_{il}^{\alpha}\eta_{il}^{\beta}} & \text{If } l \text{ and } j \text{ are admissible nodes}\\ 0 & \text{otherwise}\end{cases} \quad (1)$$

که در آن تابع P_{ij}^{k} احتمال بودن مورچه k در گره j در زمان $t+1$ می‌باشد، اگر در زمان t در گره i باشد. η_{ij} هزینه جابجایی از گره iام به گره jام، هیوریستیک (یا بینایی) مسئله و τ_{ij} بیانگر میزان رد پای فرومونی مورچه‌ها روی یالی است که گره iام را به گره jام وصل می‌کند. این پارامتر، باعث بینایی مورچه‌های مصنوعی می‌شود و α و β پارامترهای مسئله هستند که میزان اهمیت رد پا در مقابل بینایی را کنترل می‌کنند. به روز کردن ردپای شاخه‌ها شامل دو قسمت است: یکی تبخیر رد پاهای قبلی و دیگری اضافه‌کردن رد پا به شاخه‌هایی که مورچه‌ها اخیراً از آن گذشته‌اند (روابط (۲) و (۳)). این گزینه می‌تواند در زمان‌های مختلفی مثل بعد از هر سیکل (سفر) یا بعد از هر قدم صورت گیرد که در این مقاله بعد از این که همه مورچه‌ها یک سفر کامل را به اتمام رساندند، یعنی بعد از یک سیکل انجام می‌شود

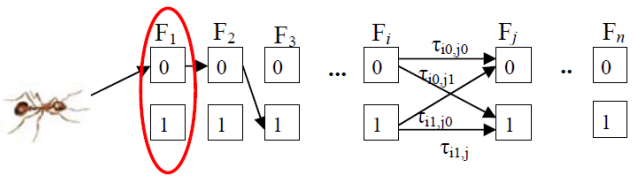
$$\tau_{ij}(new)=(1-\rho)\tau_{ij}+\Delta\tau_{ij} \quad , \quad \Delta\tau_{ij}=\sum_{k=1}^{m}\Delta\tau_{ij}^{k} \quad (2)$$

$$\Delta\tau_{ij}^{k}=\begin{cases}\dfrac{Q}{F^{k}} & \text{if the } k-\text{th ant traverse arc } (i,j) \text{ in } T_k\\ 0 & \text{otherwise}\end{cases} \quad (3)$$

که در آن $\Delta\tau_{ij}^{k}$ و $\Delta\tau_{ij}$ به ترتیب مقدار رد پایی است که مورچه kام و رد پایی که تمام مورچه‌ها بر یال (i,j) اضافه می‌کنند. F_k هزینه طی‌کردن مسیری است که مورچه kام از آن گذشته و شاخه (i,j) در آن مسیر قرار دارد. منظور از T_k، سفر مورچه kام یا مسیر حرکت این مورچه است. m تعداد مورچه‌ها و $\rho\in(0,1]$ نرخ تبخیر رد پا است که از انباشته‌شدن بیش از حد فرومون جلوگیری می‌کند و باعث می‌شود تا همواره امکان جستجوی مناسبی برای مورچه‌ها وجود داشته باشد. برای توقف تعداد سیکل‌های الگوریتم، ملاک‌های مختلفی در نظر گرفته می‌شود. از جمله این روش‌ها، توقف پس از همگراشدن مورچه‌ها به یک جواب یا توقف پس از تعداد سیکل‌های معین است که در این مقاله از مورد دوم استفاده شده است.

در حل مسأله انتخاب ویژگی توسط الگوریتم مورچه گسسته پس از آن که مورچه‌ها به طور اتفاقی روی گره‌های شبکه پخش شدند، هر مورچه

1. Touring Ant Colony Optimization



شکل ۱: الگوریتم TACO. رشته باینری به وسیله مسیری که مورچه از آن گذر کرده است، مشخص می‌شود.

به تعداد معینی که برابر با طول مجموعه ویژگی است و توسط کاربر تعیین می‌شود، گام برمی‌دارد به گونه‌ای که انتخاب هر گره به منزله انتخاب ویژگی متناظر با آن است. پس از برداشتن این تعداد گام (یعنی انتخاب تعداد از پیش تعیین شده ویژگی) یک سیکل به اتمام می‌رسد. در این حالت، حافظه هر مورچه بیانگر یک زیرمجموعه از ویژگی‌هاست که ارزش آن تعیین می‌شود. تعیین ارزش و کارایی یک زیرمجموعه از ویژگی‌ها در این مقاله با استفاده از روش‌های پیچشی تعیین می‌شود. یعنی این که دقت طبقه‌بند تعیین‌کننده میزان شایستگی (یا هزینه) یک سفر مورچه است. بعد از آن رد پاهای مورچه‌ها به روز رسانی شده و سیکل بعدی الگوریتم آغاز می‌شود.

۲-۲ الگوریتم جمعیت مورچه باینری در انتخاب ویژگی

همان گونه که ذکر شد، در سال ۲۰۰۰ در [۲۶] روشی برای حل مسایل فضای باینری با نام TACO ارائه و بعدها از آن برای حل مسایل مختلف بهینه‌سازی استفاده شد [۲۹] و [۳۰]. در این روش، دو رشته موازی از صفر و یک در نظر گرفته می‌شود که مورچه‌ها از بین آنها گذشته و یک مسیر از صفر و یک‌ها می‌سازند. شکل ۱ چگونگی روش مذکور را به تصویر کشیده و همان طور که مشهود است، برای هر ویژگی یک گره صفر در رشته بالایی و یک گره یک در رشته پایینی در نظر گرفته شده است. هر مورچه در هر حرکت دو انتخاب (صفر یا یک) دارد که با توجه به تابع احتمالی که برای حرکت تعریف شده است، یکی را به عنوان موقعیت بعدی خود انتخاب می‌کند.

هر مورچه مسیرهایی را که از آنها گذشته است به حافظه می‌سپرد. بعد از آن که سفر مورچه به پایان رسید، حافظه هر مورچه بیانگر مسیری است که مورچه از آن گذشته که جواب مسأله می‌باشد. هر مورچه بعد از اتمام سفرش، در مسیرهای عبوری خود مقداری رد پا به جا می‌گذارد. میزان رد پای فرومونی به جا مانده از هر مورچه با توجه به هزینه‌ای که مسیر طی شده به همراه داشته است، محاسبه می‌گردد. در قوانین حرکت نسخه باینری، مورچه‌ها فقط از اطلاعات فرومون مسیرها استفاده می‌کنند و وقتی یک مورچه مرحله تصمیم‌گیری را برای همه گره‌های رشته کامل کند به این معنی است که یک راه حل برای مسأله تولید شده است.

در روش TACO، تمام مورچه‌ها با توجه به شایستگیشان بر مسیرهایی که از آنها گذشته‌اند، رد پا می‌گذارند. با اقتباس از روش‌های موجود الگوریتم فضای گسسته ACO، سه روش به نام‌های ETACO[1]، RTACO[2] و BACO[3] برای بهبود TACO در [۲۷] پیشنهاد شد. این سه روش در دو مسأله طبقه‌بندی معنایی تصویر و بازشناسی ارقام دست‌نویس فارسی آزموده شدند. نتایج آزمایش‌ها نشان دادند روش BACO عملکرد بهتری نسبت به سایر روش‌های پیشنهادی و روش‌های مورد مقایسه دارد. در روش BACO پیشنهاد شده که مقدار کمینه و بیشینه رد پای مورچه‌ها بین دو مقدار τ_{min} و τ_{max} محدود شود و صرفاً مورچه‌ای که برازندگی بالاتری دارد، یعنی مسیر با کمترین هزینه را پشت سر گذاشته است، در مرحله رد پا گذاری شرکت کند.

در حل مسأله انتخاب ویژگی با استفاده از الگوریتم جمعیت مورچگان باینری به ازای هر ویژگی، یک گره (بیت) صفر و یک بیت یک با یک جایگاه مشخص و یکسان در رشته باینری در نظر گرفته می‌شود. از این رو طول رشته بیت‌های یک و صفر در رشته‌های شکل ۱ برای حل یک مسأله خاص با تعداد ویژگی‌های آن مسأله برابر است. در این مسأله چنانچه یک مورچه برای یک ویژگی خاص، بیت حاوی مقدار صفر را برگزیند به منزله عدم مشارکت ویژگی و چنانچه بیت یک را برگزیند به منزله انتخاب ویژگی و مشارکت آن تلقی می‌شود. در این روش، تمام مورچه‌ها در آغاز در ابتدای رشته بیت‌ها قرار گرفته و با شروع کار الگوریتم بر اساس تابع احتمال انتخاب (۴) گره بعدی خود را انتخاب می‌کنند

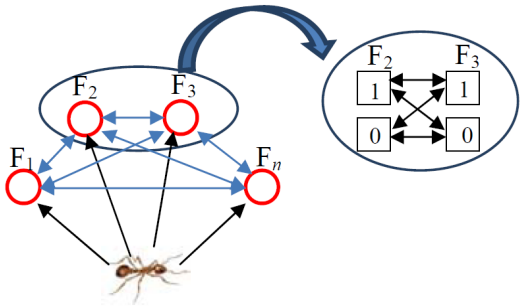
$$P^{k}_{ix,jy}(t)=\frac{\tau_{ix,jy}}{\tau_{ix,j.}+\tau_{ix,j\bar{1}}} \qquad (۴)$$

این رابطه، احتمال انتخاب بیت با مقدار $y\in\{0,1\}$ در گره بعدی (گره jام) برای مورچه kام که در زمان t در موقعیت $x\in\{0,1\}$ در گره iام قرار داشته را مشخص می‌کند. همچنین $\tau_{i.,j.}$، $\tau_{i\bar{1},j.}$، $\tau_{i.,j\bar{1}}$ و $\tau_{i\bar{1},j\bar{1}}$ به ترتیب بیانگر رد پای فرومونی بین مسیرهای متصل‌کننده گره‌های iام و jام به ترتیب روی یال‌های (۰ به ۰)، (۱ به ۰)، (۰ به ۱) و (۱ به ۱) است. لازم به یادآوری است که بنا به محدودیت‌های الگوریتم مورچه باینری همواره محدودیت $j=i+1$ برقرار است. یعنی این که هر مورچه در هر گره صرفاً می‌تواند به گره بعدی خود که از قبل مکان آن تعیین شده حرکت کرده و یکی از زیرگره‌های صفر و یک را برگزیند.

۲-۳ چالش‌ها

مشکل اصلی الگوریتم باینری مورچه در انتخاب ویژگی این است که در رشته باینری ترتیب خاصی از ویژگی‌ها قرار می‌گیرد و هر مورچه که در گره iام (متناظر با ویژگی iام) قرار گرفته فقط قادر به تصمیم‌گیری در خصوص انتخاب یا عدم انتخاب ویژگی بعدی است. حال چنانچه از این گره بگذرد و این ویژگی خاص را انتخاب نکند دیگر قادر نیست احتمال حضور این ویژگی را در گره‌های بعدی بررسی کند. این نوع خاص از بیان مسأله انتخاب ویژگی توسط الگوریتم باینری که به نحوی وابسته به ترتیب چیدن ویژگی‌ها است، می‌تواند اثرات منفی بر عملکرد نهایی الگوریتم داشته باشد. از سوی دیگر در این روش امکان ارائه یک راه حل جامع برای بینایی گره‌ها وجود ندارد. این سیاست‌ها باعث می‌شود که الگوریتم مورچه باینری کاوش کمی داشته باشد. در مقابل، الگوریتم مورچه گسسته همواره این امکان را تا پایان جستجو با خود دارد و امکان ارائه راه حل‌هایی برای تعریف بینایی در یال‌های متصل‌کننده ویژگی‌ها وجود دارد ولی ضعف آن در این است که در این الگوریتم، گذر هر مورچه از هر گره به منزله انتخاب ویژگی متناظر با آن است. بنابراین در این روش، کاربر می‌بایست قبل از اجرای الگوریتم تعداد ویژگی‌ها را تعیین نماید که این یک محدودیت جدی در انتخاب زیرمجموعه بهینه ویژگی‌ها محسوب می‌شود.

1. Elitism TACO
2. Ranked-Based TACO
3. Binary ACO



شکل ۲: الگوریتم ABACO. مسیری که مورچه از آن گذر کرده است، رشته باینری ایجادشده را مشخص می‌کند.

۳- الگوریتم پیشنهادی

در این مقاله برای رفع نقایص الگوریتم مورچگان باینری و الگوریتم مورچگان گسسته در مسأله انتخاب ویژگی، روشی ارائه شده که ترکیبی از دو الگوریتم فوق می‌باشد و تکامل‌یافته روشی است که توسط مؤلفان ارائه شده است [۲۸]. با ترکیب دو الگوریتم گسسته و باینری به گونه‌ای عمل می‌شود که ضعف‌های هر دو الگوریتم در الگوریتم جدید برطرف شود. به عبارتی از یک سو دیگر لازم نباشد که کاربر تعداد ویژگی‌ها را از پیش تعیین کند و این عمل در اختیار الگوریتم باشد، به طوری که امکان بررسی هر زیرمجموعه از ویژگی‌ها با هر اندازه دلخواهی را داشته باشد و از سوی دیگر هر مورچه در هر گره با تمام گره‌های دیگر گراف (مانند الگوریتم گسسته) در ارتباط باشد و بتواند با یک احتمال به یکی از گره‌هایی که در حافظه ندارد سفر کند. به عبارت دیگر در الگوریتم پیشنهادی، مورچه‌ها در طول کار الگوریتم می‌توانند تمام ویژگی‌ها را ببینند و بین انتخاب یا عدم انتخاب ویژگی‌ها با توجه به اطلاعات مسیرشان، تصمیم بگیرند. یعنی بر خلاف الگوریتم مورچگان باینری که مورچه‌ها یک ترتیب خاص از ویژگی‌ها را در تصمیم‌گیری دنبال کرده و در هر لحظه تنها می‌توانستند اطلاعات مسیرهای مربوط به ویژگی بعد از خود را ببینند، هر مورچه قادر است مسیرهای منتهی به تمام ویژگی‌ها را ببیند.

در روش پیشنهادی، مسأله انتخاب ویژگی مشابه حل آن توسط الگوریتم مورچه گسسته به صورت یک گراف کامل تعریف می‌شود و برای این منظور برای هر ویژگی یک گره در گراف در نظر گرفته می‌شود. از طرف دیگر برای این که مشکل الگوریتم گسسته حل شود برای هر گره دو زیر گره- مشابه با الگوریتم باینری- در نظر گرفته می‌شود که حاوی مقادیر صفر و یک هستند. هر مورچه در هر سفر خود در هر لحظه می‌تواند یکی از گره‌های دیگر را که متناظر با یکی از ویژگی‌ها است در صورتی که در حافظه مورچه نباشد بر مبنای یک تابع احتمال انتخاب نماید. شایان ذکر است که مورچه صرفاً به یکی از زیرگره‌های صفر یا یک سفر می‌کند که انتخاب زیرگره یک به منزله انتخاب آن ویژگی و انتخاب زیرگره صفر به منزله عدم انتخاب آن ویژگی است. در هر حال بعد از آن که سفر مورچه به گره مذکور انجام شد، آن گره در لیست حافظه مورچه قرار گرفته و مورچه بعد از آن دیگر به این گره سفر نخواهد کرد. بدین ترتیب ملاحظه می‌شود که با این تدبیر مسأله انتخاب ویژگی تقریباً شبیه به مسأله فروشنده دوره‌گرد مدل‌سازی می‌شود و مورچه سفر خود را از یک گره به صورت اتفاقی آغاز و بعد از گذر از تمام گره‌ها به پایان می‌رساند. نکته اصلی در این است که در روش پیشنهادی با لحاظ‌کردن دو زیرگره شامل مقادیر صفر و یک دیگر لزومی به تعیین تعداد ویژگی از قبل توسط کاربر نیست. از طرف دیگر از آنجا که هر گره شامل دو زیرگره است، بین هر دو گره به جای یک یال، چهار یال وجود دارد. شکل ۲ نحوه مدل‌سازی مسأله انتخاب ویژگی توسط الگوریتم پیشنهادی (ABACO)[1] را به تصویر کشیده است.

رابطه احتمال در روش پیشنهادی با (۵) توصیف می‌شود

$$P^{k}_{ix,jy}(t)=\begin{cases}\dfrac{\tau_{ix,jy}\eta_{ix,jy}}{\sum_{j}\tau_{ix,j}.\eta_{ix,jy}+\sum_{j}\tau_{ix,j\text{'}}\eta_{ix,j\text{'}}} & , \ j\in \text{admissible nodes} \\ \text{'}\cdot & , \ \text{otherwise}\end{cases} \quad (۵)$$

این رابطه، احتمال انتخاب بیت با مقدار $y\in\{\cdot,\text{'}\}$ در گره بعدی (گره j ام) برای مورچه k ام که در زمان t در موقعیت $x\in\{\cdot,\text{'}\}$ در گره i ام قرار داشته را مشخص می‌کند. همچنین $\tau_{i\cdot,j\cdot}$، $\tau_{i\text{'},j\cdot}$، $\tau_{i\cdot,j\text{'}}$ و $\tau_{i\text{'},j\text{'}}$ به ترتیب بیانگر رد پای فرومونی بین مسیرهای متصل‌کننده گره‌های i ام و j ام به ترتیب روی یال‌های (۰ به ۰)، (۱ به ۰)، (۰ به ۱) و (۱ به ۱) است. لازم به ذکر است که در این روش محدودیت روش باینری برطرف شده است. برای توضیح بیشتر در یک حالت خاص از (۵)، $\tau_{i_x,j\text{'}}$ بیانگر رد پای فرومونی موجود بر مسیری است که مورچه را از حالت عدم انتخاب ویژگی i ام به انتخاب ویژگی j ام هدایت می‌کند. ویژگی j ام باید متعلق به مجموعه ویژگی‌های مجاز باشد، یعنی ویژگی‌هایی که مورچه k ام هنوز از آنها بازدید نکرده است. $x\in\{\cdot,\text{'}\}$ نشان‌دهنده وجود دو جایگاه ۰ و ۱ برای ویژگی j ام است. لازم به ذکر است که در روش پیشنهادی از آنجا که مورچه‌ها قادر به دیدن و بررسی همه ویژگی‌ها هستند و مجاز به انتخاب یا عدم انتخاب ویژگی‌ها می‌باشند، احتمال پیداکردن زیرمجموعه بهینه بالاتر می‌رود.

۴- آزمایش‌ها و نتایج

برای حل مسأله انتخاب ویژگی با استفاده از الگوریتم مورچگان، هر جواب ممکن از مسأله به صورت رشته‌ای از صفر و یک‌ها تعریف می‌شود که طول آن برابر تعداد ویژگی‌ها می‌باشد. رشته به دست آمده توسط هر مورچه این گونه رمزگشایی می‌شود که وجود یک، بیانگر در نظر گرفتن ویژگی متناظر با آن و وجود صفر، بیانگر حذف ویژگی است. برای تعیین میزان شایستگی هر جواب، ویژگی‌های داده‌شده توسط هر جواب به یک طبقه‌بند داده می‌شود. زیرمجوعه ویژگی که نرخ طبقه‌بندی صحیح بالاتری را به دست بدهد، امتیاز بیشتری را کسب می‌کند. نرخ طبقه‌بندی صحیح به صورت درصد نمونه‌هایی که به درستی طبقه‌بندی شده‌اند به تعداد کل نمونه‌ها، تعریف می‌شود.

۴-۱ معیارهای ارزیابی

برای بررسی کیفیت نتایج طبقه‌بندی از دو معیار معروف دقت[2] و یادآوری[3] و نیز پارامتری به نام F_r[4] استفاده کردیم. دقت به صورت تعداد نمونه‌هایی که به درستی به دسته i نسبت داده شده‌اند به تعداد کل نمونه‌هایی که به دسته i نسبت داده شده‌اند، تعریف می‌شود (رابطه (۶)). یادآوری به صورت تعداد نمونه‌هایی که به درستی به دسته i نسبت داده شده‌اند به تعداد کل نمونه‌هایی که متعلق به دسته i هستند، تعریف

1. Advanced Binary ACO
2. Precision
3. Recall
4. Feature Reduction

جدول ۱: مشخصه مجموعه ویژگی‌های مختلف استفاده‌شده.

مجموعه داده	تعداد کلاس‌ها	تعداد ویژگی‌ها	تعداد نمونه‌ها
Abalone	۱۱	۸	۳۸۴۲
Glass	۶	۹	۲۱۴
Iris	۳	۴	۱۵۰
Letter	۲۶	۱۶	۲۰۰۰
Shuttle	۷	۹	۵۸۰۰۰
Spambase	۲	۵۷	۴۶۰۱
Tae	۳	۵	۱۵۱
Vehicle	۴	۱۸	۸۴۶
Waveform	۳	۲۱	۵۰۰۰
Wine	۳	۱۳	۱۷۸
Wisconsin	۲	۹	۶۸۳
Yeast	۹	۸	۱۴۸۴

جدول ۲: نحوه تنظیم پارامتر Q با استفاده از روش سعی و خطا به ازای مقدار ثابت پارامتر ρ . اعداد جدول حاصل میانگین‌گیری از نرخ طبقه‌بندی صحیح در ۵ اجرای مستقل از الگوریتم پیشنهادی می‌باشد.

	$\rho=۰/۲۵$		
	$Q=۰/۱$	$Q=۰/۳$	$Q=۰/۵$
Iris	۰/۹۶۶۷	۰/۹۶۳۳	۰/۹۶۶۷
Glass	۰/۷۲۷۵	۰/۷۷۸۹	۰/۷۲۷۸
Vehicle	۰/۷۴۳۰	۰/۷۵۱۳	۰/۷۵۶۷
Wine	۰/۹۶۵۴	۰/۹۷۰۳	۰/۹۷۵۹
Abalone	۰/۲۴۵۱	۰/۲۴۵۴	۰/۲۳۸۶
Letter	۰/۸۶۱۵	۰/۸۵۰۳	۰/۸۴۹۰
Shuttle	۰/۹۹۷۷	۰/۹۹۸۶	۰/۹۹۷۹
Spambase	۰/۹۲۹۱	۰/۹۲۵۵	۰/۹۲۴۶
Tae	۰/۵۸۳۳	۰/۵۴	۰/۵۷۳۳
Waveform	۰/۷۹۵۸	۰/۷۹۵۰	۰/۷۹۵۱
Wisconsin	۰/۹۶۸۵	۰/۹۷۵۰	۰/۹۷۵۱
Yeast	۰/۵۱۹۴	۰/۵۱۹۱	۰/۵۲۵۰
میانگین	۰/۷۷۵۳	۰/۷۷۶۰	۰/۷۷۵۵

می‌شود (رابطه (۷)). فرض کنید TP_i، FP_i، TN_i و FN_i به صورت زیر تعریف شوند [۳۱]

TP_i: تعداد نمونه‌های آزمونی که به درستی در iامین دسته طبقه‌بندی شده‌اند.

FP_i: تعداد نمونه‌های آزمونی که به اشتباه به iامین دسته طبقه‌بندی شده‌اند.

TN_i: تعداد نمونه‌های آزمونی که به درستی در دسته‌های دیگر طبقه‌بندی شده‌اند.

FN_i: تعداد نمونه‌های آزمونی که به اشتباه به دسته‌های دیگر طبقه‌بندی شده‌اند

$$precision_i = \frac{TP_i}{TP_i + FP_i} \quad (۶)$$

$$recall_i = \frac{TP_i}{TP_i + FN_i} \quad (۷)$$

برای بررسی تعداد ویژگی‌ها بعد از انتخاب ویژگی، نسبت به تعداد ویژگی‌های اصلی، پارامتری به نام F_r تعریف می‌شود که با رابطه زیر به دست می‌آید [۱۹]

$$F_r = \frac{p-q}{p} \quad (۸)$$

که p تعداد ویژگی‌های اصلی و q تعداد ویژگی‌ها بعد از انتخاب ویژگی است. طبق این رابطه هرچه مقدار F_r به ۱ نزدیک‌تر باشد، کاهش تعداد ویژگی‌ها بیشتر بوده و مطلوب‌تر است.

۴-۲ پایگاه داده

جهت انجام آزمایش‌ها، ۱۲ مجموعه داده شامل Abalone، Glass، Iris، Letter، Shuttle، Spambase، Tae، Vehicle، Waveform، Wine، Wisconsin و Yeast انتخاب شده‌اند. این مجموعه داده‌ها در پایگاه داده UCI [۳۲] موجود هستند و اکثر آنها به طور مکرر در بسیاری از مطالعات یادگیری ماشینی و شبکه عصبی استفاده شده‌اند [۳۳] و [۳۴]. جدول ۱ مشخصات عمومی این مجموعه داده‌ها را به صورت خلاصه نشان می‌دهد که بیانگر وجود تنوع در تعداد نمونه‌ها، ویژگی‌ها و دسته‌ها است. طبق این جدول تعداد دسته‌ها از ۲ تا ۲۶، تعداد ویژگی‌ها از ۴ تا ۵۷ و تعداد نمونه‌ها از ۱۵۰ تا ۵۸۰۰۰ متغیر است. اطلاعات دقیق‌تر این مجموعه داده‌ها در وب سایت UCI موجود است.

۴-۳ الگوریتم‌های مقایسه

برای ارزیابی کارایی الگوریتم پیشنهادی، الگوریتم گسسته مورچگان، ACO، الگوریتم مورچگان باینری BACO، الگوریتم وراثتی GA و الگوریتم جمعیت ذرات باینری BPSO، در حل مسأله انتخاب ویژگی پیاده‌سازی شده‌اند.

در تمام الگوریتم‌های مورد مقایسه، اندازه جمعیت و تعداد تکرارها برابر ۵۰ در نظر گرفته شده‌اند. در پیاده‌سازی اقسام مختلف الگوریتم مورچه، $\tau_0 = ۰/۱$ (رد پای اولیه)، $\tau_{\min} = ۰/۱$ (مینیمم رد پا)، $\tau_{\max} = ۶$ (ماکسیمم رد پا)، هزینه سفر (F_k) برابر عکس نرخ بازشناسی درست و پارامترهای کنترلی α و β به ترتیب برابر ۱ و ۰ در نظر گرفته شده‌اند. از آنجایی که مقدار دو پارامتر ρ (ضریب تبخیر) و Q در عملکرد الگوریتم تأثیر مستقیم دارد، تنظیم آنها اهمیت بالایی دارد. در اینجا از روش سعی و خطا برای تنظیم این دو پارامتر استفاده گردیده به این نحو که ابتدا مقدار ρ ثابت فرض شده و میانگین نرخ طبقه‌بندی صحیح برای الگوریتم پیشنهادی به ازای مقادیر مختلف Q به دست می‌آید (جدول ۲). سپس پارامتر Q به ازای بهترین مقدار ρ با همین روش به دست می‌آید (جدول ۳). بر اساس نتایج به دست آمده از این آزمایش‌ها مقادیر ρ و Q به ترتیب برابر با ۰/۰۵ و ۰/۳ در نظر گرفته شدند.

در الگوریتم وراثتی، همبری[1] از نوع تک‌نقطه‌ای با احتمال ۰/۹ و جهش[2] از نوع دودویی با احتمال ۰/۰۱ لحاظ شده‌اند. در الگوریتم جمعیت ذرات، وزن اینرسی $w = ۱$ و ضرایب $c_1 = c_2 = ۱$ در نظر گرفته شده‌اند.

در همه الگوریتم‌های مورد مقایسه، تعداد ویژگی‌ها در حین کار الگوریتم‌ها تعیین می‌شود. تنها در الگوریتم مورچگان گسسته (ACO) این تعداد باید توسط کاربر مشخص شود. بنابراین برای ایجاد شرایط یکسان در مقایسه کارکرد الگوریتم‌ها، ابتدا الگوریتم پیشنهادی برای ۲۰ مرتبه به

1. Crossover
2. Mutation

جدول ۳: نحوه تنظیم پارامتر ρ با استفاده از روش سعی و خطا به ازای مقدار ثابت پارامتر Q. اعداد جدول حاصل میانگین‌گیری از نرخ طبقه‌بندی صحیح در ۵ اجرای مستقل از الگوریتم پیشنهادی می‌باشد.

	$Q=۰/۳$		
	$\rho=۰/۲۵$	$\rho=۰/۱۵$	$\rho=۰/۰۵$
Iris	۰/۹۶۳۳	۰/۹۸	۰/۹۷
Glass	۰/۷۷۸۹	۰/۷۶۰۳	۰/۷۶۹۷
Vehicle	۰/۷۵۱۳	۰/۷۴۴۲	۰/۷۵۳۸
Wine	۰/۹۷۰۳	۰/۹۶۰۳	۰/۹۶۷۷
Abalone	۰/۲۴۵۴	۰/۲۴۰۱	۰/۲۴۲۶
Letter	۰/۸۵۰۳	۰/۸۵۷۴	۰/۸۶۰۱
Shuttle	۰/۹۹۸۶	۰/۹۹۸۱	۰/۹۹۸۳
Spambase	۰/۹۲۵۵	۰/۹۲۵۵	۰/۹۲۱۵
Tae	۰/۵۴	۰/۵۸	۰/۶۰۳۳
Waveform	۰/۷۹۵۰	۰/۷۹۸۵	۰/۷۹۱۲
Wisconsin	۰/۹۷۵۰	۰/۹۷۲۸	۰/۹۷۵۰
Yeast	۰/۵۱۹۱	۰/۵۲۲۴	۰/۵۲۲۵
میانگین	۰/۷۷۶۰	۰/۷۷۸۳	۰/۷۸۱۳

صورت مستقل اجرا شده، سپس میانگین تعداد ویژگی‌هایی را که این الگوریتم برای هر مجموعه داده انتخاب کرده بود به عنوان ورودی به ACO اعمال می‌شود.

۴-۴ نتایج

برای طبقه‌بندی از طبقه‌بند k همسایه نزدیک‌تر استفاده شده $(k=۱)$ که در این طبقه‌بند، ابتدا فاصله داده ورودی با تمام داده‌های آموزش محاسبه می‌شود که در اینجا از فاصله اقلیدسی استفاده شده است. سپس تعداد k داده‌ای که به داده ورودی نزدیک‌تر هستند، انتخاب می‌شوند. داده ورودی به گروهی تعلق می‌یابد که در آن تعداد بیشتری از k داده (با فاصله کمتر) وجود داشته باشد. ۶۰ درصد داده‌ها به طور تصادفی برای آموزش و ۴۰ درصد باقیمانده برای آزمون انتخاب شدند.

جدول ۴ میانگین نرخ طبقه‌بندی صحیح و همچنین مقدار پارامتر F_r را برای تمام الگوریتم‌ها روی هر مجموعه داده در ۲۰ اجرای مستقل نشان می‌دهد. در این جدول عدد داخل پرانتز نشان‌دهنده رتبه آن الگوریتم است. با تأمل در این داده‌ها مشاهده می‌شود که ABACO توانسته است در اکثر مجموعه داده‌ها به دقت طبقه‌بندی بالاتری نسبت به سایر الگوریتم‌ها دست یابد، ضمن این که پارامتر F_r را در محدوده این الگوریتم‌ها نگه داشته است. برای اطمینان از اهمیت آماری نتایج به دست آمده، از آزمون فردمن[1] استفاده می‌شود. با دست‌یابی به $p-value$ برابر با ۳×۱۰^{-۵} $(p-value<۰/۰۵)$ می‌توان از صحت مقایسه الگوریتم پیشنهادی با سایر الگوریتم‌ها اطمینان حاصل کرد. همچنین نتایج مقایسه پارامترهای دقت و یادآوری الگوریتم‌ها در جدول ۵ آورده شده است. طبق (۶) و (۷) هرچه مقدار این پارامترها به ۱ نزدیک‌تر باشد، الگوریتم مورد نظر عملکرد بهتری دارد. در جدول ۶ مقایسه دیگری بر مبنای مجموع رتبه‌های موجود در جدول ۳ برای هر الگوریتم انجام شده است. مقدار کمتر مجموع رتبه‌ها برای یک الگوریتم، نشان‌دهنده نتایج میانگین بهتر در میان سایر الگوریتم‌ها می‌باشد. اگرچه در بعضی موارد این کمیت دقت پایینی دارد اما استفاده از آن در آمار غیر پارامتری مرسوم است. با توجه به این جدول، ABACO رتبه اول و BACO رتبه دوم را کسب کرده‌اند. الگوریتم‌های GA، BPSO و ACO به ترتیب حائز رتبه‌های سوم تا پنجم شده‌اند. نتایج به دست آمده از الگوریتم پیشنهادی برتری این الگوریتم را در انتخاب ویژگی‌های مهم نسبت به سایر الگوریتم‌هایی که قبلاً توانایی خود را به اثبات رسانده بودند، نشان می‌دهد.

۵-۴ تحلیل نتایج و بحث

در این بخش به طور مختصر دلایل موفقیت ABACO را نسبت به سایر الگوریتم‌ها شرح می‌دهیم. در زیر، ویژگی‌های شاخص الگوریتم پیشنهادی ذکر شده است:

الگوریتم ABACO به مورچه‌ها اجازه می‌دهد تا بین تمام ویژگی‌های موجود کاوش کنند، در حالی که در اکثر روش‌های انتخاب ویژگی مبتنی بر الگوریتم مورچه، کاوش میان ویژگی‌ها تا زمانی ادامه پیدا می‌کند که شرایط توقف محقق شود. در نتیجه مورچه‌ها قادر به دیدن همه ویژگی‌ها نیستند و نمی‌توانند زیرمجموعه بهینه از ویژگی‌ها را انتخاب کنند.

مشخصه دوم این الگوریتم، روش جستجوی جدید آن است. بر خلاف روش‌های دیگر انتخاب ویژگی مبتنی بر مورچه که مورچه‌ها مجبور بودند هر ویژگی را که مشاهده می‌کنند انتخاب کنند، در اینجا مورچه‌ها قدرت انتخاب یا عدم انتخاب ویژگی‌هایی را که مشاهده می‌کنند، دارند. این روش جستجو بین ABACO و BACO مشترک است.

در نهایت، مشخصه‌ای که سبب برتری الگوریتم پیشنهادی نسبت به BACO می‌شود، دید جامعی است که مورچه‌ها به ویژگی‌ها دارند. همان طور که قبلاً گفته شد، الگوریتم BACO ویژگی‌ها را به صورت یک رشته باینری شامل دو مقدار صفر و یک پشت سر هم می‌چینند. بنابراین هر مورچه در هر لحظه تنها می‌تواند نسبت به انتخاب یا عدم انتخاب ویژگی بعد از خود تصمیم بگیرد و قادر به دیدن همه ویژگی‌ها نیست. این امر کیفیت جواب‌های به دست آمده را به ترتیب چیدمان ویژگی‌ها وابسته می‌کند. در نتیجه از آنجایی که در الگوریتم ABACO مورچه‌ها قادر به دیدن همه ویژگی‌هایی هستند که قبلاً از آنها بازدید نکرده‌اند، می‌توانند در مورد انتخاب ویژگی بعد بهتر تصمیم بگیرند. در واقع دیگر مجبور نیستند بین انتخاب یا عدم انتخاب یک ویژگی از پیش تعیین شده قضاوت کنند.

۵- جمع‌بندی

انتخاب ویژگی امر مهمی است که به طور مؤثر روی دقت طبقه‌بندی و بازشناسی اثر دارد. در این مقاله یک روش جدید انتخاب ویژگی مبتنی بر الگوریتم مورچگان باینری ارائه شد. الگوریتم پیشنهادی که ABACO نام دارد، دارای توانایی جستجوی بالایی در فضای مسأله است. این الگوریتم با الگوریتم مورچگان گسسته و باینری و همچنین دو الگوریتم ژنتیک و جمعیت ذرات باینری در حل مسأله انتخاب ویژگی مقایسه شده و توانست به نتایج بهتری دست یابد.

برای بررسی کارایی الگوریتم‌های مورد مقایسه از ۱۲ مجموعه داده از پایگاه داده UCI استفاده شد که تعداد ویژگی‌های کم، متوسط و زیاد را پوشش می‌دهند. نتایج به دست آمده، برتری الگوریتم پیشنهادی را در مقایسه با سایر الگوریتم‌ها در حل مسأله انتخاب ویژگی به اثبات می‌رساند.

1. Friedman

جدول ۴: نتایج پنج الگوریتم ABACO، BACO، ACO، GA و BPSO. مقادیر میانگین پارامترهای دقت طبقه‌بندی (C_r) و میزان کاهش ویژگی‌ها (F_r) در ۲۰ اجرای مستقل گزارش شده است. عدد داخل پرانتز، رتبه هر الگوریتم را نشان می‌دهد.

BPSO		GA		ACO		BACO		ABACO		
F_r	C_r	F_r	C_r	F_r	C_r	F_r	C_r	F_r	C_r	
۰٫۴۳۸	(۲) ۰٫۲۴۱	۰٫۳۷۳	(۲) ۰٫۲۴۱	۰٫۳۸۸	(۳) ۰٫۲۳۸	۰٫۳۸۷	(۱) ۰٫۲۴۳	۰٫۳۸۶	(۲) ۰٫۲۴۱	Abalone
۰٫۳۶۷	(۳) ۰٫۷۳۴	۰٫۳۳۹	(۴) ۰٫۷۳۳	۰٫۳۷۲	(۵) ۰٫۷۱۳	۰٫۳۴۷	(۲) ۰٫۷۴۳	۰٫۳۶۱	(۱) ۰٫۷۵۸	Glass
۰٫۴۵	(۳) ۰٫۹۶۵	۰٫۳۸۸	(۲) ۰٫۹۶۷	۰٫۳۱۳	(۴) ۰٫۹۶۳	۰٫۴۵۹	(۲) ۰٫۹۶۷	۰٫۳۷۵	(۱) ۰٫۹۷۴	Iris
۰٫۳۴۸	(۴) ۰٫۸۲۵	۰٫۳۴۱	(۳) ۰٫۸۳۷	۰٫۳۱۳	(۵) ۰٫۸۰۶	۰٫۳۳۲	(۱) ۰٫۸۵۹	۰٫۳۳۵	(۲) ۰٫۸۵۶	Letter
۰٫۵۳۱	(۱) ۰٫۹۹۸	۰٫۵۶۱	(۲) ۰٫۹۹۷	۰٫۴۷۲	(۱) ۰٫۹۹۸	۰٫۵۶۵	(۱) ۰٫۹۹۸	۰٫۴۵۳	(۱) ۰٫۹۹۸	Shuttle
۰٫۴۷۷	(۵) ۰٫۹	۰٫۴۵۴	(۳) ۰٫۹۰۶	۰٫۴۶۱	(۴) ۰٫۹۰۱	۰٫۴۶۵	(۲) ۰٫۹۱۹	۰٫۴۳۹	(۱) ۰٫۹۲۱	Spambase
۰٫۳۸۰	(۳) ۰٫۵۶۷	۰٫۴۴۱	(۵) ۰٫۵۵۶	۰٫۴۱	(۱) ۰٫۵۷۸	۰٫۳۶۳	(۴) ۰٫۵۵۸	۰٫۳۸۹	(۲) ۰٫۵۷۳	Tae
۰٫۴۷۸	(۴) ۰٫۷۲۲	۰٫۴۵۴	(۳) ۰٫۷۳۷	۰٫۴۳۱	(۵) ۰٫۷۱۸	۰٫۴۳۹	(۲) ۰٫۷۴۹	۰٫۴۲۸	(۱) ۰٫۷۵۳	Vehicle
۰٫۲۸۳	(۳) ۰٫۷۹۲	۰٫۳۲۴	(۴) ۰٫۷۷۶	۰٫۳۷۱	(۵) ۰٫۷۶۸	۰٫۲۶۱	(۲) ۰٫۷۹۳	۰٫۲۷۲	(۱) ۰٫۷۹۵	Waveform
۰٫۵۳۱	(۴) ۰٫۹۴۱	۰٫۴۹۰	(۳) ۰٫۹۵۷	۰٫۵۰۴	(۵) ۰٫۹۳۸	۰٫۴۹۴	(۲) ۰٫۹۶۴	۰٫۵۴۹	(۱) ۰٫۹۶۹	Wine
۰٫۴۰۶	(۳) ۰٫۹۶۸	۰٫۳۶۸	(۲) ۰٫۹۷۵	۰٫۳۸۹	(۳) ۰٫۹۶۸	۰٫۳۸۷	(۲) ۰٫۹۷۵	۰٫۳۴۸	(۱) ۰٫۹۷۶	Wisconsin
۰٫۲۰۶	(۵) ۰٫۵۰۷	۰٫۱۴۱	(۳) ۰٫۵۱۳	۰٫۱۵۶	(۴) ۰٫۵۰۹	۰٫۰۸۹	(۲) ۰٫۵۱۵	۰٫۰۷۶	(۱) ۰٫۵۲۹	Yeast

جدول ۵: پارامترهای دقت و یادآوری برای سنجش کارایی الگوریتم‌های مورد مقایسه.

BPSO		GA		ACO		BACO		ABACO		
Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	
۰٫۲۱۷	۰٫۲۲۱	۰٫۲۲۲	۰٫۲۲۵	۰٫۲۱۹	۰٫۲۲۲	۰٫۲۲۲	۰٫۲۲۳	۰٫۲۱۹	۰٫۲۲۱	Abalone
۰٫۶۷۳	۰٫۷۰۸	۰٫۶۸۲	۰٫۷۱۹	۰٫۶۵۴	۰٫۶۴۸	۰٫۶۹۵	۰٫۷۳	۰٫۶۷۶	۰٫۶۹۳	Glass
۰٫۹۶۵	۰٫۹۶۵	۰٫۹۶۸	۰٫۹۶۹	۰٫۹۶۲	۰٫۹۶۳	۰٫۹۶۷	۰٫۹۶۷	۰٫۹۷۴	۰٫۹۷۴	Iris
۰٫۸۵۸	۰٫۸۶۳	۰٫۸۳۹	۰٫۸۴۳	۰٫۸۰۸	۰٫۸۱۲	۰٫۸۶۳	۰٫۸۶۵	۰٫۸۶۱	۰٫۸۶۳	Letter
۰٫۹۱	۰٫۹۷۰	۰٫۸۹۴	۰٫۹۲۵	۰٫۹۱۴	۰٫۹۳۱	۰٫۹۳۰	۰٫۹۴۴	۰٫۹۰۷	۰٫۹۱۲	Shuttle
۰٫۸۹۵	۰٫۸۹۶	۰٫۹۰۱	۰٫۹۰۳	۸۹۶	۰٫۸۹۸	۰٫۹۱۷	۰٫۹۱۷	۰٫۹۱۶	۰٫۹۱۹	Spambase
۰٫۵۷۱	۰٫۵۷۸	۰٫۵۵۶	۰٫۵۶۷	۰٫۵۸۰	۰٫۵۸۹	۰٫۵۶۲	۰٫۵۶۸	۰٫۵۷۶	۰٫۵۸۵	Tae
۰٫۷۲۴۷	۰٫۷۱۸۴	۰٫۷۳۹۵	۰٫۷۳۴۳	۰٫۷۲۱۱	۰٫۷۱۶	۰٫۷۵۴	۰٫۷۴۹	۰٫۷۶۲	۰٫۷۵۵	Vehicle
۰٫۷۹۵	۰٫۷۹۵	۰٫۷۷۷	۰٫۷۷۷	۰٫۷۶۸	۰٫۷۶۸	۰٫۷۹۴	۰٫۷۹۴	۰٫۷۹۷	۰٫۷۹۷	Waveform
۰٫۹۴۶	۰٫۹۴۴	۰٫۹۶۴	۰٫۹۵۸	۰٫۹۴۳	۰٫۹۴۱	۰٫۹۷۲	۰٫۹۶۵	۰٫۹۷۳	۰٫۹۷۰	Wine
۰٫۹۶۶	۰٫۹۶۳	۰٫۹۷۵	۰٫۹۷۱	۰٫۹۶۶	۰٫۹۶۴	۰٫۹۷۴	۰٫۹۷۳	۰٫۹۷۶	۰٫۹۷۳	Wisconsin
۰٫۴۶۴	۰٫۴۶۶	۰٫۴۷۶	۰٫۴۸۰	۰٫۴۷۲	۰٫۴۶۴	۰٫۵۰۷	۰٫۵۰۹	۰٫۵۱۰	۰٫۵۰۴	Yeast

جدول ۶: مجموع رتبه‌های به دست آمده روی ۱۲ مجموعه داده برای هر الگوریتم.

BPSO	GA	ACO	BACO	ABACO
۴۰ (۴)	۳۶ (۳)	۴۵ (۵)	۲۳ (۲)	۱۵ (۱)

مراجع

[1] M. Kabir, Md. Shahjahan, and K. Murase, "A new local search standard, based hybrid genetic algorithm for feature selection," *Neurocomputing*, vol. 74, no. 17, pp. 2914-2928, Oct. 2011.

[2] S. M. Vieira, J. M. C. Sousa, and T. A. Runkler, "Two cooperative ant colonies for feature selection using fuzzy models," *Expert Systems with Applications*, vol. 37, no. 4, pp. 2714-2723, Apr. 2010.

[3] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. of Machine Learning Research*, vol. 3, pp. 1157-1182, Mar. 2003.

[4] M. Dash and H. Liu, "Feature selection for classification," *Intelligent Data Analysis*, vol. 1, no. 1-4, pp. 131-156, 1997.

[5] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Trans. on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 491-502, Apr. 2005.

[6] S. Ding, "Feature selection based F- score and ACO algorithm in support vector machine," in *Proc. 2nd Int. Symp. on Knowledge Acquisition and Modeling*, vol. 1, pp. 10-23, Nov. 2009.

[7] L. T. Vinh, S. Lee, Y. T. Park, and B. J. d'Auriol, "A novel feature selection method based on normalized mutual information," *Appl Intell*, vol. 37, no. 1, pp. 100-120, Jul. 2010.

[8] M. H. Aghdam, N. Ghasem-Aghaee, and M. E. Basiri, "Text feature selection using ant colony optimization," *Expert Systems with Applications*, vol. 36, no. 3, pp. 6843-6853, Apr. 2009.

[9] M. Kabir, M. Islam, and K. Murase, "A new wrapper feature selection approach using neural network," *Neurocomputing*, vol. 73, no. 16-18, pp. 3273-3283, Oct. 2011.

[10] R. Kohavi and G. John, "Wrappers for feature selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273-324, Dec. 1997.

[11] P. Bermejo, J. A. Gamez, and J. M. Puerta, "A GRASP algorithm for fast hybrid (filter - wrapper) feature subset selection in high - dimensional datasets," *Pattern Recognition Letters*, vol. 32, no. 5, pp. 701-711, Apr. 2011.

[12] J. Huang, Y. Cai, and X. Xu, "A hybrid genetic algorithm for feature selection wrapper based on mutual information," *Pattern Recognition Letters*, vol. 28, no. 13, pp. 1825-1844, Oct. 2007.

[13] R. K. Sivagaminathan and S. Ramakrishnan, "A hybrid approach for feature subset selection using neural networks and ant colony optimization," *Expert Systems with Applications*, vol. 33, no. 1, pp. 49-60, Jul. 2007.

[14] I. Oh, J. S. Lee, and B. R. Moon, "Hybrid genetic algorithm for feature selection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1424-1437, Nov. 2004.

[۱۵] س. م. رضوی، ه. صدوقی یزدی و ا. کبیر، "انتخاب ویژگی برای بازشناسی ارقام دست‌نویس فارسی به کمک الگوریتم وراثتی،" *هفتمین کنفرانس سالانه انجمن کامپیوتر ایران*، صص. ۲۸۵-۲۹۲، ۱۳۸۰.

[۱۶] م. رستمی شهر بابکی و ح. نظام آبادی پور، "انتخاب ویژگی در طبقه‌بندی معنایی تصاویر با استفاده از الگوریتم PSO،" *پانزدهمین کنفرانس سالانه مهندسی کامپیوتر ایران*، صص. ۲۶۹-۲۷۵، ۱۳۸۴.

[17] L. Y. Chuang, S. W. Tsai, and C. H. Yang, "Improved binary particle swarm optimization using catfish effect for feature selection," *Expert Systems with Applications*, vol. 38, no. 10, pp. 12699-12707, Sep. 2011.

[18] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "A simultaneous feature adaptation and feature selection method for content - based image retrieval systems," *Knowledge Based System*, vol. 39, pp. 85-94, Feb. 2013.

[19] E. Rashedi and H. Nezamabadi-pour, "Feature subset selection using improved binary gravitational search algorithm," *J. of Intelligent and Fuzzy Systems*, vol. 26, no. 3, pp. 1211-1221, May 2013.

[۲۰] ع. راشدی، ح. نظام آبادی پور و ح. توحیدی، "انتخاب ویژگی با استفاده از الگوریتم جستجوی گرانشی"، *سومین کنفرانس بین‌المللی فناوری اطلاعات و دانش، مشهد*، آذر ۱۳۸۶.

[21] S. Sarafrazi and H. Nezamabadi-pour, "Facing the classification of binary problems with a GSA-SVM hybrid system," *Mathematical and Computer Modelling*, vol. 57, no. 1-2, pp. 270-278, Jan. 2013.

[22] M. Dorigo and L. M. Gambardella, "Ant colonies for the traveling salesman problem," *BioSystems*, vol. 43, no. 2, pp. 73-81, Jul. 1997.

[23] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Trans. on Evolutionary Computation*, vol. 1, no. 1, pp. 53-66, Apr. 1997.

[24] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: optimization by a colony of cooperative agents," *IEEE Trans. on System, Man, and Cybernetics*, vol. 26, no. 1, pp. 1-13, Feb. 1996.

[25] A. Al-Ani, "Feature subset selection using ant colony optimization," *Int. J. of Computational Intelligence*, vol. 2, no. 1, pp. 53-58, 2005.

[26] T. Hiroyasu, M. Miki, Y. One, and Y. Minami, *Ant Colony for Continuous Functions*, the Science and Engineering, Doshisha University, 2000.

[۲۷] ح. توحیدی، ح. نظام آبادی پور و س. سریزدی، "انتخاب ویژگی با استفاده از الگوریتم جمعیت مورچگان باینری،" *اولین کنگره فازی و سیستم‌های هوشمند، دانشگاه فردوسی مشهد، ایران*، صص. ۲۶۹-۲۷۵، ۹-۷ شهریور ۱۳۸۶.

[28] S. Kashef and H. Nezamabadi-pour, "A new feature selection algorithm based on binary ant colony optimization," in *5th Conf. on Information and Knowledge Technology, IKT2013,* pp. 50-54, *Shiraz, Iran*, May 2013.

[29] N. Karaboga, A. Kalinli, and D. Karaboga, "Designing digital IIR filters using ant colony optimization algorithm," *Engineering Applications of Artificial Intelligence*, vol. 17, no. 3, pp. 301-309, Apr. 2004.

[30] L. Ozbakir, A. Baykasoglu, S. Kulluk, and H. Yapici, "TACO - miner: an ant colony based algorithm for rule extraction from trained neural networks," *Expert Systems with Applications*, vol. 36, no. 10, pp. 12295-12305, Dec. 2009.

[31] M. Janaki Meena, K. R. Chandran, A. Karthik, and A. Vijay Samuel, "An enhanced ACO algorithm to select features for text categorization and ites parallelization," *Expert Systems with Applications*, vol. 39, no. 5, pp. 5861-5871, Apr. 2012.

[32] *UCI Machine Learning Repository*, Center for Machine Learning and Intelligent Systems, http://archieve.ics.uci.edu/ml/datasets.html.

[33] L. Y. Chuang, C. H. Yang, and J. C. Li, "Chaotic maps based on binary particle swarm optimization for feature selection," *Applied Soft Computing*, vol. 11, no. 1, pp. 239-248, Jan. 2011.

[34] C. T. Su and H. C. Lin, "Applying electromagnetism-like mechanism for feature selection," *Information Science*, vol. 181, no. 5, pp. 972-986, Mar. 2011.

شیما کاشف مدرک کارشناسی و کارشناسی ارشد خود را در رشته‌ی مهندسی برق - الکترونیک به ترتیب در سال‌های ۱۳۹۰ و ۱۳۹۲ از دانشگاه شهید باهنر کرمان دریافت کرد. رساله ایشان در زمینه انتخاب ویژگی با استفاده از الگوریتم جمعیت مورچگان است. وی در سال ۱۳۹۳ در همان دانشگاه وارد مقطع دکتری برق -الکترونیک شد. زمینه‌های تحقیقاتی مورد علاقه‌ی او الگوریتم‌های تکاملی و رایانش نرم است.

حسین نظام‌آبادی‌پور دوره کارشناسی خود را در مهندسی برق الکترونیک در دانشگاه شهید باهنر کرمان در سال ۱۳۷۷ به پایان رساند. پس از آن، مدارک کارشناسی ارشد و دکتری خود را نیز در مهندسی برق الکترونیک از دانشگاه تربیت مدرس به‌ترتیب در سال‌های ۱۳۷۹ و ۱۳۸۳ دریافت کرد. وی هم‌اکنون استاد بخش مهندسی برق دانشگاه شهید باهنر کرمان است. زمینه‌های پژوهشی مورد علاقه‌ی او پردازش تصویر، بازشناسی الگو، کاربرد رایانش نرم در پردازش تصویر و روش‌های بهینه‌سازی ابتکاری است.