
بسم الله الرحمن الرحیم

دانشکده مهندسی برق و رباتیک

پایان‌نامه دوره کارشناسی ارشد مهندسی برق- الکترونیک

کاهش ویژگی با استفاده از الگوریتم بهینه سازی توده ذرات وکلاسه بندی با ماشین بردار پشتیبان

نگارش:

محسن صدیقی ناو

استاد راهنما:

دکتر سید علی سلیمانی ایوری

دکتر حسین خسروی

استاد مشاور:

دکتر علیرضا احمدی‌فرد

تابستان ۱۳۹۱

تقدیم به

پدر و مادرم و همسر عزیزم

تشکر و قدردانی

سپاس فراوان ایزد یکتا را که مهربانیش، یادش و همراهی پیوسته اش، همواره انگیزه من برای حرکت به سمت هدف بوده است.

ارج می‌نهم زحمات بی دریغ استاد راهنماهای بزرگوارم، دکتر علی سلیمانی و دکترحسین خسروی، که بدون راهنمایی های دلسوزانه و پیشنهادهای موثری که در طول تحقیق به بنده ارائه دادند این اثر به انجام نمی‌رسید.

همچنین از استاد مشاور گرانقدر خود دکتر علیرضا احمدی فرد نیز کمال تشکر و قدر دانی را دارم.

از تمامی اساتید گروه الکترونیک دانشگاه صنعتی شاهرود خصوصا دکتر هادی گرایلو و دکترامیدرضا معروضی سپاسگزاری و تشکر نموده و برای همه این بزرگواران آرزوی موفقیت و سلامت دارم.

در پایان جا دارد از زحمات و حمایتها و همراهی های دلسوزانه همسرم مهندس سمیرا محمد زاده که سختی ها و کاستی های مرا در طول مدت انجام پروژه و تحصیل تحمل نموده، تشکر و قدردانی نمایم.

چکیده

انتخاب ویژگی و کاهش ابعاد داده امری مهم و قابل توجه در بازشناسی الگو می‌باشد که در سالهای اخیر توجه زیادی بر آن بوده است. این امر باعث افزایش سرعت پردازش در سیستم های بلادرنگ و کاهش حافظه برای ذخیره سازی اطلاعات می‌شود. در این راستا نقش الگوریتم‌های بهینه سازی خصوصاً الگوریتم بهینه سازی توده ذرات قابل توجه بوده است. به طوریکه با استفاده از الگوریتم بهینه سازی توده ذرات و انتخاب روش مناسب جهت محاسبه مقدار برازندگی در PSO می‌توان تا مقدار قابل توجه‌ای ابعاد بردارهای ویژگی را کاهش داد تا به نتیجه قابل قبول و بهتر دست‌یابیم.

در این پایان‌نامه یک سیستم بازشناسی الگو برای بازشناسی ارقام دستنویس ارائه شده است. در این سیستم پس از استخراج ویژگی با استفاده از ترکیب دو روش هیستوگرام گرادیان و مکان مشخصه توسعه یافته بر روی تصاویر ارقام دستنویس مرحله انتخاب ویژگی انجام گرفته است. در بلوک انتخاب ویژگی از الگوریتم بهینه سازی توده ذرات باینری BPSO و طبقه‌بند ماشین بردار پشتیبان SVM جهت محاسبه مقدار برازندگی استفاده شده است. همچنین از داده‌های چهره پایگاه ORL، اثر انگشت FVC2004 و داده های UCI برای تایید کارایی سیستم استفاده شده است. طبق نتایج بدست آمده برای بازشناسی ارقام دستنویس فارسی بدون کاهش ویژگی به نرخ بازشناسی ۹۹,۴۰٪ و با استفاده از BPSO در انتخاب ویژگی تا ۱۹۳ ویژگی از ۴۰۰ ویژگی بردار اصلی به نرخ بازشناسی ۹۹,۱۱٪ دست‌یافتیم. این نتایج نشان از عملکرد خوب و قابل قبول سیستم پیشنهادی دارد.

کلیدواژه: BPSO، ماشین بردار پشتیبان، بازشناسی ارقام دستنویس فارسی، هیستوگرام گرادیان، مکان مشخصه توسعه یافته

فهرست مطالب

عنوان **صفحه**

فهرست اصطلاحات اختصاری

عنوان	علامت اختصاری
Support Vector Machine	SVM
Feature Selection	FS
Genetic Algorithm	GA
Ant Colony	ACO
Tabu Search	TS
Particle Swarm Optimization	PSO
Binary Particle Swarm optimization	BPSO
Fast Fourier Transform	FFT
Discrete Cosine Transform	DCT
Discrete Wavelet Transform	DWT
Discrete Fourier Transform	DFT
K-nearest neighbor	K-NN
Discrete Multi wavelet Transform	DMWT
Multi-Layer Perceptron	MLP
Optical character recognition	OCR
Principal Component Analysis	PCA
Factor Analysis	FA
Independent Component Analysis	ICA
Random Projection	RP
Projection Pursuit	PP
Karhunen Loeve Transform	KLT
Singular Value Decomposition	SVD
Artificial Neural Network	ANN
Deoxyribonucleic Acid	DNA
Fitness Function	FF
Swarm Intelligence	SI
Travelling Salesman Problem	TSP
Quadratic assignment problem	QAP
Tabu List	TL
Local Search	LS
Neural Network	NN
Hilbert Transform	HT
Sobel-Roberts Features	SRF
Face Recognition	FR
Equal Error Rate	EER
False Match Rate	FMR
False NonMatch Rate	FNMR

فهرست جدول‌ها

عنوان **صفحه**

فهرست شکل‌ها

عنوان **صفحه**

فصل ۱

مقدمه و مروری بر مطالعات و تحقیقات گذشته

۱-۱- پیشگفتار

مسأله انتخاب ویژگی[۱]، یکی از مسایلی است که در مبحث یادگیری ماشین[۲] و در بررسی مسایل شناخت الگو[۳] از دیدگاه آمار مطرح است. هدف ما از انتخاب و کاهش ویژگی[۴] افزایش سرعت پردازش[۵] و حذف ویژگی‌های کم اهمیت است. انتخاب ویژگی امکان دارد علاوه بر این که در دستیابی به پاسخ بهتر ما را یاری نکند ما را از رسیدن به آن نیز منحرف نماید. یک روش انتخاب ویژگی خوب می‌تواند تعداد ویژگی‌های موجود را کاهش وکارایی و دقت طبقه‌بندی[۶] را افزایش دهد. این مسأله در بسیاری از کاربردها مانند طبقه‌بندی اهمیت به سزایی دارد، زیرا در این گونه موارد تعداد زیادی ویژگی وجود دارد، که بسیاری از آنها یا بلا استفاده هستند و یا اینکه بار اطلاعاتی چندانی ندارند. حذف کردن این ویژگی‌ها مشکلی از لحاظ اطلاعاتی ایجاد نمی‌کند، ولی حذف ویژگی‌های نامرتبط و اضافی بار محاسباتی را کاهش می‌دهد. علاوه بر این ویژگی‌های نامرتبط باعث می‌شوند که اطلاعات غیر مفید زیادی را به همراه داده-های مفید ذخیره کنیم. در این راستا کارهای فراوانی در دهه های اخیر صورت گرفته است از جمله می-توان به استفاده از الگوریتم‌های بهینه‌سازی در مسأله انتخاب ویژگی نام برد. روشهای مختلفی برای این منظور استفاده شده است از روشهای مبتنی بر استخراج ویژگی که از تبدیلات استفاده می‌کنند تا روشهای مبتنی بر انتخاب ویژگی که ویژگی‌ها را در فضای تحلیل خود مورد بررسی قرار می‌دهند از جمله این روشها می‌توان به الگوریتم ژنتیک[۷]، الگوریتم تقسیم ومحدود کردن، الگوریتم جستجوی ترتیبی، اطلاعات دو طرفه، جستجوی ممنوعه[۸]، کلونی مورچگان[۹] و از همه مهمتر در سالهای اخیر به الگوریتم بهینه‌سازی توده ذرات[۱۰] اشاره نمود.

---

[1] Feature selection
[2] Machine learning
[3] Pattern recognition
[4] Feature Reduction
[5] Processing
[6] Classification
[7] Genetic Algorithm
[8] Tabu Search
[9] Ant Colony
[10] Particle Swarm Optimization

۱-۲- انگیزه

الگوریتم‌های زیادی جهت استخراج ویژگی[۱] مورد بررسی و معرفی قرار گرفته و استفاده می‌شوند و حتی هم اکنون نیز در راه بهبود این الگوریتم ها تحقیقات ومقالات زیادی در دست اجرا است و هر یک به نوبه خود در جایی که مورد استفاده قرار می‌گیرند مزیتها و محدودیتهایی را شامل هستند. این الگوریتم‌ها داده‌ها را در ابعاد مختلف تولید می‌نمایند. همچنین پردازش سریع اطلاعات به عنوان یک چالش برای محققان محسوب می‌شود و برای پردازش سریع و بلادرنگ[۲] نیاز به داده‌ها با ابعاد ویژگی کم است. همان گونه که گفته شد ویژگی‌هایی نیز در بین این داده‌های استخراجی وجود دارد که اثر سوء بر نرخ بازشناسی[۳] دارند در همین راستا بهتر است علاوه بر تحقیق و استفاده از انواع روش‌های استخراج ویژگی، روی مسأله انتخاب ویژگی نیز تمرکز کرده و با کاهش ابعاد داده سرعت و دقت شناسایی رابالا نگه داریم.

برای این منظور الگوریتم‌های بهینه‌سازی بر پایه جستجوی تصادفی[۴] کارایی بهتری نسبت به الگوریتم‌هایی که جستجو را به صورت کامل[۵] یا مکاشفه‌ای[۶] انجام می‌دهند داشته و در این میان الگوریتم بهینه‌سازی توده ذرات (PSO) بر اساس نتایج حاصل شده از مقالات و کتابهای مختلف نسبت به سایر الگوریتم‌های ذکر شده در رسیدن به پاسخ بهینه در عمل انتخاب زیر مجموعه ویژگی مناسب موفق تر عمل کرده و نتایج بهتر و قابل قبول‌تری را بدست آورده است. از طرفی برای یافتن بهترین زیر مجموعه ویژگی از میان سایر زیرمجموعه‌ها از تابع شایستگی استفاده می‌شود که کار بر روی این قسمت از الگوریتم‌های بهینه سازی به خلق روشهای بهینه در انتخاب ویژگی منجر می‌شود. در این پایان‌نامه از طبقه بند ماشین بردار پشتیبان[۷](SVM) به عنوان تابع برازندگی الگوریتم بهینه‌سازی توده ذرات استفاده نموده ایم. ماشین بردار پشتیبان روش مناسب و مورد اعتماد که کارایی فوق العاده بالایی در طبقه بندی

[1] Feature Extraction
[2] Real-time
[3] Recognition Rate
[4] Random Search
[5] Completa Search
[6] Heuristic Search
[7] Support Vector Machines

الگو دارد و در سالهای اخیر بطور وسیعی در مسایل طبقه بندی و رگرسیون[1] خطی و غیر خطی نیز مورد استفاده قرار گرفته است و نتایج بسیار بهتری نسبت به سایر روش‌های مشابه ارائه می‌دهد.

۱-۳-اهداف پایان نامه

هدف اصلی که ما در این پایان نامه دنبال می‌نماییم ارائه الگوریتم جهت استخراج ویژگی بر تصاویر ارقام دستنویس و روشی نو برای بهبود الگوریتم انتخاب ویژگی و افزایش سرعت پردازش در سیستم های بلادرنگ و صرفه جویی در زمان و حجم ذخیره سازی[2] اطلاعات است. در این پایان نامه از الگوریتم بهینه سازی توده ذرات باینری[3] (BPSO) برای انتخاب زیر مجموعه ویژگی‌های بهینه استفاده شده و از طبقه بند ماشین بردار پشتیبان برای محاسبه تابع برازندگی[4]، استفاده شده است. برای قسمت بررسی صحت و کارایی الگوریتم پیشنهادی از داده‌های ارقام دستنویس[5] فارسی هدی به عنوان داده های اصلی و داده-های پایگاه داده UCI، تصاویر چهره پایگاه داده ORL وداده های اثر انگشت FVC2004 استفاده شده است. همچنین برای مقایسه الگوریتم پیشنهادی از الگوریتم ژنتیک بر روی داده های چهره استفاده نموده ایم.

برای پیاده سازی الگوریتم PSO و ژنتیک از محیط نرم افزاری MATLAB استفاده نموده ایم. داده ها در دو فاز آموزش[6] و تست[7] مورد ارزیابی قرار گرفته اند. در فاز آموزش پس از استخراج ویژگی با الگوریتم پیشنهادی در این بخش (ترکیب روشهای هیستوگرام گرادیان و مکان مشخصه توسعه یافته بر ارقام دستنویس فارسی و الگوریتم سوبل رابرتز بر چهره و اثر انگشت) با استفاده از الگوریتم باینری PSO یکی از بهترین الگوهای انتخاب ویژگی موجود بدست آمده و بر اساس همین الگو[8] و پارامتر های ذخیره شده

[1] Regression
[2] Storage
[3] Binary Particle Swarm optimization
[4] Fitness function
[5] Handwriting digits
[6] Training Phase
[7] Testing
[8] Pattern

طبقه‌بند ماشین بردار پشتیبان در فاز آموزش، در فاز تست ویژگی های استخراج شده از داده های تست را انتخاب کرده و عمل طبقه بندی داده های تست صورت می‌گیرد.

۱-۴- مروری بر مطالعات و تحقیقات گذشته

در زمینه انتخاب ویژگی با استفاده از الگوریتم‌های بهینه سازی خصوصاً PSO کارهای فراوانی روی داده های مختلف صورت گرفته است. همچنین در سالهای اخیر از الگوریتم BPSO با تابع برازندگی‌های مختلف برای انتخاب ویژگی بر روی داده های فارسی و عربی نیز استفاده شده است اما کار بیشتری مورد نیاز است. در این بخش به بررسی تعدادی از تحقیقاتی که در سالهای اخیر در زمینه انتخاب ویژگی بر روی داده های مختلف استاندارد که برای کاربردهای مختلف صورت گرفته می‌پردازیم.

۱. در مقاله ای آقای عبدالقادر و دوستانش از PSO و GA برای انتخاب زیرمجموعه ویژگی مناسب بر روی ویژگی‌های استخراج شده با روش‌های تبدیل موجک گسسته[1] (DWT) و تبدیل کسینوسی گسسته[2] (DCT) از دیتا بیس استاندارد چهره ORL استفاده نموده‌اند و در بازشناسی نهایی از ۱۰ نفر موجود از ۴ نفر جهت آموزش و ۶ نفر جهت تست استفاده نموده‌اند که در بهترین حالت بازشناسی به نرخ ۹۶٫۸٪ دست یافته اند [۱].

۲. در مقاله ای آقای چنگ و دوستانش از طبقه بند ماشین بردار پشتیبان روش یکی در برابر دیگری و تعداد کلاسهایی که توسط آن درست تشخیص داده شده به عنوان تابع برازندگی در PSO استفاده کرده اند. دراین مقاله از داده های دستنوشته استفاده شده است و الگوریتم پیشنهادی نسبت به سایر الگوریتم ها نتایج بهتری را حاصل نموده است [۲].

۳. در مقاله ای آقای لی یه چوانگ و دوستانش از فاصله نزدیکترین همسایگی[3] (K-NN) ویژگی‌ها نسبت به هم و تعداد ویژگی‌هایی که درست در کلاس خود قرار گرفته‌اند برای محاسبه مقدار

[1] Discrete Wavelet Transform
[2] Discrete Fourier Transform
[3] K-nearest neighbor

برازندگی استفاده شده است و نتایج نشان از توانایی بالای این الگوریتم نسبت به سایر الگوریتم ها است [۳].

۴. در مقاله ای آقای عادل عبدل وهاب و دوستانش از PSO برای انتخاب زیرمجموعه ویژگی مناسب بر روی ویژگی‌های استخراج شده با روش تبدیل موجک چند مرحله‌ای[1] (DMWT) از دیتا بیس استاندارد چهره ORL استفاده نموده‌اند و در بازشناسی نهایی از ۱۰ نفر موجود از ۴ نفر جهت آموزش و ۶ نفر جهت تست استفاده نموده‌اند که در بهترین حالت بازشناسی به نرخ ۹۶٫۸٪ دست یافته اند [۴].

۵. در مقاله ای آقایان علائی و دوستانش یک روش قوی در استخراج ویژگی برای بالا بردن نرخ بازشناسی بر اساس اصلاح ویژگی کانتور ارائه دادند. در این مقاله از ماشین بردار پشتیبان (SVM) برای طبقه‌بندی استفاده شده است. همچنین از داده ارقام دستنویس فارسی هدی در این مقاله استفاده شده است. درنتیجه برای تعداد ۶۰۰۰۰ نمونه آموزش و۲۰۰۰۰ نمونه تست به نرخ بازشناسی ۹۸٫۷۱٪ و علاوه بر آن با معرفی و ارائه تکنیک (Five-Fold Cross Validation) جهت اصلاح ویژگی‌های کانتور برای ۸۰۰۰۰ نمونه فوق در بازشناسی به نرخ ۹۹٫۳۷٪ دست یافته اند [۵].

۶. در مقاله ای آقایان علائی و دوستانش با ترکیب ویژگی های استخراج شده با ۲ روش Chain-Code و ویژگی های اصلاح شده انتقال و با استفاده از طبقه بند SVM بر روی داده های دیتا بیس هدی در بازشناسی به نرخ ۹۹٫۰۲٪ دست یافته اند [۶].

۷. در مقاله ای آقایان پروین و دوستانش از الگوریتم ژنتیک (GA) جهت ساخت ترکیب طبقه بندها و انتخاب وزن هایشان استفاده کردند. فرض اصلی در این مقاله این است که قابلیت پیش بینی هر یک از طبقه بندها در میان سایر طبقه بندها متفاوت است. در این مقاله با پیاده سازی این

[1] Discrete Multi wavelet Transform

روش و استفاده از ترکیب تعدادی شبکه پرسپترون چند لایه[1] (MLP) بر روی داده های ارقام دستنویس فارسی در بازشناسی به نرخ ۹۸,۲۷٪ دست یافتند [۷].

۸. در مقاله ای آقایان نحوی، کیانی و ابراهیم پور روش جدیدی بر مبنای تبدیل گسسته کسینوسی برای استخراج بردار ویژگی از تصاویر گرادیان ارائه دادند که پس ازاعمال الگوریتم گرادیان، ضرایب تبدیل گسسته کسینوسی تصاویر گرادیان محاسبه و ضرایب فرکانس پایین آن به عنوان ویژگی‌های تصویر استفاده شده است. آزمایشهای انجام شده بر روی مجموعه داده ارقام دستنویس فارسی نشان می‌دهد که روش پیشنهادی خطای نرخ بازشناسی را به میزان ۳۰ درصد کاهش می‌دهد. آزمایش‌ها بر روی مجموعه داده هدی انجام شده که شامل ۶۰۰۰۰ نمونه آموزش و ۲۰۰۰۰ نمونه آزمایش است که در بازشناسی به نرخ ۹۹,۱۵٪ دست یافته اند [۸].

۱-۵- ساختار پایان نامه

پس از مقدمه و مروری بر مطالعات پیشین، در فصل دوم این پایان نامه بحث استخراج و انتخاب ویژگی مطرح شده و در فصل سوم مروری بر الگوریتم‌های بهینه‌سازی وچگونگی استفاده از الگوریتم بهینه سازی توده ذرات و ژنتیک در انتخاب ویژگی شده است.

فصل چهارم به بررسی طبقه‌بند ماشین بردار پشتیبان خطی و غیرخطی می‌پردازیم و عملکرد ماشین بردار پشتیبان در طبقه‌بندی های چند کلاسه نیز مورد بررسی قرار می‌گیرد.

فصل پنجم به الگوریتم پیشنهادی جهت استخراج و انتخاب ویژگی اختصاص داده شده است.

فصل ششم به نتایج شبیه‌سازی ساختار پیشنهاد شده در فصل پنجم را برای تشخیص ارقام دستنویس فارسی و داده‌های پایگاه UCI، چهره ORL، اثر انگشت و در انتها به بررسی کلی مطالب می‌پردازیم.

فصل هفتم به جمع بندی ونتیجه‌گیری از کار انجام شده به همراه پیشنهاد برای ادامه کاراختصاص یافته است.

[1] Multi-Layer Perceptron

فصل ۲

استخراج و انتخاب ویژگی

۲-۱- استخراج ویژگی

استخراج ویژگی فرآیندی است که در آن با انجام عملیاتی بر روی داده‌ها، ویژگی‌های بارز و تعیین کننده آن مشخص می‌شود. هدف از استخراج ویژگی این است که داده‌های خام به شکل قابل استفاده‌تری برای پردازش‌های آماری بعدی در آیند. برای مثال در پردازش تصویر برای اینکه از روی الگوهای یک تصویر هویت آن تصویر مشخص شود باید یک سری مشخصات عام و یا خاص از دل تصویر بیرون کشیده شود. به این کار استخراج ویژگی گفته می‌شود. روشهای مختلف استخراج ویژگی بنابر کاربردهایشان ممکن است یک یا چند کار زیر را انجام دهند.

۱. حذف نویز داده‌ها.
۲. جداسازی اجزای مستقل داده‌ها.
۳. کاهش ابعاد برای تولید بازنمایی مختصر تر از داده‌ها.
۴. افزایش بعد برای تولید باز نمایی جدایی پذیرتر از داده‌ها.

از جمله روشهای استخراج ویژگی متداول می‌توان به روشهای آماری و ساختاری و روشهای مبتنی بر تبدیلات اشاره نمود.

عموماً استخراج ویژگی در بازشناسی الگو بعد از مرحله پیش‌پردازش (حذف نویز، نازک سازی، نرمال سازی اندازه و اوریب شدگی و ...) قرار می‌گیرد. این مرحله یکی از مهمترین مراحل در سیستم‌های OCR[1] و بازشناسی الگو است و مستقیما برروی کیفیت بازشناسی تأثیر دارد. در استخراج ویژگی به هر الگوی ورودی یا بسته به ورودی سیستم بازشناسی، یک بردار ویژگی یا یک کد نسبت داده می‌شود که معرف آن الگو در فضای ویژگی است و آن را از دیگر الگوها متمایز می‌سازد.

در انتخاب بردارهای ویژگی لازم است موارد زیر مورد توجه قرار گیرد:

۱. بردار ویژگی هر الگو باید تا حد امکان از بردارهای ویژگی دیگر الگو ها متمایز باشد (یعنی فاصله بین بردارهای ویژگی در فضای ویژگی حداکثر باشد).

[1] Optical character recognition

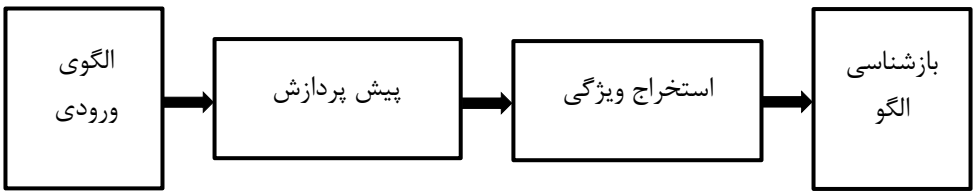
۲. بردارهای ویژگی الگوها باید تا بیشترین حد ممکن، خصوصیات شکل و ساختار الگو را از تصویر آنها استخراج نماید.

۳. تا حد ممکن نسبت به نویز، تغییر اندازه و نوع فونت، چرخش و دیگر تغییرات احتمالی الگوها دارای ثبات باشد.

۴. شرایط نوع و خصوصیات الگوهای ورودی در انتخاب بردارهای ویژگی اثر می‌گذارد. به عنوان مثال باید توجه نمود که آیا حروف یا الگویی که بایستی تشخیص داده شود جهت و اندازه مشخصی دارند یا خیر، دستنوشته هستند یا چاپی، یا اینکه تا چه حد به نویز آغشته شده‌اند.

۵. در مورد حروفی که با چندین شکل نوشته می‌شوند لازم است که بیش از یک کلاس الگو به یک کاراکتر خاص تعلق یابد.

همان‌طور که گفته شد استخراج ویژگی یا بازنمایی یک مرحله بسیار مهم در حصول راندمان مناسب برای سیستم‌های بازشناسی است. ولی برای دست یابی به عملکرد بهینه لازم است که دیگر مراحل نیز بهینه گردند و باید توجه نمود که این مراحل مستقل از هم نیستند. یک روش خاص استخراج ویژگی‌ها، طبیعت خروجی مرحله پیش‌پردازش را به ما دیکته می‌کند یا حداقل ما را در انتخابمان محدود می‌سازد. در **شکل ۲-۱** مکان قرارگیری مرحله استخراج ویژگی در سیستم بازشناسی الگو نمایش داده شده است.



شکل ۲-۱ : نمودار بلوکی یک سیستم بازشناسی الگوی ساده

در این پایان‌نامه ما بر روی داده‌های ارقام دستنویس فارسی و چهره پایگاه داده ORL، الگوریتم پیشنهادی جدیدی جهت استخراج ویژگی، پیاده نموده‌ایم که در فصل ۵ و ۶ بطور مفصل شرح داده شده است.

۲-۲-انتخاب ویژگی

پیشرفتهای بوجود آمده در جمع آوری داده و قابلیتهای ذخیره سازی در طی دهه‌های اخیر باعث شده در بسیاری از علوم با حجم بزرگی از اطلاعات روبرو شویم. محققان در زمینه‌های مختلف مانند مهندسی، ستاره شناسی، زیست شناسی و اقتصاد هر روز با مشاهدات بیشتر و بیشتری روبرو می‌شوند. در مقایسه با بسترهای داده‌ای قدیمی و کوچکتر، بسترهای داده‌ای امروزی چالشهای جدیدی در تحلیل داده‌ها بوجود آورده‌اند. روشهای آماری سنتی به دو دلیل امروزه کارائی خود را از دست داده‌اند. علت اول افزایش تعداد مشاهدات[1] است، و علت دوم که از اهمیت بالاتری برخوردار است افزایش تعداد متغیرهای مربوط به یک مشاهده می‌باشد. تعداد متغیرهایی که برای هر مشاهده باید اندازه‌گیری شود ابعاد داده نامیده می‌شود. عبارت متغیر[2] بیشتر در آمار استفاده می‌شود در حالی که در علوم کامپیوتر و یادگیری ماشین بیشتر از عبارات ویژگی[3] استفاده می‌گردد.

داده‌های که دارای ابعاد زیاد هستند علیرغم فرصتهایی که به وجود می‌آورند، چالشهای محاسباتی زیادی را ایجاد می‌کنند. یکی از مشکلات داده‌های با ابعاد زیاد این است که در بیشتر مواقع تمام ویژگی-های داده‌ها برای یافتن دانشی که در داده‌ها نهفته است مهم و حیاتی نیستند. به همین دلیل در بسیاری از زمینه‌ها کاهش ابعاد داده یکی از مباحث قابل توجه باقی مانده است.

روشهای کاهش ابعاد داده به دو دسته تقسیم می‌شوند [۹]:

۱. **روشهای مبتنی بر استخراج ویژگی**: این روشها یک فضای چند بعدی را به یک فضای با ابعاد کمتر نگاشت می‌کنند. در واقع با ترکیب مقادیر ویژگیهای موجود، تعداد کمتری ویژگی بوجود می‌آورند بطوریکه این ویژگیها دارای تمام (یا بخش اعظمی از) اطلاعات موجود در ویژگی‌های اولیه باشند.

[1] observations
[2] variable
[3] Feature

۲. **روشهای مبتنی بر انتخاب ویژگی**: این روشها سعی می‌کنند با انتخاب زیرمجموعه‌ای از ویژگی‌های اولیه، ابعاد داده‌ها را کاهش دهند. در پاره‌ای از اوقات تحلیلهای داده‌ای نظیر طبقه-بندی برروی فضای کاسته شده نسبت به فضای اصلی بهتر عمل می‌کند.

۲-۲-۱- روشهای مبتنی بر استخراج ویژگی

این روشها به دو دسته‌ی خطی و غیرخطی تقسیم می‌شوند. روشهای خطی که ساده‌ترند و فهم آنها راحت‌تر است بدنبال یافتن یک زیرفضای تخت عمومی[۱] هستند. اما روشهای غیرخطی که مشکل‌ترند و تحلیل آنها سخت‌تر است بدنبال یافتن یک زیرفضای تخت محلی[۲] می‌باشند [۱۰].

از روشهای خطی می‌توان به DFT، DWT، PCA[۳] و FA[۴] اشاره کرد که آنها را بطور مختصر در ادامه توضیح خواهیم داد. روشهای دیگر خطی عبارتند از:

۱. PP[۵] : برخلاف روشهای PCA و FA می‌تواند اطلاعات بالاتر از مرتبه‌ی دوم را ترکیب نماید. بنابراین روش مناسبی است برای بسترهای داده‌ای غیر گاوسی.

۲. ICA[۶] : این روش نیز یک نگاشت خطی انجام می‌دهد اما بردارهای این نگاشت لزوماً بر یکدیگر عمود نیستند، در حالی که در روشهای دیگر مانند PCA این بردارها بر هم عمودند.

۳. RP[۷] : یک روش ساده و در عین حال قدرتمند برای کاهش ابعاد داده است که از ماتریسهای نگاشت تصادفی برای نگاشت داده‌ها به یک فضای با ابعاد کمتر استفاده می‌کند.

از روشهای غیرخطی نیز می‌توان به موارد زیر اشاره کرد:

۱. Principal Curves
۲. Self-Organizing Maps
۳. Vector Quantization
۴. Genetic and Evolutionary Algorithms

[1] Global flat subspace
[2] Locally flat subspace
[3] Principal Component Analysis
[4] Factor Analysis
[5] Projection Pursuit
[6] Independent Component Analysis
[7] Random Projection

۵. Regression

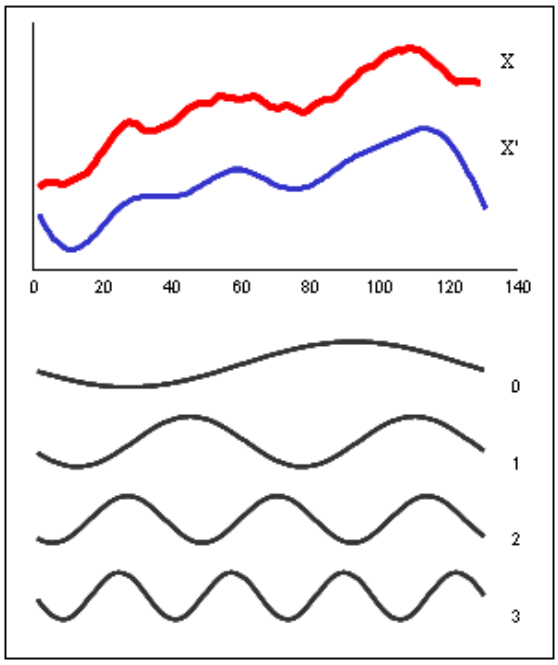
مسأله‌ی کاهش ابعاد داده را بطور ریاضی می‌توان به اینصورت بیان کرد: یک متغیر تصادفی p بعدی $X = (X_1, X_2, \dots, X_p)^T$ داریم. می‌خواهیم متغیر k بعدی $S = (S_1, S_2, \dots, S_k)^T$ را به گونه‌ای پیدا کنیم که اولاً $k \leq p$ باشد و ثانیاً S محتویاتی که در x وجود دارد را بر اساس معیاری خاص دارا باشد. روشهای خطی سعی می‌کنند هر یک از این k مؤلفه را از ترکیب خطی p مؤلفه‌ی اولیه بدست آورند. با استفاده از معادله (۲-۱).

$$S_i = w_{i,1}x_1 + \cdots w_{i,p}x_p \quad , \quad For \quad i = 1, \dots, k \quad or$$
$$S = W.x \qquad (۱-۲)$$

که $W_{k\times p}$ ماتریس وزنهای نگاشت خطی می‌باشد.

۲-۲-۱-۱- تبدیل فوریه گسسته DFT

در بسیاری از کاربردها مرسوم است که از ترکیب توابع پایه‌ای برای تقریب یک تابع استفاده شود. به عنوان مثال هر تابع پیوسته را می‌توان توسط مجموعه‌ای از توابع چند جمله‌ای نمایش داد. تبدیل فوریه نوعی تبدیل است که یک تابع را بصورت توابع پایه‌ای سینوسی که هر کدام در مقادیری ضرب شده‌اند نشان می‌دهد **شکل ۲-۲**. از تبدیل فوریه در بسیاری از زمینه‌های علمی مانند فیزیک، هندسه، آمار و پردازش سیگنال استفاده می‌شود[۱۱].



شکل ۲-۲: عملکرد تبدیل فوریه در تجزیه توابع به توابع پایه ای سینوسی را نشان می‌دهد

تبدیل فوریه یک تبدیل برگشت پذیر است. یعنی پس از تبدیل تابعی توسط آن دوباره می‌توان آن تابع را بازسازی نمود. این تبدیل می‌تواند به دو صورت پیوسته یا گسسته انجام شود. در کامپیوتر و بخصوص در پردازش سیگنال معمولاً از تبدیل فوریه‌ی گسسته (DFT) استفاده می‌شود. خوشبختانه الگوریتم‌های سریعی تحت عنوان FFT[1] برای تبدیل فوریه‌ی گسسته به وجود آمده است.

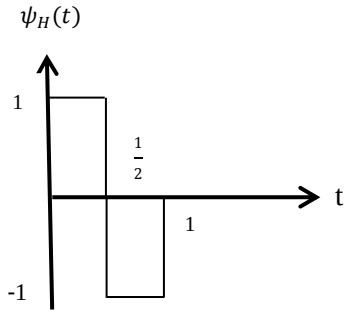
۲-۲-۱-۲- تبدیل موجک گسسته DWT

تبدیل DWT برای اولین بار توسط شخصی به نام Alfred Haar بوجود آمد. تاکنون نسخه‌های مختلفی برای DWT ارائه شده است که می‌توان به موارد زیر اشاره کرد:

۱. Haar Wavelet
۲. Newland Transform
۳. Undecimated Wavelet Transform

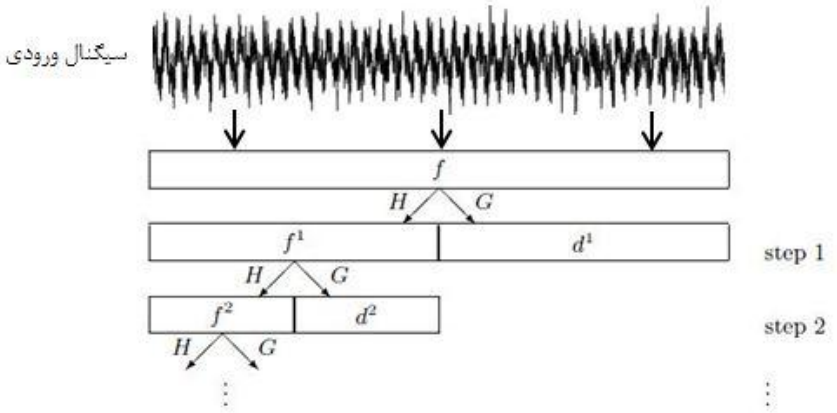
این تبدیل نیز همانند تبدیل فوریه بسیار پرکاربرد است و در بسیاری از زمینه‌های علوم و مهندسی مورد توجه قرار گرفته است. تبدیل Haar شکل ۲-۳ بدلیل سادگی در پیاده سازی و سرعت اجرای بالا، از محبوبیت بیشتری نسبت به سایر نسخه‌های DWT برخوردار است[۱۱].

[1] Fast Fourier Transform



شکل ۲-۳: تابع تبدیل هار

این تبدیل به اینصورت است که یک توالی به طول 2^n در ورودی داریم. این اعداد بصورت جفت جفت با هم جمع شده و این حاصل جمع‌ها به مرحله‌ی بعد فرستاده می‌شوندکه دنباله بدست آمده را سیگنال تقریب می‌نامیم. همچنین اختلاف هر جفت نیز محاسبه و ذخیره می‌شود و این دنباله را جزئیات می-نامیم. دوباره این مرحله تکرار می‌شود با این تفاوت که در ورودی، سیگنال تقریب قرار می‌گیرد. این فرایند بطور بازگشتی تکرار می‌شود تا در نهایت یک عدد که حاصل جمع کل اعداد است بدست آید. این عدد به همراه 2^n-1 اختلاف جفتها که در مراحل مختلف الگوریتم محاسبه شده بعنوان خروجی این تبدیل بازگردانده می‌شود. شکل ۲-۴ مراحل اعمال تبدیل موجک بر روی سیگنال ورودی را نمایش می-دهد.

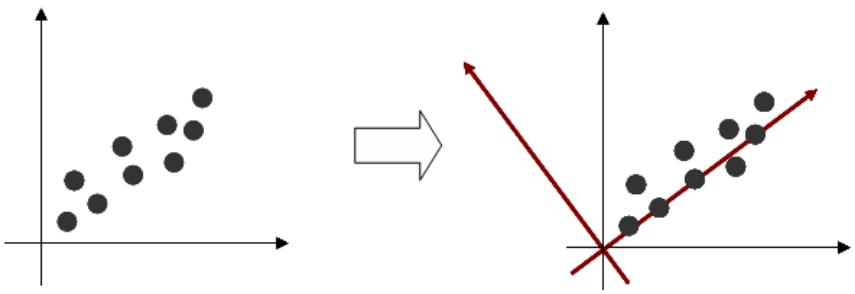


شکل ۲-۴: مراحل اعمال تبدیل موجک بر روی سیگنال ورودی

در شکل ۲-۴ سیگنال ورودی در ابتدا به عنوان سیگنال تقریب اولیه f انتخاب می‌شود سپس با استفاده از دو فیلتر H وG به ترتیب فیلترهای پایین و بالا گذر سیگنالهای تقریب f^1 و جزئیات d^1 را تولید می-کنند و این فرآیند تا تعداد مراحلی که کاربر انتخاب می‌نماید ادامه دارد.

۲-۲-۱-۳- روش PCA

تکنیک PCA بهترین روش برای کاهش ابعاد داده به صورت خطی می‌باشد. یعنی با حذف ضرایب کم‌اهمیت بدست آمده از این تبدیل، اطلاعات از دست رفته نسبت به روشهای دیگر کمتر است. البته کاربرد PCA محدود به کاهش ابعاد داده نمی‌شود و در زمینه‌های دیگری مانند شناسایی الگو و تشخیص چهره نیز مورد استفاده قرار می‌گیرد. در این روش محورهای مختصات جدیدی برای داده‌ها تعریف شده و داده‌ها براساس این محورهای مختصات جدید بیان می‌شوند. اولین محور باید در جهتی قرار گیرد که واریانس داده‌ها ماکسیمم شود (یعنی در جهتی که پراکندگی داده‌ها بیشتر است). دومین محور باید عمود بر محور اول به گونه‌ای قرار گیرد که واریانس داده‌ها ماکسیمم شود. به همین ترتیب محورهای بعدی عمود بر تمامی محورهای قبلی به گونه‌ای قرار می‌گیرند که داده‌ها در آن جهت دارای بیشترین پراکندگی باشند. درشکل **۲-۵** این مطلب برای داده‌های دو بعدی نشان داده شده است[۱۲].



شکل ۲-۵: انتخاب محورهای جدید برای داده‌های دو بعدی

روش PCA به نامهای دیگری چون SVD[1]، KLT[2]، HT[3] و EOF[4] معروف است.

[1] Singular Value Decomposition
[2] Karhunen Loeve Transform
[3] Hotelling Transform
[4] Empirical Orthogonal Function

۲-۲-۱-۴- روش FA

FA یکی از روشهای آماری است که می‌تواند چندین متغیر تصادفی مشاهده شده را توسط تعداد کمتری متغیر تصادفی (که در داده‌ها پنهان هستند) نمایش دهد. این متغیرهای تصادفی پنهان، فاکتور نامیده می‌شوند. این روش سعی می‌کند متغیرهای تصادفی مشاهده شده را توسط ترکیب خطی فاکتورها بعلاوه‌ی مقداری خطا مدلسازی نماید. روش FA از رشته هوش سنجی سرچشمه گرفته و در زمینه‌های علوم اجتماعی، بازاریابی، مدیریت تولید، تحقیق در عملیات و علوم کاربردی دیگر که با حجم بزرگی از داده‌ها سروکار دارند مورد استفاده قرار گرفته است. این روش برای اولین بار حدود ۱۰۰ سال پیش توسط یک روانشناس به نام Charles Spearman ابداع شد. این شخص نظریه‌ای به نام g-theory ارائه داد و در آن ادعا کرد که تمام توانمندیهای ذهنی افراد مانند مهارتهای ریاضی، مهارتهای هنری، دایره لغات، توانایی استدلالهای منطقی و غیره را می‌توان توسط یک فاکتور به نام هوش عمومی[1] بیان کرد. البته این نظریه امروزه رد شده و تحقیقات انجام شده نشان می‌دهد که توانمندیهای ذهنی حداقل از سه فاکتور به نامهای توانائی ریاضی، توانائی شفاهی و توانائی منطقی تشکیل شده است. روانشناسان زیادی بر این باورند که علاوه بر این سه فاکتور، فاکتورهای دیگری وجود دارد که می‌تواند بر توانمندیهای ذهنی افراد تأثیرگذار باشد[۱۱].

۲-۲-۲- روشهای مبتنی بر انتخاب ویژگی

مسأله انتخاب ویژگی، یکی از مسائلی است که در مبحث یادگیری ماشین و همچنین شناسائی آماری الگو مطرح است. این مسأله در بسیاری از کاربردها مانند طبقه بندی اهمیت به سزائی دارد، زیرا در این کاربردها تعداد زیادی ویژگی وجود دارد، که بسیاری از آنها یا بلااستفاده هستند و یا اینکه بار اطلاعاتی چندانی ندارند. حذف نکردن این ویژگی‌ها مشکلی از لحاظ اطلاعاتی ایجاد نمی‌کند ولی بار محاسباتی را برای کاربرد مورد نظر بالا می‌برد. علاوه بر این باعث می‌شود که اطلاعات غیر مفید زیادی را به همراه داده‌های مفید ذخیره کنیم.

[1] General Intelligence

برای مسأله انتخاب ویژگی، راه حل‌ها و الگوریتم‌های فراوانی ارائه شده است که بعضی از آنها قدمت سی یا چهل ساله دارند. مشکل بعضی از الگوریتم‌ها در زمانی که ارائه شده بودند، بار محاسباتی زیاد آنها بود، اگر چه امروزه با ظهور کامپیوترهای سریع و منابع ذخیره سازی بزرگ این مشکل، به چشم نمی‌آید ولی از طرف دیگر، مجموعه‌های داده‌ای بسیار بزرگ برای مسائل جدید باعث شده است که همچنان پیدا کردن یک الگوریتم سریع برای این کار مهم باشد.

در ادامه ما ابتدا تعاریفی که برای انتخاب ویژگی ارائه شده‌اند را ارائه می‌دهیم. سپس روش‌های مختلف برای این مسأله را بر اساس نوع و ترتیب تولید زیرمجموعه ویژگی‌های کاندید و همچنین نحوه ارزیابی این زیرمجموعه‌ها دسته‌بندی می‌کنیم.

۲-۲-۳- تعاریف انتخاب ویژگی

مسأله انتخاب ویژگی بوسیله نویسندگان مختلف، از دیدگاه‌های متفاوتی مورد بررسی قرار گرفته است. هر نویسنده نیز با توجه به نوع کاربرد، تعریفی را از آن ارائه داده است. در ادامه چند مورد از این تعاریف را بیان می‌کنیم[۱۳]:

۱. **تعریف ایده‌آل:** پیدا کردن یک زیرمجموعه با حداقل اندازه ممکن، برای ویژگی‌ها است، که برای هدف مورد نظر اطلاعات لازم و کافی را در بر داشته باشد. بدیهی است که هدف تمام الگوریتم‌ها و روش‌های انتخاب ویژگی همین زیر مجموعه است.

۲. **تعریف کلاسیک:** انتخاب یک زیرمجموعه M عنصری از میان N ویژگی، بطوریکه $M < N$ باشد و همچنین مقدار یک تابع معیار[1] برای زیرمجموعه مورد نظر، نسبت به سایر زیرمجموعه‌های هم‌اندازه دیگر بهینه باشد. این تعریفی است که Fukunaga و Narenda در سال ۱۹۷۷ ارائه داده‌اند.

[1] Criterion function

۳. **افزایش دقت پیشگوئی:** هدف انتخاب ویژگی این است که یک زیرمجموعه از ویژگی‌ها برای افزایش دقت پیشگوئی انتخاب شوند. به عبارت دیگر کاهش اندازه ساختار بدون کاهش قابل ملاحظه در دقت پیشگوئی طبقه‌بندی کننده‌ای که با استفاده از ویژگی‌های داده شده بدست می‌آید.

۴. **تخمین توزیع کلاس اصلی:** هدف از انتخاب ویژگی این است که یک زیرمجموعه کوچک از ویژگی‌ها انتخاب شوند، توزیع ویژگی‌هایی که انتخاب می‌شوند، بایستی تا حد امکان به توزیع کلاس اصلی با توجه به تمام مقادیر ویژگی‌های انتخاب شده نزدیک باشد.

در اصل روش‌های مختلف انتخاب ویژگی، تلاش می‌کنند تا از میان ۲^N زیر مجموعه کاندید، بهترین زیرمجموعه را پیدا کنند. در تمام این روشها بر اساس کاربرد و نوع تعریف، زیر مجموعه‌ای به عنوان جواب انتخاب می‌شود، که بتواند مقدار یک تابع ارزیابی را بهینه کند. با وجود اینکه هر روشی سعی می‌کند که بتواند، بهترین ویژگی‌ها را انتخاب کند، اما با توجه به وسعت جواب‌های ممکن، و اینکه این مجموعه‌های جواب بصورت توانی با N افزایش پیدا می‌کنند، پیدا کردن جواب بهینه مشکل و در N های متوسط و بزرگ بسیار پر هزینه است.

بطور کلی روش‌های مختلف انتخاب ویژگی را بر اساس نوع جستجو به دسته‌های مختلفی تقسیم بندی می‌کنند. در بعضی از روش‌ها تمام فضای ممکن جستجو می‌گردد. در سایر روش‌ها که می‌تواند مکاشفه‌ای و یا جستجوی تصادفی باشد، در ازای از دست دادن مقداری از کارآئی، فضای جستجو کوچکتر می‌شود. برای اینکه بتوانیم تقسیم بندی درستی از روش‌های مختلف انتخاب ویژگی داشته باشیم، به این صورت عمل می‌کنیم که فرآیند انتخاب ویژگی در تمامی روش‌ها را به این بخش‌ها تقسیم می‌کنیم:

۲-۲-۴- فرآیند انتخاب ویژگی

۱. **تابع تولید کننده**[1]: این تابع زیر مجموعه‌های کاندید را برای روش مورد نظر پیدا می‌کند.

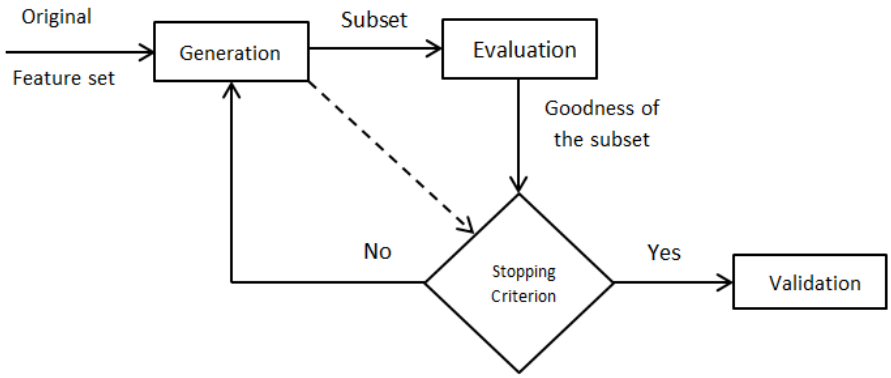
[1] Generation procedure

۲. **تابع ارزیابی**[1]: زیرمجموعه مورد نظر را بر اساس روش داده شده، ارزیابی و یک عدد به عنوان میزان خوبی روش باز می‌گرداند. روش‌های مختلف سعی در یافتن زیرمجموعه‌ای دارند که این مقدار را بهینه کند.

۳. **شرط خاتمه:** برای تصمیم‌گیری در مورد زمان توقف الگوریتم.

۴. **تابع تعیین اعتبار**[2]: تصمیم می‌گیرد که آیا زیر مجموعه انتخاب شده معتبر است یا خیر؟

در ادامه در **شکل ۲-۶** فرآیند انتخاب ویژگی بر اساس توضیحات داده شده نشان داده شده است.



شکل ۲-۶: فرآیند انتخاب ویژگی

تابع تولید کننده در واقع تابع جستجو است. این تابع زیرمجموعه‌های مختلف را به ترتیب تولید می‌کند، تا بوسیله تابع ارزیابی، مورد ارزیابی قرا بگیرد. تابع تولید کننده از یکی از حالت‌های زیر شروع به کار می‌کند:

۱) بدون ویژگی

۲) با مجموعه تمام ویژگی‌ها

۳) با یک زیرمجموعه تصادفی

در حالت اول ویژگی‌ها به ترتیب به مجموعه اضافه می‌شوند و زیرمجموعه‌های جدید را تولید می‌کنند. این عمل آنقدر تکرار می‌شود تا به زیر مجموعه مورد نظر برسیم. به اینگونه روش‌ها، روشهای پائین به بالا می‌گویند.

[1] Evaluation function
[2] Validation procedure

در حالت دوم از یک مجموعه شامل تمام ویژگی‌ها، شروع می‌کنیم و به مرور و در طی اجرای الگوریتم، ویژگی‌ها را حذف می‌کنیم، تا به زیرمجموعه دلخواه برسیم. روش‌هایی که به این صورت عمل می‌کنند، روشهای بالا به پائین نام دارند.

یک تابع ارزیابی، میزان خوب بودن یک زیرمجموعه تولید شده را بررسی کرده و یک مقدار به عنوان میزان خوب بودن زیرمجموعه مورد نظر باز می‌گرداند. این مقدار با بهترین زیرمجموعه قبلی مقایسه می-شود. اگر زیرمجموعه جدید، بهتر از زیرمجموعه‌های قدیمی باشد، زیرمجموعه جدید به عنوان زیرمجموعه بهینه، جایگزین قبلی می‌شود.

باید توجه داشت که بدون داشتن یک شرط خاتمه مناسب، فرآیند انتخاب ویژگی ممکن است، برای همیشه درون فضای جستجو، برای یافتن جواب سرگردان بماند. شرط خاتمه می‌تواند بر پایه تابع تولید کننده باشد، مانند:

۱) هر زمان که تعداد مشخصی ویژگی انتخاب شدند.

۲) هر زمان که به تعداد مشخصی تکرار رسیدیم.

و یا اینکه بر اساس تابع ارزیابی انتخاب شود، مانند:

۱) وقتی که اضافه یا حذف کردن ویژگی، زیرمجموعه بهتری را تولید نکند.

۲) وقتی که به یک زیرمجموعه بهینه بر اساس تابع ارزیابی برسیم.

تابع تعیین اعتبار جزئی از فرآیند انتخاب ویژگی نیست، اما در عمل بایستی یک زیرمجموعه ویژگی را در شرایط مختلف تست کنیم تا ببینیم که آیا شرایط مورد نیاز، برای حل مسأله مورد نظر ما را دارد یا نه؟ برای اینکار می‌توانیم از داده‌های نمونه‌برداری شده و یا مجموعه داده های شبیه سازی شده استفاده کنیم.

۲-۲-۵- توابع تولید کننده

اگر تعداد کل ویژگی‌ها برابر N باشد، تعداد کل زیرمجموعه‌های ممکن برابر 2^N می‌شود. این تعداد برای N های متوسط هم خیلی زیاد است. بر اساس نحوه جستجو در میان این تعداد زیرمجموعه، روشهای مختلف انتخاب ویژگی را می‌توان به سه دسته زیر تقسیم‌بندی نمود:

۱) جستجوی کامل

۲) جستجوی مکاشفه‌ای

۳) جستجوی تصادفی

در ادامه به معرفی هر کدام از این دسته‌ها می‌پردازیم.

۲-۲-۵-۱- جستجوی کامل[1]

در روش‌هایی که از این نوع جستجو استفاده می‌کنند، تابع تولید کننده بر اساس تابع ارزیابی استفاده شده، تمام فضای جواب (زیرمجموعه‌های ممکن) را برای یافتن جواب بهینه جستجو می‌کند. البته Schlimmer استدلال آورده است که[۱۴]: "کامل بودن جستجو به این معنی نیست که جستجو باید جامع باشد".

توابع مکاشفه‌ای مختلف زیادی طراحی شده‌اند، تا جستجو را بدون از دست دادن شانس پیدا کردن جواب بهینه، کاهش دهند. اما با توجه به بزرگی فضای جستجو، $O(2^N)$، این روش‌ها باعث می‌شوند که فضای کمتری جستجو شود. روش‌ها و تکنیک‌های مختلفی برای اینکار استفاده شده‌اند، بعضی از آنها از تکنیک بازگشت به عقب[2] نیز در جریان کار استفاده کرده‌اند، مانند:

branch and bound، best first search و beam search.

[1] Complete Search
[2] Backtracking

۲-۲-۵-۲- جستجوی مکاشفه ای[1]

در روش‌های با این نوع جستجو، در هر بار اجرای الگوریتم، یک ویژگی از مجموعه ویژگی‌ها انتخاب شده، اضافه و یا از آن حذف می‌شود. به همین دلیل پیچیدگی زمانی آنها محدود و کمتر از $O(N^2)$ می‌باشد. در اینگونه موارد، اجرای الگوریتم خیلی سریع می‌باشد و پیاده سازی آنها نیز بسیار ساده است.

۲-۲-۵-۳- جستجوی تصادفی[2]

روش‌هایی که از این نوع جستجو استفاده می‌کنند، محدوده کمتری از فضای کل حالات را جستجو می‌کنند، که اندازه این محدوده به حداکثر تعداد تکرار الگوریتم بستگی دارد. در این روش‌ها پیدا شدن جواب بهینه به اندازه منابع موجود و زمان اجرای الگوریتم بستگی دارد. در هر بار تکرار، تابع تولید کننده تعدادی از زیرمجموعه‌های ممکن از فضای جستجو را به صورت تصادفی انتخاب می‌کند و در اختیار تابع ارزیابی قرار می‌دهد. تابع تولید کننده تصادفی پارامترهایی دارد که بایستی تنظیم شوند، تنظیم مناسب این پارامترها در سرعت رسیدن به جواب و پیدا شدن جواب‌های بهتر موثر است.

۲-۲-۶- توابع ارزیابی کننده

پیدا شدن یک زیرمجموعه بهینه از مجموعه ویژگی‌ها، به صورت مستقیم به انتخاب تابع ارزیابی بستگی دارد. چرا که اگر تابع ارزیابی به زیرمجموعه ویژگی بهینه یک مقدار نامناسب نسبت دهد، این زیرمجموعه هیچگاه بعنوان زیرمجموعه بهینه انتخاب نمی‌شود. مقادیری که توابع ارزیابی مختلف به یک زیرمجموعه می‌دهند، با هم متفاوت است.

توابع ارزیابی را می‌توان به انواع مختلفی دسته‌بندی کرد. در اینجا ما دسته‌بندی‌ای که توسط Dash و Liu ارائه شده است را بیان می‌کنیم[۱۳]. آنها این معیارها را به پنج دست تقسیم کرده‌اند:

[1] Heuristic Search
[2] Random Search

۱. **معیارهای مبتنی بر فاصله**[1]: در این معیارها، مثلاً برای یک مسأله دو کلاسه، یک ویژگی یا یک مجموعه ویژگی مثل X بر یک ویژگی یا یک مجموعه ویژگی دیگر مثل Y ارجحیت دارد، اگر که با آن مجموعه ویژگی مقادیر بزرگتری برای اختلاف بین احتمالات شرطی دو کلاس داشته باشیم. نمونه‌ای از این معیارها همان معیار فاصله اقلیدسی می‌باشد.

۲. **معیارهای مبتنی بر اطلاعات**[2]: این معیارها میزان اطلاعاتی را که بوسیله یک ویژگی بدست می‌آید را در نظر می‌گیرند. ویژگی X در این روش‌ها بر ویژگی Y اولویت دارد، اگر اطلاعات بدست آمده از ویژگی X بیشتر از اطلاعاتی باشد، که از ویژگی Y بدست می‌آید. نمونه‌ای از این معیارها همان معیار آنتروپی می‌باشد.

۳. **معیارهای مبتنی بر وابستگی**[3]: این معیارها که با عنوان معیارهای همبستگی[4] نیز شناخته می‌شوند، قابلیت پیشگوئی مقدار یک متغیر بوسیله یک متغیر دیگر را اندازه‌گیری می‌کنند. ضریب[5] یکی از معیارهای وابستگی کلاسیک است و می‌توانیم آنرا برای یافتن همبستگی بین یک ویژگی و یک کلاس به کار ببریم. اگر همبستگی ویژگی X با کلاس C بیشتر از همبستگی ویژگی Y با کلاس C باشد، در اینصورت ویژگی X بر ویژگی Y برتری دارد. با یک تغییر کوچک، می‌توانیم وابستگی یک ویژگی با ویژگی‌های دیگر را اندازه‌گیری کنیم. این مقدار درجه افزونگی این ویژگی را نشان می‌دهد. همه توابع ارزیابی بر پایه معیار وابستگی را می‌توانیم بین دو دسته معیارهای مبتنی بر فاصله و اطلاعات تقسیم کنیم. اما به خاطر اینکه این روش‌ها از یک دید دیگر به مسأله نگاه می‌کنند، این کار را انجام نمی‌دهیم.

۴. **معیارهای مبتنی بر سازگاری**[6]: این معیارها جدیدتر هستند و اخیراً توجه بیشتری به آنها شده است. این معیارها خصوصیات متفاوتی نسبت به سایر معیارها دارند، زیرا که به شدت به داده‌های آموزشی تکیه دارند و در انتخاب یک زیرمجموعه از ویژگی‌ها تمایل دارند، که مجموعه ویژگی-

[1] Distance Measures
[2] Information Measures
[3] Dependence Measures
[4] Correlation
[5] Coefficient
[6] Consistency Measures

های کوچکتری را انتخاب کنند. این روش‌ها زیرمجموعه‌های با کمترین اندازه را بر اساس از دست دادن یک مقدار قابل قبول سازگاری که توسط کاربر تعیین می‌شود، پیدا می‌کنند.

۵. **معیارهای مبتنی بر خطای طبقه بندی کننده**[1]: روش‌هایی که این نوع از تابع ارزیابی را استفاده می‌کنند، با عنوان (wrapper methods) شناخته می‌شوند. دقت عملکرد در این روش‌ها برای تعیین کلاسی که نمونه داده شده متعلق به آن است، برای نمونه‌های دیده نشده بسیار بالا است، اما هزینه‌های محاسباتی در آنها نیز نسبتاً زیاد است.

در جدول (۲-۱) در زیر مقایسه‌ای بین انواع مختلف تابع ارزیابی، صرف نظر از نوع تابع تولید کننده مورد استفاده، انجام شده است. پارامترهایی که برای مقایسه استفاده شده‌اند به صورت زیر می‌باشند:

۱. **عمومیت**[2]: اینکه بتوان زیرمجموعه انتخاب شده را برای طبقه‌بندی کننده‌های متفاوت به کار ببریم.

۲. **پیچیدگی زمانی**: زمان لازم برای پیدا کردن زیرمجموعه ویژگی جواب.

۳. **دقت**: دقت پیشگوئی با استفاده از زیرمجموعه انتخاب شده.

علامت (---) که در ستون آخر آمده است، به این معنی است که در مورد میزان دقت حاصل نمی‌توانیم مطلبی بگوئیم. بجز خطای طبقه‌بندی کننده، دقت سایر توابع ارزیابی به مجموعه داده مورد استفاده و طبقه بندی کننده‌ای که بعد از انتخاب ویژگی برای طبقه‌بندی کلاس‌ها استفاده می‌شود، بستگی دارد.

[1] Classifier Error Rate Measures
[2] Generality

جدول ۲-۱: مقایسه توابع ارزیابی مختلف

نوع تابع ارزیابی	عمومیت	پیچیدگی زمانی	دقت
معیار فاصله	دارد	پائین	---
معیار اطلاعات	دارد	پائین	---
معیار وابستگی	دارد	پائین	---
معیار سازگاری	دارد	متوسط	---
خطای طبقه بندی کننده	ندارد	بالا	خیلی زیاد

فصل ۳

مروری بر الگوریتم‌های بهینه‌سازی

۳-۱- مقدمه‌ای بر زندگی مصنوعی

اصطلاح زندگی مصنوعی[1] به نظم جدیدی اطلاق می‌شود که به بررسی مسائل علمی، مهندسی و اجتماعی به منظور ترکیب با رفتارهایی همانند زندگی می‌پردازند. بدین صورت که چگونه می‌توان با گسترش افق دید در حین مطالعه‌ی پدیده‌های زیستی و بدست آوردن مدلهای مشابه، از آنها در حل مسائل محاسباتی بهره جست[۱۵]. هم اکنون تعداد زیادی روش محاسباتی الهام گرفته شده از سیستم های زیستی وجود دارد. مانند شبکه‌های عصبی مصنوعی[2] (ANN) که مدل ساده شده‌ای از مغز انسان می‌باشد، الگوریتم ژنتیک که از روند تکامل انسان در طول نسلهای مختلف الهام گرفته شده است و الگوریتم کلونی مورچگان (ACO) که در واقع مدلی از رفتار و نحوه جستجوی مورچه‌ها برای یافتن غذا است. و الگوریتم بهینه سازی توده ذرات (PSO) که در این پایان‌نامه بیشتر مورد توجه است بر اساس حرکت موجودات واقعی مانند دسته پرندگان[3] یا گروه ماهی‌ها[4]، ابداع شده است در این فصل به معرفی الگوریتم ژنتیک، بهینه سازی توده ذرات، کلونی مورچگان و جستجوی ممنوعه می‌پردازیم.

۳-۲- الگوریتم GA

الگوریتم ژنتیک تکنیک جستجویی در علم رایانه برای یافتن راه‌حل تقریبی برای بهینه‌سازی و مسائل جستجو است. الگوریتم ژنتیک نوع خاصی از الگوریتم‌های تکامل است که از تکنیکهای زیست‌شناسی مانند وراثت و جهش استفاده می‌کند. الگوریتم ژنتیک روش یادگیری بر پایه تکامل بیولوژیک است. این روش در سال ۱۹۷۰ توسط John Holland معرفی گردید، این روش با نامEvolutionary Algorithms نیز خوانده می‌شود[۱۶]. الگوریتم ژنتیک به روش بهینه‌سازی الهام گرفته ازطبیعت جاندار است که می-توان در طبقه‌بندی‌ها از آن به عنوان یک روش عددی، جستجوی مستقیم و تصادفی معرفی کرد. این الگوریتم الگوریتمی مبتنی بر تکرار است و اصول اولیه آن از علم ژنتیک اقتباس شده است. این الگوریتم

[1] Artificial Life
[2] Artificial Neural Network
[3] Birds flock
[4] Fish School

در مسایل متنوعی نظیر بهینه سازی، شناسایی و کنترل سیستم، پردازش تصویر و مسایل ترکیبی، تعیین توپولوژی وآموزش شبکه عصبی مصنوعی و سیستمهای مبتنی بر تصمیم بکار می‌رود. چنانکه می‌دانیم علم ژنتیک علمی است که درباره چگونگی وراثت و انتقال صفحات بیولوژیکی از نسلی به نسل بعد صحبت می‌کند. عامل اصلی انتقال صفحات بیولوژیکی در موجودات زنده کروموزوم‌ها و ژن‌ها می‌باشد و نحوه عملکرد آنها به گونه‌ای است که در نهایت ژنها و کروموزوم‌های برتر و قوی‌تر باقی مانده و ژن‌های ضعیف‌تر از بین می‌روند. به عبارت دیگر نتیجه عملیات متقابل ژن‌ها و کروموزوم‌ها باقی ماندن موجودات اصلح و برتر می‌باشند. این الگوریتم برای بهینه سازی، جستجو و یادگیری ماشین مورد استفاده قرار می-گیرد[۱۷]. الگوریتم ژنتیک به دلیل تقلید نمودن از طبیعت دارای چند اختلاف اساسی با روشهای جستجوی مرسوم می‌باشدکه در ادامه به تعدادی اشاره شده است. الگوریتم ژنتیک بارشته‌های بیتی کار می‌کند که هر کدام از این رشته‌ها کل مجموعه متغیرها را نشان می‌دهد، حال آنکه بیشتر روشها بطور مستقل با متغیرهای ویژه برخورد می‌کنند. الگوریتم ژنتیک برای راهنمایی جهت جستجو، انتخاب تصادفی انجام می‌دهد که به این ترتیب به اطلاعات مشتق نیاز ندارد. در الگوریتم ژنتیک، روش‌های جستجو بر اساس مکانیزم انتخاب و ژنتیک طبیعی ذکر شده در بالا، عمل می‌نمایند. این الگوریتم‌ها مناسب‌ترین رشته‌ها رااز میان اطلاعات تصادفی سازمان‌دهی شده انتخاب می‌کنند. در هر نسل، یک گروه جدید رشته‌ها با استفاده از بهترین قسمت‌های دنباله‌های قبلی و بخش جدید اتفاقی برای رسیدن به یک جواب مناسب به وجود می‌آیند. با وجود اینکه الگوریتم‌ها تصادفی هستند، ولی در زمره الگوریتم‌های تصادفی ساده نیستند. آنها بطور کارآمدی به اکتشاف اطلاعات گذشته در فضای جستجو می‌پردازند تا در یک نقطه جستجوی جدیدی با پاسخ‌های بهتر، به سمت بهترین جواب پیش روند. هنگام پیش آمد سازی الگوریتم‌های ژنتیک عمل پیش آمد سازی ساده را نمی‌پیمایند بلکه آنها داده‌های پیشین را با تفکر انتخاب جستجوی جدید برای رسیدن پیشرفت مورد نظر، توأم می‌کنند. الگوریتم‌های ژنتیک در هر تکرار چند نقطه از فضای جستجو را در نظر می‌گیرند بنابراین شانس اینکه به یک ماکزیموم محلی همگرا شود کاهش می‌یابد[۱۷]. در بیشتر روشهای جستجوی مرسوم (روش گرادیان)، قاعده تصمیم حاکم به این صورت عمل می‌کند که از یک نقطه به نقطه دیگر حرکت می‌کند. این روشها می‌توانند در فضاهای

جستجو دارای چند نقطه بیشینه باشند. زیرا ممکن است آنها به یک ماکزیموم محلی همگرا شوند. لیکن الگوریتم ژنتیک جمعیت‌های کاملی از نقاط را تولید نموده سپس هرنقطه را به صورت انفرادی امتحان می‌کند و با ترکیب محتویات آنها، یک جمعیت جدید راکه شامل نقاط بهبود یافته است، تشکیل می‌دهد. صرف نظر از انجام یک جستجو، ملاحظه همزمان تعدادی نقطه در الگوریتم ژنتیک آنها را با ماشین‌های موازی قابل تطبیق می‌سازد، زیرا در اینجا تکامل هر نقطه یک فرآیند مستقل است. لذا الگوریتم ژنتیک فقط نیاز به اطلاعاتی در مورد کیفیت حل‌های ایجاد شده به وسیله هر مجموعه از متغیرها دارد. در صورتی که بعضی از روش‌های بهینه‌سازی نیاز به اطلاعات یا حتی نیاز به شناخت کامل از ساختمان مسأله و متغیرها دارند. چون الگوریتم ژنتیک نیاز به چنین اطلاعات مشخصی از مسأله ندارد بنابراین قابل انعطاف‌تر از بیشتر روش‌های جستجو است.

۳-۲-۱- ویژگی‌ها در الگوریتم GA

الگوریتم ژنتیک در مسائلی که فضای جستجوی بزرگی داشته باشند می‌تواند بکار گرفته شود. همچنین در مسائلی با فضای فرضیه پیچیده که تاثیر اجزا آن در فرضیه کلی ناشناخته باشد می‌توان از GA برای جستجو استفاده نمود. همچنین برای بهینه‌سازی گسسته بسیار مورد استفاده قرار می‌گیرد. الگوریتم‌های ژنتیک را می‌توان براحتی با صورت موازی اجرا نمود از اینرو می‌توان کامپیوترهای ارزان قیمت‌تری را به صورت موازی مورد استفاده قرار داد.

۳-۲-۲- مکانیزم الگوریتم GA

الگوریتم ژنتیک، به عنوان یک الگوریتم محاسباتی بهینه‌سازی، با در نظر گرفتن مجموعه‌ای از نقاط فضای جواب در هر تکرار محاسباتی، بنحو موثری نواحی مختلف فضای جواب را جستجو می‌کند. در مکانیزم جستجو گرچه مقدار تابع هدف تمام فضای جواب محاسبه نمی‌شود ولی مقدار محاسبه شده تابع هدف برای هر نقطه، در متوسط‌گیری آماری تابع هدف برای هر نقطه، در متوسط‌گیری آماری تابع هدف در کلیه زیر فضاهایی که آن نقطه بدان‌ها وابسته بوده، دخالت داده می‌شود و این زیر فضاها بطور موازی

از نظر تابع هدف متوسط‌گیری آماری می‌شوند. این مکانیزم را توازی ضمنی می‌گویند. این روش باعث می‌شود که جستجوی فضا به نواحی از آن که متوسط آماری تابع هدف در آنها زیاد بوده و امکان وجود نقطه بهینه مطلق در آن بیشتر است، سوق پیدا کند. چون در این روش بر خلاف روشهای تک مسیری، فضای جواب بطور همه جانبه جستجو می‌شود، و امکان کمتری برای همگرایی به یک نقطه بهینه محلی وجود خواهد داشت. امتیاز دیگر این الگوریتم آن است که هیچ محدودیتی برای تابع بهینه شونده مثل مشتق‌پذیری یا پیوستگی لازم ندارد و در روند جستجوی خود تنها به تعیین مقدار تابع هدف در نقاط مختلف نیاز دارد و هیچ اطلاعات کمکی دیگری، مثل مشتق تابع را استفاده نمی‌کند پس می‌توان در مسائل مختلف از خطی، پیوسته یا گسسته استفاده شود[۱۷].

در هر تکرار هر یک از رشته‌های موجود در جمعیت رشته‌ها، رمزگشایی شده و مقدار تابع هدف برای آن بدست می‌آید. بر اساس مقادیر بدست آمده تابع هدف در جمعیت رشته‌ها، به هر رشته یک عدد برازندگی نسبت داده می‌شود. این عدد برازندگی احتمال انتخاب را برای هر رشته تعیین خواهد کرد. بر اساس این احتمال انتخاب مجموعه‌ای از رشته‌ها انتخاب شده و با اعمال عملکردهای ژنتیکی روی آن‌ها رشته‌های جدیدی جایگزین رشته‌هایی از جمعیت اولیه می‌شوند. تا تعداد جمعیت رشته‌ها در تکرارهای محاسباتی مختلف ثابت باشد[۱۷].

مکانیزم تصادفی که روی انتخاب و حذف نقاط عمل می‌کند به گونه‌ای می‌باشد که رشته یا نقاطی که عدد برازندگی بیشتری دارند، احتمال بیشتری برای ترکیب و تولید نقاط جدید داشته و در مرحله جایگزینی نسبت به دیگر نقاط یا رشته‌ها مقاوم‌تر هستند. بدین لحاظ جمعیت دنباله‌ها در یک رقابت بر اساس تابع هدف در طی نسل‌های مختلف، کامل شده و متوسط مقدار تابع هدف در جمعیت رشته‌ها افزایش می‌یابد. بطور کلی در این الگوریتم ضمن آنکه در هر تکرار محاسباتی، توسط عملگرهای ژنتیکی نقاطی جدید از فضای جواب مورد جستجو قرار می‌گیرند توسط مکانیزم انتخاب، روند جستجو نواحی از فضا را که متوسط آماری تابع هدف در آنها بیشتر است، کنکاش می‌کند. بر اساس سیکل اجرایی فوق، در هر تکرار محاسباتی توسط عملگرهای ژنتیکی نقاط جدیدی از فضای جواب مورد جستجو قرار می‌گیرند.

در هرتکرار محاسباتی، سه عملگر اصلی روی رشته‌ها عمل کرده که این سه عملگر عبارتند از دو عملگر ژنتیکی و یک عملگر انتخاب تصادفی[۱۷].

در تولید مثل ابتدا ترکیب یا تغییر[۱] اتفاق می‌افتد سپس ژن‌های والدین برای ایجاد کروموزوم‌های جدید ترکیب می‌شوند و سپس جنین تشکیل شده دچار تغییر می‌شود. جهش به این معناست که عناصر DNA[۲] کمی تغییر پیدا می‌کنند. این تغییرات اغلب نتیجه نسخه برداری غلط از ژن‌های والدین است. میزان شایستگی[۳] موجود زنده به واسطه بقای آن اندازه‌گیری می‌شود. در الگوریتم ژنتیک، مجموعه‌ای از متغیرهای طراحی را توسط رشته‌هایی با طول ثابت[۴] یا متغیر[۵] کدگذاری می‌کنند که در سیستم‌های بیولوژیکی آنها را کروموزوم یا فرد[۶] می‌نامند. هر متغیر طراحی از چند حرف که ژن[۷] می‌نامند تشکیل شده. هر رشته یک نقطه پاسخ در فضای جستجو را نشان می‌دهد. به ساختمان رشته‌ها یعنی مجموعه‌ای از پارامترها که توسط یک کروموزوم خاص نمایش داده می‌شود (Genotype) می‌گویند و به مقدار رمز گشایی آن (Phenotype) می‌گویند. الگوریتم وراثتی فرآیندهای تکراری هستند، که هر مرحله تکراری را نسل و مجموعه‌های از پاسخ‌ها در هر نسل را جمعیت نامیده‌اند. الگوریتم ژنتیک، جستجوی اصلی را در فضای پاسخ به اجرا می‌گذارد. این الگوریتم‌ها با تولید نسل[۸] آغاز می‌شوند که وظیفه ایجاد مجموعه نقاط جستجوی اولیه به نام جمعیت اولیه[۹] را بر عهده دارند و بطور انتخابی یا تصادفی تعیین می‌شوند. از آنجایی که الگوریتم‌های ژنتیک برای هدایت عملیات جستجو به طرف نقطه بهینه، از روشهای آماری استفاده می‌نمایند درفرآیندی که به انتخاب طبیعی وابسته است، جمعیت موجود به تناسب برازندگی افراد آن برای نسل بعد انتخاب می‌شود. پس عملگرهای ژنتیکی شامل انتخاب[۱۰]، پیوند، جهش[۱۱]، و دیگر عملگرهای احتمالی اعمال شده و جمعیت جدید به وجود می‌آید. پس از آن جمعیت جدیدی جایگزین

[1] Crossover
[2] Deoxyribonucleic acid
[3] Fitness
[4] Fixed length
[5] Variable
[6] Individual
[7] Gene
[8] Seeding
[9] Initial Population
[10] Selection
[11] Mutation

جمعیت پیشین می‌شود و این چرخه ادامه می‌یابد[۱۶]. معمولا جمعیت جدید برازندگی بیشتری دارند و این بدان معناست که از نسلی به نسل دیگر جمعیت بهبود می‌یابد و هنگامی جستجو نتیجه بخش خواهد بود که به حداکثر نسل ممکن رسیده باشیم و یا همگرایی حاصل شده باشد و یا معیارهای توقف برآورده شده باشند.

۳-۲-۳- عملگرهای الگوریتم GA

بطور خلاصه الگوریتم ژنتیک از عملگرهای زیر تشکیل شده است:

Encoding: این مرحله شاید مشکلترین مرحله حل مسأله به روش الگوریتم ژنتیک باشد. الگوریتم ژنتیک بجای اینکه بر روی پارامترها یا متغیرهای مسأله کارکند، با شکل کد شده آنها سروکار دارد. یکی از روشهای کدکردن کدکردن دودویی می‌باشد که در آن هدف تبدیل جواب مسأله به رشته‌ای از اعداد باینری است.

Evaluation: تابع برازندگی را از اعمال تبدیل مناسب بر روی تابع هدف یعنی تابعی که قرار است بهینه شود به دست می‌آید. این تابع هر رشته را با یک مقدار عددی ارزیابی می‌کند که کیفیت آن را مشخص می‌نماید. هر چه کیفیت رشته جواب بالاتر باشد مقدار برازندگی جواب بیشتر است و احتمال مشارکت برای تولید نسل بعدی نیز افزایش خواهد یافت.

Crossover: مهمترین عملگر در الگوریتم ژنتیک، عملگر ترکیب است. ترکیب فرآیندی است که در آن نسل قدیمی کروموزوم‌ها با یکدیگر مخلوط و ترکیب می‌شوند تا نسل تازه‌ای از کروموزوم‌ها بوجود بیاید. جفت‌هایی که در قسمت انتخاب، به عنوان والد در نظر گرفته شدند، در این قسمت ژن‌هایشان را با هم مبادله می‌کنند و اعضایی جدید به وجود می‌آورند. ترکیب در الگوریتم ژنتیک باعث از بین رفتن پراکندگی یا تنوع ژنتیکی جمعیت می‌شود. زیرا اجازه می‌دهد ژنهای خوب یکدیگر را بیابد[۱۸].

Mutation : جهش، جهش نیز عملگر دیگری است که جوابهای ممکن دیگری را تولید می‌کند. در الگوریتم ژنتیک بعد از اینکه یک عضو در جمعیت جدید بوجود آمد، هر ژن آن با احتمال جهش، جهش می‌یابد. در جهش ممکن است ژنی از مجموعه ژنهای جمعیت حذف شود تا ژنی که تا حالا در جمعیت وجود نداشته است به آن اضافه شود. جهش یک ژن به معنای تغییر آن ژن است و وابسته به نوع کد گذاری، روشهای متفاوت جهش استفاده می‌شود[۱۹].

Decoding : عکس عمل Encoding می‌باشد. در این مرحله بعد از اینکه الگوریتم بهترین جواب را برای مسأله ارائه داد لازم است عکس‌عمل رمزگزاری بر روی جوابها صورت گیرد تا بتوانیم نسخه جواب را به وضوح در دست داشته باشیم.

۳-۲-۴- پارامترهای الگوریتم GA

۱. Fitness Function : تابعی برای ارزیابی یک فرضیه که مقداری عددی به هر فرضیه نسبت می‌دهد.
۲. Fitness_threshold : مقدار آستانه که شرط پایان را معین می‌کند.
۳. P : تعداد فرضیه‌هائی که باید در جمعیت در نظر گرفته شوند.
۴. R : درصدی از جمعیت که درهر مرحله توسط الگوریتم Crossover جایگزین می‌شوند.
۵. M : نرخ Mutation .

۳-۲-۵- فضای فرضیه در الگوریتم GA

روش متداول پیاده‌سازی الگوریتم ژنتیک بدین ترتیب است که، مجموعه‌ای از فرضیه‌هاکه Population نامیده می‌شود تولید و بطور متناوب با فرضیه‌های جدیدی جایگزین می‌گردد. در هر بار تکرار تمامی فرضیه‌ها با استفاده از یک تابع ارزیابی یا برازندگی مورد ارزیابی قرار داده می‌شود آنگاه تعدادی از بهترین فرضیه‌ها با استفاده ازیک تابع احتمال انتخاب شده و جمعیت جدید راتشکیل می-

دهند. تعدادی از این فرضیه‌های انتخاب شده به همان صورت مورد استفاده قرار داده شده و مابقی با استفاده از اپراتورهای ژنتیکی نظیر Crossover و Mutation برای تولید فرزندان بکار می‌روند.

۳-۲-۶- انتخاب در الگوریتم GA

در مرحله انتخاب، یک جفت از کروموزوم‌ها برگزیده می‌شوند تا با هم ترکیب شوند. عملگر انتخاب رابط بین دو نسل است و بعضی از اعضای نسل کنونی را به نسل آینده منتقل می‌کنند. بعد از انتخاب، عملگرهای ژنتیک روی دو عضو برگزیده اعمال می‌شوند. معیار در انتخاب اعضا ارزش تطابق آنها می‌باشد اما روند انتخاب حالتی تصادفی دارد. شاید انتخاب مستقیم و ترتیبی به این شکل که بهترین اعضا دو به دو انتخاب شوند در نگاه اول روش مناسبی به نظر برسد اما باید به نکته ای توجه داشت. در الگوریتم ژنتیک ما با ژنها روبرو هستیم. یک عضو با تطابق پایین اگر چه در نسل خودش عضو مناسبی نمی‌باشد اما ممکن است شامل ژنهای خوب بوده و اگر شانس انتخاب شدنش برابر صفر باشد، این ژنهای خوب نمی‌توانند به نسلهای بعد منتقل شوند. پس روش انتخاب باید به گونه‌ای باشد که به این عضو نیز شانس انتخاب شدن بدهد. راه حل مناسب، طراحی روش انتخاب به گونه‌ای است که احتمال انتخاب شدن اعضای با تطابق بالاتر بیشتر باشد. انتخاب باید به گونه‌ای صورت گیرد که تا جایی که ممکن است هر نسل جدید نسبت به نسل قبلی‌اش تطابق میانگین بهتری داشته باشد[۱۷]. روشهای متداول انتخاب عبارتند از:

انتخاب چرخ رولت، انتخاب ترتیبی، انتخاب بولتزمن، انتخاب حالت پایدار، نخبه سالاری، انتخاب رقابتی، انتخاب مسابقه، انتخاب مسابقه تصادفی و انتخاب جایگزینی نسل اصلاح شده. که هر کدام به نوبه خود از ویژگی‌های خاصی برخوردارند و بسته به ویژگی‌های موجود روش انتخاب بهتر را انتخاب می-نماییم.

۳-۲-۷- مراحل الگوریتم GA

- Initialize : جمعیت را با تعداد p فرضیه بطور تصادفی مقداردهی اولیه می‌نماییم.
- Evaluate : برای هر فرضیه h در p مقدار تابع Fitness(h) را محاسبه می‌نماییم.
- تازمانیکه $[\max_h \text{Fitness}(h)] < \text{Fitness\_threshold}$ یک جمعیت جدید ایجاد کنید.
- فرضیه‌ای که دارای بیشترین مقدار Fitness است را برگردانید.

۳-۲-۸- نحوه ایجاد جمعیت جدید در الگوریتم GA

۱. Select : تعداد (1-r)*P فرضیه از میان p انتخاب و به P_s اضافه کنید. احتمال انتخاب یک فرضیه h_i از میان p با استفاده از رابطه (۳-۱) بدست می‌آید.

$$P(h_i) = \frac{Fitness(h_i)}{\sum_j Fitness(h_j)} \qquad (۱-۳)$$

مقدار j برابر کل فرضیه‌ها و مقدار i فرضیه‌ای است که احتمال آن قرار است محاسبه شود. هر چه تناسب فرضیه‌ای بیشتر باشد احتمال انتخاب آن بیشتر است. این احتمال همچنین با مقدار تناسب فرضیه‌های دیگر نسبت عکس دارد .

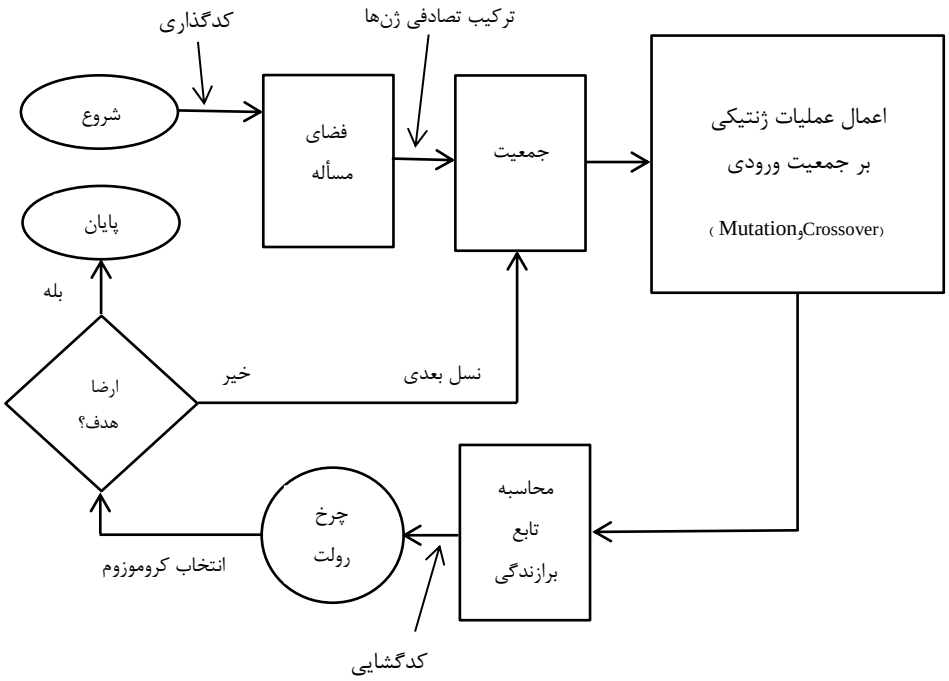
۲. Crossover : با استفاده از احتمال به دست آمده توسط رابطه فوق، تعداد (R*P)/2 زوج فرضیه از میان p انتخاب و با استفاده از اپراتور Crossover دو فرزند ازآنان ایجاد کنید و فرزندان را به P_s اضافه نمایید.

۳. Mutation: تعداد m درصد از اعضا P_s را با احتمال یکنواخت انتخاب و یک بیت از هر یک آنها را بصورت تصادفی معکوس کنید.

۴. Update : $P \leftarrow P_s$

۵. برای هر فرضیه h در p مقدار تابع Fitness را محاسبه نمایید.

در شکل ۳-۱ فلوچارت عملکرد الگوریتم ژنتیک نشان داده شده است.



شکل ۳-۱: فلوچارت عملکرد الگوریتم ژنتیک

۳-۲-۹- نقاط قوت و مزایای الگوریتم GA

اولین و مهمترین نقطه قوت این الگوریتم‌ها این است که الگوریتم‌های ژنتیک ذاتا موازی‌اند. اکثر الگوریتم‌های دیگر موازی نیستند و فقط می‌توانند فضای مسأله مورد نظر را در یک جهت و در یک لحظه جستجو کنند و اگر راه حل پیدا شده یک جواب بهینه محلی باشد و یا زیرمجموعه‌ای از جواب اصلی باشد باید تمام کارهایی که تا به حال انجام شده را کنار گذاشت و دوباره از اول شروع نمود. از آنجایی که ژنتیک چندین نقطه شروع دارد، در یک لحظه می‌تواند فضای مسأله را از چند جهت مختلف جستجو نماید. اگر یکی از راه‌ها به نتیجه نرسید سایر راه‌ها ادامه می‌یابد و منابع بیشتری در اختیارشان قرار می-گیرد. به دلیل موازی بودن الگوریتم ژنتیک و این که چندین رشته در یک لحظه مورد ارزیابی قرار می-گیرند الگوریتم فوق برای مسائلی که فضای راه حل بزرگی دارند بسیار مفید است. در ادامه به چند مورد از مزایای الگوریتم ژنتیک اشاره شده است:

۱. با متغیر های پیوسته و هم گسسته می‌تواند عمل بهینه‌سازی را انجام دهد.

۲. نیازی به محاسبه مشتق توابع ندارد.

۳. بطور هم‌زمان می‌تواند تمامی ناحیه جستجو شونده را جستجو کند.

۴. قادر به بهینه‌سازی مسائل با تعداد متغیرهای زیاد می‌باشد.

۵. قابل اجرا از طریق کامپیوترهای موازی است.

۶. قادر است تا چند جواب بهینه را هم زمان بدست آورد نه فقط یک جواب.

۷. توابع هزینه‌ای که بسیار پیچیده باشند نیز از این طریق قابل بهینه‌سازی می‌باشند و احتمال اینکه الگوریتم در ماکزیمم‌محلی قرار گیرد کم است.

۸. با کدبندی قادر است سرعت الگوریتم را تا جایی افزایش دهد.

۳-۲-۱۰- محدودیت‌های الگوریتم GA

یک مشکل چگونگی نوشتن عملگر Fitness است که منجربه بهترین راه حل برای مسأله شود. اگر این کارکرد برازش به خوبی و قوی انتخاب نشود ممکن است باعث شود که راه‌حلی برای مسأله پیدا نکنیم یا مسأله‌ای دیگر را به اشتباه حل کنیم. به علاوه برای انتخاب تابع مناسب برای Fitness، پارامترهای دیگری مثل اندازه جمعیت، نرخ جهش و Crossover، قدرت و نوع انتخاب هم باید مورد توجه قرارگیرد. مشکل دیگر این الگوریتم، که آن را نارس می‌نامند، این است که اگر یک ژنوم که فاصله‌اش با سایر ژنوم-های نسل‌اش زیاد باشد(خیلی بهتر از بقیه باشد) و خیلی زود دیده شود(ایجاد شود) ممکن است محدودیت ایجاد کند و راه حل را به سوی جواب بهینه محلی سوق دهد. این اتفاق معمولا در جمعیت-های کم اتفاق می‌افتد.

۳-۳- بهینه‌سازی توده ذرات

الگوریتم بهینه‌سازی توده ذرات (PSO) از آن جمله الگوریتم‌های تکاملی است که از مدلهای واقعی در طبیعت الهام گرفته شده است. PSO از رفتار جمعی پرندگان در یافتن غذا بهره می‌گیرد بدین ترتیب که جمعیتی متشکل از یک سری ذرات تشکیل می‌دهد که هر ذره معرف یک پرنده در فضای جستجو است و PSO با به روز کردن[۱] موقعیت[۲] ذرات با توجه به میزان شایستگی، جمعیت را به سمت جواب بهینه[۳] هدایت می‌کند.

PSO در دو ترکیب اصلی اصلوب شناسی[۴] ریشه دارد و شاید آشکارترین آنها با زندگی مصنوعی در حالت کلی و با گروه پرندگان یا ماهی‌ها و تئوری جمعی بطور خاص گره خورده است. می‌توان پیشنهاد کرد که قوانین منطقی خاصی بر نحوه رفتار موجودات اجتماعی حکم فرمایی می‌کند. پرندگان تنها با تنظیم حرکت فیزیکی خود با اجتناب از تصادم (برخورد) به دنبال غذا می‌گردند و بطور تئوری حداقل هر پرنده به عنوان یکی از اعضاء گروه از تجربه قبلی خود و یافته‌های سایر اعضاء برای یافتن غذا بهره می‌برد و این مشارکت یک مزیت قطعی بر جستجوی رقابتی برای یافتن غذای توزیع شده است. پایه اصلی نظریه PSO همین تسهیم اطلاعات بین اعضاء یک گروه است[۲۰].

۳-۳-۱- PSO به عنوان یک الگوریتم جستجو

امروزه با بزرگ شدن مسائل و اهمیت یافتن سرعت به پاسخ و عدم پاسخگویی روشهای کلاسیک، از الگوریتم های جستجوی رندوم بجای جستجوی همه جانبه[۵] فضای مسأله، استقبال بیشتری می‌شود. در این بین در سالهای اخیر استفاده از الگوریتم جستجوی هیوریستیک (شهودی) همچون الگوریتم ژنتیک، الگوریتم کلونی مورچگان و ... رشد چشمگیری داشته است. الگوریتم جستجوی pso یکی از جدیدترین

[1] Update
[2] Position
[3] Optimum Solution
[4] Methodology
[5] Exhaustive search

روشهای جستجو است. در این روش با تنظیم مسیر حرکت، یک جمعیت از ذرات[1] در فضای مسأله بر پایه اطلاعات مربوط به بهترین کارایی قبلی مربوط به هر ذره و بهترین کارایی قبلی مربوط به همسایگان هر ذره عمل جستجو را در فضای مسأله انجام می‌دهد.

PSO یک تکنیک بهینه سازی مبتنی بر قوانین احتمال است که توسط دکتر راسل ابرهارت[2] و دکتر جیمزکندی[3] در سال ۱۹۹۵ ارائه شد و از رفتار اجتماعی پرندگان یا ماهی‌ها در پیدا کردن غذا الهام گرفته است[۲۱,۲۰]. فرض بر این است که یک گروه از پرندگان بصورت تصادفی در یک منطقه به دنبال غذا می‌گردند در حالی که تنها در یک قسمت از ناحیه‌ی جستجو، غذا وجود دارد. پرندگان از مکان غذا اطلاعی ندارند و تنها میزان فاصله‌ی خود تا آن محل را می‌دانند. استراتژی بکار رفته این است که پرندگان به دنبال پرنده‌ای حرکت می‌کنند که نزدیکترین فاصله را تا غذا دارد. در PSO هر جواب مسأله، یک پرنده در فضای جستجو است که ذره نام دارد. هر ذره دارای یک مقدار شایستگی[4] است که توسط تابع شایستگی مسأله بدست می‌آید. پرنده‌ای که به غذا نزدیکتر است، مقدار شایستگی بیشتری دارد. این الگوریتم ماهیت پیوسته دارد و در کارهای متعدد کارایی خود را ثابت کرده است. امروزه این الگوریتم در بسیاری از کاربردها استفاده می‌شود. الگوریتم PSO ذاتاً یک الگوریتم پیوسته است. برای حل مسائل گسسته نسخه باینری آن نیز ارائه شده است. یکی ازمشخصه‌های اصلی این روش، تفکر و هوش است. تفاوت عمده‌ای که این روش با الگوریتم‌های دیگرچون الگوریتم ژنتیک دارد، این است که اعضای جامعه از وضعیت سایر اعضا و یا بهترین عضو جامعه باخبر هستند و نتیجه به دست آمده توسط آن‌ها را در تصمیم‌گیری‌های خود لحاظ می‌کنند. همچنین اعضا، بهترین نتیجه خود، درطی اجرای الگوریتم را به یادداشته و همواره سعی می‌کنند تا آنرا نیز درتصمیمات خود دخالت دهند. به همین دلیل، در صورتی که اشتباهی رخ دهد و تصمیم بدی بگیرند، به زودی آنرا جبران می‌کنند. به این ترتیب، اعضای جامعه می-توانند به راحتی محدوده اطراف خود را جستجو کنند، بدون آنکه نگران خراب‌تر شدن نتیجه باشند. اگر

[1] Particles
[2] Russel Eberhart
[3] James Kennedy
[4] Fittness Value

تصمیمات جدید خوب باشند، پذیرفته می‌شوند و اگر بد باشند، الگوریتم قابلیت جبران اشتباهات به وجود آمده را دارد.

۳-۳-۲- هوش دسته جمعی[1]

هوش‌جمعی (SI) خاصیتی است سیستماتیک که در این سیستم، عامل‌ها بطور محلی با هم همکاری می‌نمایند و رفتار جمعی تمام عامل‌ها باعث یک همگرایی در نقطه‌ای نزدیک به جواب بهینه سراسری می‌شود نقطه قوت این الگوریتم عدم نیاز به یک کنترل سراسری می‌باشد. هر ذره (عامل) در این الگوریتم خود مختاری نسبی دارد که می‌تواند در سراسر فضای جواب حرکت کند و می‌بایست با سایر ذرات (عامل ها) همکاری داشته باشد.

الگوریتم PSO که به نام‌های الگوریتم انبوه ذرات، الگوریتم ازدحام ذرات و الگوریتم پرندگان نیز مشهور است، یکی از الگوریتم‌های بسیار پرکاربرد در زمینه بهینه‌سازی استاتیک و دینامیک است. این الگوریتم سرعت همگرایی مناسبی دارد و در اغلب کاربردها، به عنوان گزینه اول مورد استفاده قرار می-گیرد. با وجود قدمت ۱۴ ساله این الگوریتم، که در مقایسه با عمر الگوریتم ژنتیک بسیار کمتر است، گرایش به سمت این الگوریتم قدرتمند، هر روز بیشتر و بیشتر می‌شود. Swarm را می‌توان به صورت مجموعه‌ای سازمان یافته از موجوداتی تعریف کرد که با یکدیگر همکاری می‌کنند. در کاربردهای محاسباتی هوش جمعی از موجوداتی مانند مورچه‌ها، زنبورها، موریانه‌ها، دسته‌های ماهیان و دسته پرندگان الگو برداری می‌شود. در این نوع اجتماعات هر یک از موجودات ساختار نسبتا ساده‌ای دارند، ولی رفتار جمعی آنها بی‌نهایت پیچیده است. برای مثال در کلونی مورچه‌ها که در ادامه این فصل آورده شده، هر یک از مورچه‌ها یک کار ساده مخصوص را انجام می‌دهد، ولی عمل و رفتار مورچه‌ها بطور جمعی، ساختن بهینه لایه محافظت از ملکه و نوزادان، تمیز کردن لانه، یافتن بهترین منابع غذایی و بهینه‌سازی استراتژی حمله را تضمین می‌کند. عبارت Swarm در زبان انگلیسی به اجتماع دسته انبوهی از جانوران و حشرات اشاره می‌کند. رفتار کلی یک جمعیت، به صورت غیرخطی از آمیزش رفتارهای تک‌تک اجتماع به

[1] Swarm Intelligence

دست می‌آید، یا به عبارتی یک رابطه بسیار پیچیده بین رفتار جمعی و رفتار فردی یک اجتماع وجود دارد. رفتار جمعی فقط وابسته به رفتار فردی افراد اجتماع نیست، بلکه به چگونگی تعامل میان افراد نیز وابسته است. تعامل بین افراد، تجربه افراد درباره محیط را افزایش می‌دهد و موجب پیشرفت اجتماع می-شود.

۳-۳-۳- الگوریتم PSO

PSO با یک گروه از جوابهای تصادفی (ذره ها) شروع به کار می‌کند، سپس برای یافتن جواب بهینه در فضای مسأله با به روز کردن نسلها به جستجو می‌پردازد. هر ذره بصورت چند بعدی[1] با دو مقدار V_{id} وX_{id} که به ترتیب معرف وضعیت مکانی و سرعت مربوط به بعد d ام از i امین ذره هستند، تعریف می-شود. در هر مرحله از حرکت جمعیت، هر ذره با دو مقدار بهترین[2] به روز می‌شود[۲۰]. اولین مقدار، بهترین جواب از لحاظ شایستگی است که تاکنون برای هر ذره بطور جداگانه بدست آمده است(مقدار شایستگی باید ذخیره شود) این مقدار P_best نامیده می‌شود. مقدار بهترین دیگری که توسط PSO بدست می‌آید، بهترین مقداری است که تاکنون توسط تمام ذره‌ها در میان جمعیت بدست آمده است این مقدار بهترین کلی[3] است و G_best نام دارد. اگر ذره در یک همسایگی مکانی خودش به عنوان جمعیت شرکت کند این مقدار تنها در همان همسایگی محاسبه می‌شود و بهترین محلی[4] است که L_best نام دارد. پس دو نسخه از PSO وجود دارد:

۱. همسایگی فراگیر: که هر ذره به سمت بهترین موقعیت پیشین خود و بهترین ذره در کل گروه حرکت می‌کند.
۲. همسایگی محلی: که هر ذره به سمت بهترین موقعیت پیشین خود و بهترین ذرات در محدوده همسایگی محلی حرکت می‌کنند.

[1] Multi dimensional
[2] Best Value
[3] Global best
[4] Local best

بعد از یافتن دو مقدار P_best و G_best یا L_best هر ذره سرعت و موقعیت جدید خود را طبق روابط (۳-۲) و (۳-۳) به روزرسانی می‌کند.

$$V_{id}(t+1) = w \times V_{id}(t) + C_1 \times Rand \times (P_{best} - X_{id}) + C_2 \times Rand \times (G_{best} - X_{id}) \quad (۳-۲)$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \quad (۳-۳)$$

که در روابط بالا w وزن اینرسی[1]، c_1 وc_2 عوامل یادگیری[2] که به ضرایب شتاب نیز مشهورند و عموما در بازه (0,2) انتخاب می‌شوند و مجموع آنها نبایستی از ۴ تجاوز نماید تا مانع همگرایی نشوند و مقدار Rand یک عدد تصادفی در بازه (0,1) است. برای جلوگیری از واگرایی الگوریتم، مقدار نهایی سرعت هر ذره محدود می‌شود. سرعت ذره‌ها در هر بعد، محدود به ماکزیمم سرعت V_{max} است که چگونگی حرکت ذره‌ها در فضای جستجو را مشخص می‌کند و اگر V_{max} خیلی کوچک باشد، ممکن است ذره‌ها بخوبی در فضاهای مناسب محلی کاوش نکنند و در بهینه محلی بدام بیافتند. از سویی دیگر اگر این مقدار بیش از حد زیاد باشد ممکن است ذرات در فضاهای دور از جواب حرکت کنند[۲۲]. رابطه (۳-۴) این نگاشت را بیان می‌نماید.

$$V_{id} \in [-V_{max}, +V_{max}] \quad (۳-۴)$$

رابطه (۳-۲) شامل جمع سه عبارت است که عبارت اول نسبتی از سرعت جاری ذره است و به همراه عبارت دوم که متناسب با تفاضل مکان پرنده با بهترین موقعیت قبلی آن و عبارت سوم که تفاضل مکان آن با بهترین جواب در میان کل جمعیت است و سبب هدایت سرعت جدید ذره به سمت جواب بهینه می‌شود.

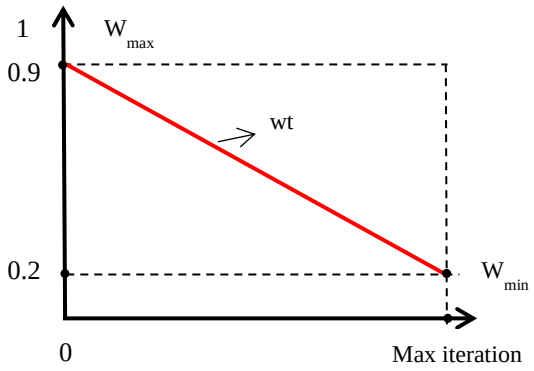
W و c_1 و c_2 از پارامترهای PSO هستند و همگرایی وابسته به مقدار این پارامترهاست. معمولا c_1 را برابر c_2 و عددی بین ۱,۵ تا ۲ قرار می‌دهند. همگرایی شدیداً به مقدار W وابسته است و بهتر است

[1] Inertia weight
[2] Learning factors

بصورت دینامیک تعریف شود(در بازه ۰,۸ تا ۰,۲)، بدین ترتیب که بصورت خطی در طی روند تکامل جمعیت، کاهش یابد در ابتدا باید بزرگ باشد تا امکان یافتن جوابهای خوب در همان مراحل اولیه فراهم شود و در مراحل پایانی کوچک بودن w همگرایی بهتری را سبب می‌شود[۲۳]. در این پایان‌نامه این مقدار را بر اساس تعداد تکرار[1] با معادله (۳-۵) تعیین می‌کنیم که مقدار $W_{initial}$ برابر با ۰,۹ می‌باشد.

$$W(t) = W_{initial} - \frac{0.5 \times iteration}{\max iteration}$$
$$w(t) = \left(w_{initial} - w_{final}\right) \times (max_{iteration} - iteration) + 0.2 \qquad (۵-۳)$$
$$w_{initial} = 0.9 \ , \ w_{final} = 0.2$$
$$w(t+1) = w(t) + dw \quad where \quad dw = \frac{w_{min} - w_{max}}{max_{iteration}}$$

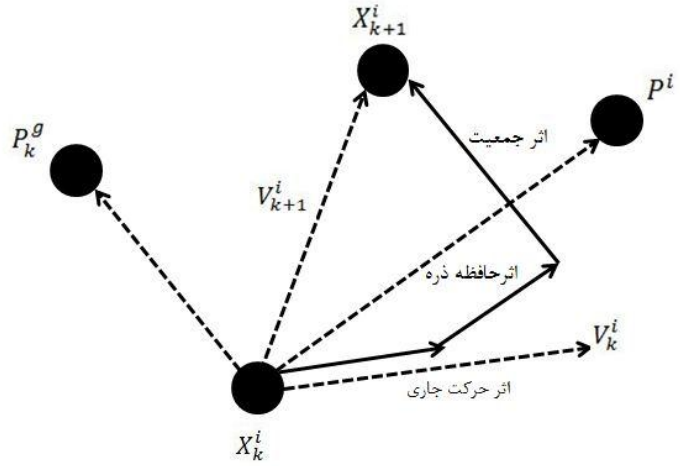
مقدار W همچنین در کاهش زمان نیز موثر بوده و طبق تعریف بین W_{max} و W_{min} قرار می‌گیرد که هر چقدر مقدار W زیاد باشد یعنی به W_{max} نزدیک، مقدار تکرار کاهش یافته و بلعکس طرحی که برای این منظور توسط ابرهارت و همکارش معرفی شده در **شکل ۳-۲** نشان داده شده است.



شکل ۳-۲: تغییرات ضریب اینرسی بر اساس تعداد تکرار

PSO ای که با استفاده از این روابط جمعیت خود را به روز می‌کنند، PSO پایه یا استاندارد نام دارد. در **شکل ۳-۳** نحوه تأثیر روابط به روزرسانی سرعت و موقعیت بر موقعیت هر ذره و فلوچارت الگوریتم بهینه‌سازی توده ذرات PSO در **شکل ۳-۴** به تصویر کشیده شده است.

[1] Iteration



شکل ۳-۳: به روز شدن سرعت و موقعیت ذره

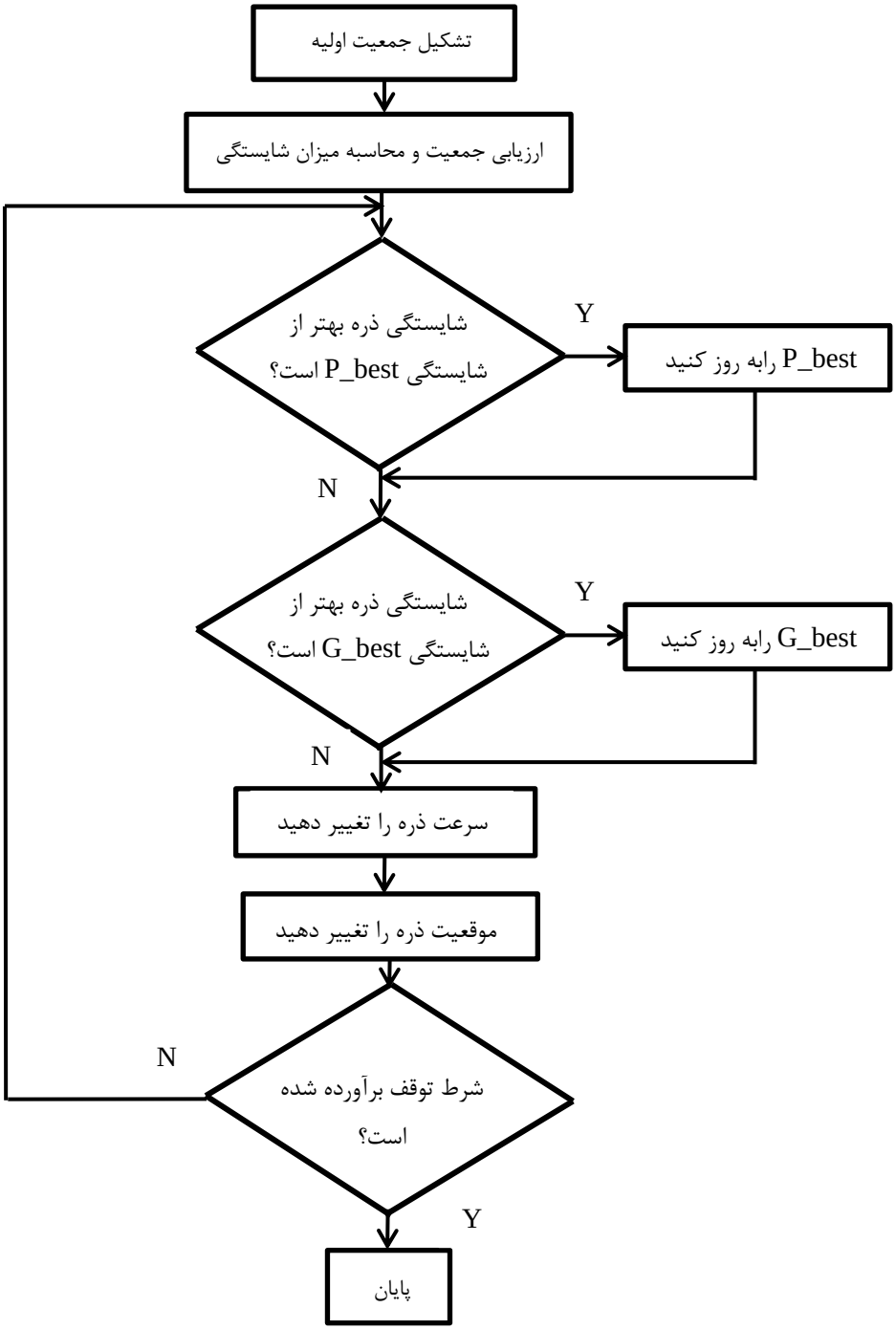
شرط توقف[1] را می‌توان به چند صورت تعریف نمود، مثلا بر روی حداکثر تعداد تکرار، یا اینکه ماکزیمم تغییر در بهترین شایستگی طبق رابطه (۳-۶) برای تعداد معینی از حرکت جمعیت (S)، کمتر از تلرانس تعریف شده باشد[۲۴].

$$\left|f(p_k^g) - f(p_{k-q}^g)\right| \leq \varepsilon \quad where \quad \varepsilon = 10^{-5} \;\&\; q = 1,2,\dots,s \qquad (۳-۶)$$

۳-۳-۴- الگوریتم باینری PSO

برای حل مسائل گسسته روش باینری این الگوریتم توسط آقای کندی در سال ۱۹۹۷ ارائه شد و به باینری PSO شهرت یافت[۲۵]. استفاده از الگوریتم باینری PSO همچنین زمان محاسبات را کاهش می-دهد. تفاوت عمده این روش با الگوریتم اصلی در روابط بروزرسانی موقعیت و سرعت می‌باشد که ابتدا باید مقدار بردار سرعت توسط تابع سیگموید به بازه [۰،۱] منتقل شود سپس موقعیت جدید ذره محاسبه شود.

[1] Stop criterion



شکل ۳-۴: فلوچارت الگوریتم PSO

موقعیت هر ذره در رشته باینری با دو مقدار صفر و یک در هر بعد مشخص می‌شود و وضعیت انتخاب ویژگی را نمایش می‌دهد. هرذره بر اساس معادله‌های (۳-۷) و (۳-۸) به روزرسانی می‌شود که سرعت بر اساس معادله (۳-۲) بدست می‌آید.

$$S(V_i^{t+1}) = \frac{1}{1 + e^{-V_i^{t+1}}} \quad (۷-۳)$$

$$X_i^{t+1} = \begin{cases} 1 & if \;\; Rand < S(V_i^{t+1}) \\ 0 & else \end{cases} \quad (۸-۳)$$

در اینجا Rand یک عدد تصادفی در بازه [0,1] می‌باشد.

۳-۳-۵- مزیت های PSO در قیاس با سایر الگوریتم های جستجو

اکثر تکنیکهای تکاملی روند زیر را دنبال می‌کنند[۱۵]:

۱. شروع بکار با یک جمعیت تصادفی اولیه
۲. محاسبه مقدار شایستگی برای هر جزء
۳. تولید مجدد جمعیت براساس مقادیر شایستگی
۴. اگر نیاز برآورده نشده باشد همین روند از مرحله دوم تکرار می‌شود.

همانطور که مشاهده می‌شود PSO نقاط مشترک زیادی با دیگر الگوریتم‌ها در طی این فرآیند مشابه دارد، اما وجود یکسری تفاوتها آن را از سایر روشها متمایز کرده است. مهمترین مسأله سادگی PSO است که از چند نظر قابل بررسی است. اول اینکه پیاده‌سازی آن ساده است و برنامه آن از چند خط فراتر نمی-رود، درثانی این سادگی موجب بالا رفتن سرعت در محاسبات و رسیدن سریع به جواب دلخواه با حجم کم حافظه مورد نیاز، می‌شود[۲۰]. برای درک بهتر در ادامه تفاوت‌های PSO با GA بیان می‌شود. مکانیزم تسهیم اطلاعات در PSO با GA متفاوت است بطوریکه در GA تمام کروموزوم‌ها[1] اطلاعات را با هم دیگر به مشارکت می‌گذارند، بنابراین کل جمعیت همانند یک گروه به سمت منطقه بهینه حرکت می‌کند اما در PSO تنها P_best وG_best یا L_best اطلاعات را به سایرین می‌دهند و در سیر تکامل، ذره ها تنها تحت تأثیر بهترین جوابها حرکت می‌کنند و بنابراین تمام ذره‌ها به سرعت به بهترین جواب در اغلب

[1] Chromosoms

موارد همگرا می‌شود[۱۵]. نقطه ضعف GA محاسبات پرهزینه آن است، ولی PSO همان کارآمدی را در یافتن جواب بهینه دارد و با توجه به عملگرهای کم آن از لحاظ محاسباتی نیز کم هزینه است[۲۴].

۳-۳-۶- کاربردهای PSO

PSO در حوزه وسیعی از حل مسائل من جمله در بهینه‌سازی توابع مشکل و چند متغیره، با سرعت و کارایی بالا قابل استفاده است[۲۶]. در ادامه جهت آشنایی فهرست وار به برخی از کاربردهای PSO اشاره می‌شود:

۱. شبکه عصبی مصنوعی
۲. کنترل سیستم های فازی[1]
۳. بازشناسی الگو
۴. انتخاب ویژگی
۵. خوشه بندی اطلاعات[2]
۶. مسیریابی و ...

۳-۳-۷- رهیافت های جدید برای بهبود همگرایی PSO

جمعیت PSO در مراحل ابتدائی جستجو به سرعت همگرا می‌شود و در نتیجه احتمال افتادن در بهینه های محلی[3] نیز بالاست[۲۷]. راه حلهای مختلفی برای فرار از بهینه‌های محلی و ارتقاء کارایی الگوریتم ارائه شده است[۲۸]. که در ادامه به مهمترین آنها اشاره می‌شود. اصلی‌ترین راهکار برای جلوگیری از همگرایی زودرس[4] تعریف همسایگی برای ذره‌هاست. همان‌طور که در بخش قبلی اشاره شد دو مدل Global و Local برای الگوریتم ارائه شده است که با استفاده از مدل محلی امکان عبور از بهینه-های محلی بیشتر است. در رابطه (۳-۲) مقدار L_best جایگزین G_best می‌شود، یعنی در به روزرسانی

[1] Fuzzy System Control
[2] Data Clustering
[3] Local Optima
[4] Premature convergence

سرعت ذره‌ها بجای یافتن بهترین جواب در کل جمعیت، باید برای هر ذره در یک همسایگی خودش بهترین را پیدا کرد و مسأله مهم تعیین این همسایگی است. در ادامه دو روش تعیین همسایگی که بیشتر مورد استفاده قرار می‌گیرند، به دو صورت ثابت و متغیر آمده است. در بیشتر موارد i امین ذره در یک همسایگی ثابت شامل خودش، i+1 امین وi-1 امین ذره فرض می‌شود که بدیهی است در این صورت در بردار ذره‌ها اولین و آخرین ذره کنار یکدیگرند[۳۰،۲۹]. همسایگی را می‌توان به گونه دیگر نیز تعریف نمود که در مراحل ابتدایی تنها شامل خود ذره باشد و همین طور که تعداد نسلها یا مقدار تکرار افزایش می‌یابد، همسایگی نیز بسط یافته تا در نهایت تمام جمعیت را در بر بگیرد که البته این عمل زمان محاسبات را افزایش می‌دهد. در این مدل در ابتدا تعداد همسایه‌ها 2(k) است و بطور تدریجی تا اندازه جمعیت منهای یک (در حین افزایش تعداد تکرار الگوریتم) طبق رابطه (۳-۹) افزایش می‌یابد[۳۰].

$$k = 2 + round(\frac{(Population\_size - 3) \times iteration\_num}{\max\ iteration\_num}) \qquad (۳-۹)$$

۳-۴- الگوریتم بهینه‌سازی کلونی مورچگان ACO

در مسایل بهینه‌سازی با تعداد پارامتر زیاد، روش‌های کلاسیک کارایی چندانی ندارند. در این گونه موارد بهتر است بجای بررسی تمام فضای جواب، از روش‌های دیگری استفاده شود که به صورت هوشمند ،فضای گستره جستجو را کاهش می‌دهند. یکی دیگر از این روش‌ها، الگوریتم بهینه‌سازی کلونی مورچه‌ها (ACO) می‌باشد. ACO یک فرااکتشاف الهام گرفته شده از رفتار مورچگان طبیعی، در یافتن کوتاهترین مسیر برای رسیدن به منابع غذایی است که برای اولین بار در سال ۱۹۹۷ توسط Dorigo و Gambardella ارائه شد[۳۱]. الگوریتم کلونی مورچگان، شاخه‌ای از علم تازه توسعه‌یافته هوش مصنوعی، به نام هوش ازدحامی است که درباره هوش جمعی گروه‌های عامل ساده، مطالعه می‌کند[۳۲].

الگوریتم ACO از رفتار جامعه مورچگان الهام گرفته شده است. مورچه‌ها حشراتی اجتماعی هستند که در کلونی‌ها زندگی می‌کنند و رفتار آن‌ها بیشتر از این که در جهت بقای یک جزء از آن باشد، در

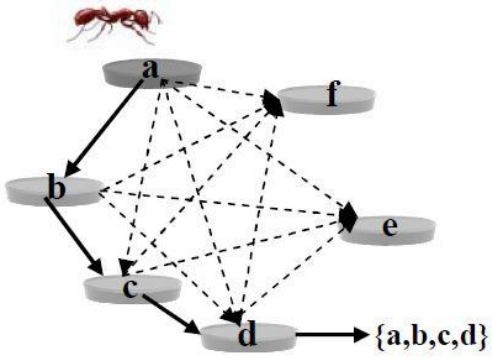
جهت بقای کلونی است. مورچه‌ها حس بینایی ندارند و استعداد خوبی در یافتن کوتاهترین مسیر بین منبع غذایی و آشیانه خود دارند که این امر به وسیله ترشح ماده شیمیایی به نام فرومون به هنگام حرکت انجام می‌گیرد. این الگوریتم برای اولین بار در حل مسأله فروشنده دوره‌گرد[1] (TSP) به کار گرفته شد[۳۳]. سپس بطور موفقیت‌آمیز به مسایل مشکل زیادی، نظیر مسأله تخصیص منشور قائم[2] (QAP) [۳۴]، مسیریابی در شبکه‌های ارتباطی، مسایل رنگ‌آمیزی گراف، زمان‌بندی‌ها و انتخاب ویژگی اعمال شد. اگر ویژگی‌ها به صورت گراف نشان داده شوند، مورچه‌ها می‌توانند بهترین ترکیبات ویژگی را به هنگام پیمایش گراف، کشف کنند. مورچه‌ها به صورت غیر مستقیم و از طریق ماده‌ای به نام فرومون با هم ارتباط بر قرار می‌کنند و اطلاعات مربوط به مسیرها را در اختیار یکدیگر قرار می‌دهند. مورچه‌ها با ترشح مقدار معینی فرومون، مسیر خود را مشخص و علامت‌گذاری می‌کنند که البته این ماده به زودی تبخیر می‌شود، این اثر در کوتاه مدت به عنوان رد مورچه بر سطح زمین باقی می‌ماند و مورچه‌های دیگر را به سمت خود جذب می‌کند. سایر مورچه‌ها با بررسی و مشاهده فرومون موجود در مسیر، راه یافتن غذا را پیدا می‌کنند. به این ترتیب یک مورچه می‌تواند با استفاده از یک تابع احتمالی مسیر خود را انتخاب کند. هر چه تعداد مورچه‌های عبور کرده از یک مسیر بیشتر باشد یا فرومون بیشتری داشته باشد، آن‌گاه احتمال آنکه آن مسیر توسط مورچه بعدی انتخاب شود، بیشتر است. در ابتدا مورچه‌ها پس از خارج شدن از لانه خود با احتمال مساوی به جهات مختلف، برای یافتن غذا حرکت می‌کنند. مورچه‌ها در مسیرهای ناهموار، با سرعت مساوی حرکت می‌کنند و در مسیر خود فرومون ترشح می‌کنند. بعد از این که یک مورچه در مسیر خود غذایی پیدا کرد، به سرعت از همان مسیر به سمت لانه مراجعت می‌کند و میزان ترشح فرومون را در مسیر افزایش می‌دهد. به این ترتیب سایر مورچه‌ها که مسیرهای دیگر را تجربه کرده‌اند، به تدریج به یک مسیر هدایت خواهند شد. نکته بسیار با اهمیت این است که هر چند احتمال مسیر پرفرومون توسط مورچه‌ها بیشتر است، ولی این کماکان احتمال است و قطعیت نیست. اگر فرض کنیم که بجای این احتمال، قطعیت وجود می‌داشت، آن‌گاه اساسا این روش ممکن نبود به جواب برسد.

[1] Travelling salesman problem
[2] Quadratic assignment problem

نکته دیگر مسأله تبخیر شدن فرومون برجای گذاشته شده است. تبخیر شدن فرومون و احتمال، به مورچه‌ها امکان پیدا کردن مسیر کوتاه‌تر جدید را می‌دهد. برای پیاده‌سازی کلونی مورچه‌ها، ازمورچه‌های مصنوعی در بهینه‌سازی استفاده می‌شود. این عناصر تفاوت‌های اساسی با مورچه‌های واقعی دارند که عبارتند از:

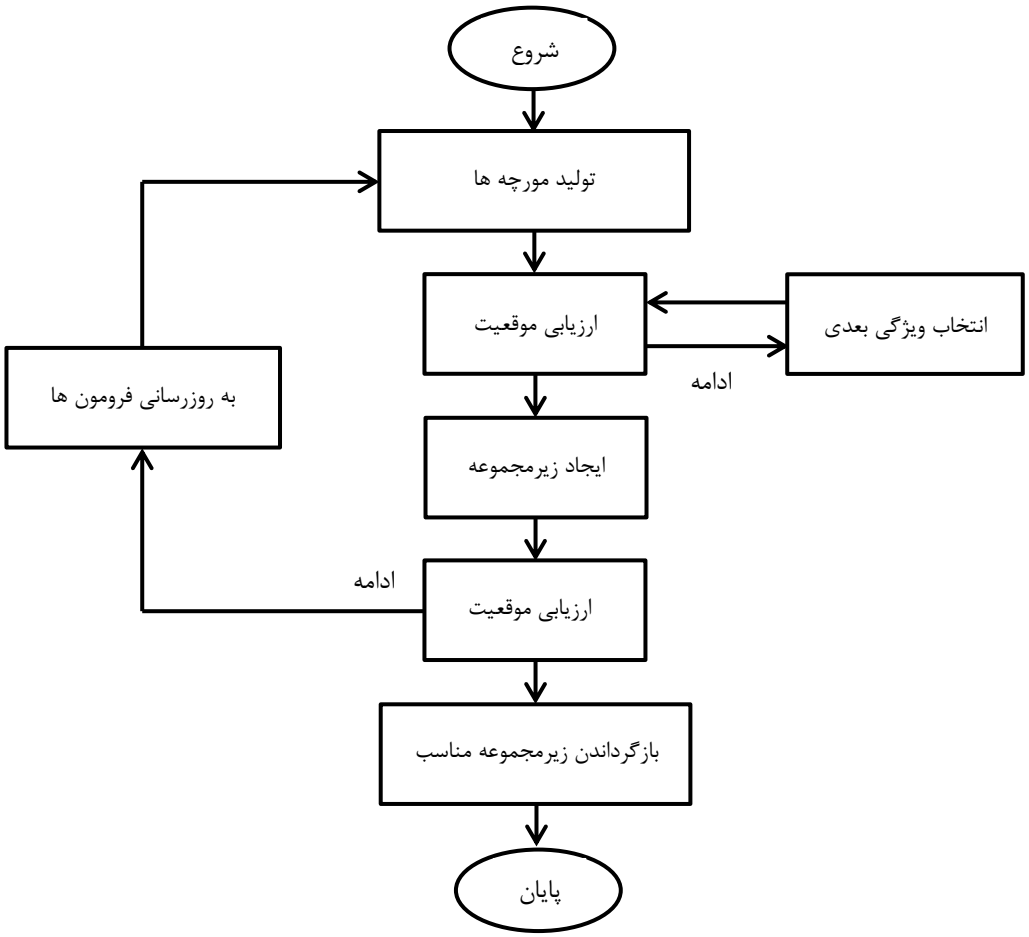
۱. **حافظه** : برای مورچه‌های مصنوعی می‌توان یک حافظه درنظرگرفت که مسیرهای حرکت را در خود نگه دارد.
۲. **موانع ساختگی** : تغییر دادن جزئیات مسأله برای بررسی الگوریتم در رسیدن به جواب‌های متنوع.
۳. **حیات در محیط گسسته** : مورچه‌های واقعی نمی‌توانند جدا از کلونی به حیات خود ادامه دهند.

چنانچه بتوان ساختارمسأله رابه صورت یک گراف نمایش داد، آنگاه می‌توان از الگوریتم‌های ACOبرای یافتن کوتاه‌ترین مسیر در گراف که همان پاسخ مسأله است، استفاده کرد. که در مسأله انتخاب ویژگی می‌توان از این روش بهره برد[۳۵]. این امردر **شکل ۳-۵** نشان داده شده است. هرمورچه به صورت تکاملی اقدام به تغییر یا ساخت بخشی ازپاسخ مسأله می‌نماید.



شکل ۳-۵: نحوه انتخاب ویژگی از روی گراف ویژگی

فرآیند انتخاب ویژگی در ACO را می‌شود در شکل ۳-۶ مشاهده کرد[۳۶]:



شکل ۳-۶: فلوچارت انتخاب ویژگی با الگوریتم کلونی مورچگان ACO

در این شیوه، کار با تولید k عامل که هر کدام به صورت تصادفی در گراف قرار داده شده‌اند، شروع می‌شود (هر عامل، پیمایش را از یک ویژگی مختلف بطور تصادفی آغاز می‌کند). تعداد این عامل‌ها می‌تواند برابر تعداد ویژگی‌ها باشد. عامل‌ها از موقعیت‌های آغازین، شاخه‌ای را طبق فرمول احتمال گذار که احتمال حرکت عامل k را از ویژگی i به j در زمان t معین می‌کند، انتخاب می‌کنند. رابطه تابع احتمال (۳-۱۰) بدین قرار است:

$$P_{ij}^{k}(t) = \begin{cases} \frac{[\tau_{ij}(t)^{\alpha}].[\eta_{ij}]^{\beta}}{\sum_{l \in J_i^k}[\tau_{il}(t)^{\alpha}].[\eta_{il}]^{\beta}} & if\ \ j \in J_i^k \\ 0 & otherwise \end{cases} \qquad (۳-۱۰)$$

که j_i^k مجموعه ویژگی‌هایی است که توسط عامل k ملاقات نشده است.

η_{ij}یک تابع اکشافی مطلوبیت انتخاب ویژگی j در هنگام ویژگی i است.

τ_{ij} مقدار فرومون موجود در شاخه (i,j) می‌باشد.

مقادیر α و β دو پارامتری هستند که اهمیت وابستگی و ارتباط اثر فرومون و اطلاعات اکتشافی را تعیین می‌کنند. این مقادیر بطور آزمایشی تعیین می‌شوند و در بازه [0,1] قرار دارند. اگر مقدار α خیلی بزرگتر از β باشد، مورچه‌ها بیشتر بر اساس اثر فرومون تصمیم می‌گیرند و اگر مقدار β خیلی بزرگتر از αباشد، مورچه‌ها بطور حریصانه شاخه‌ای را انتخاب می‌کنند که اطلاعات اکتشافی بیشتری دارد[۳۷]. بعد از هر انتخاب مورچه، فرومون موجود بر روی شاخه باید طبق رابطه (۳-۱۱) به روز شود:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho.\Delta\tau_{ij}(t)$$
$$\Delta\tau_{ij}(t) = \sum_{k=1}^{n}\left(\frac{\gamma'(s^k)}{|s^k|}\right) \qquad (۱۱-۳)$$

که درآن $(0\leq \rho \leq 1)$ ثابت ماندگاری به منظور شبیه‌سازی تبخیر فرومون است. S^k زیر مجموعه ویژگی یافته شده توسط مورچه k است. γ' مقدار بهترین اندازه‌گیری شده برای زیرمجموعه ویژگی که توسط مورچه‌ها بدست آمده است. اگر مجموعه جواب بهینه پیدا شد یا تعداد تکرارها به حد نهایی رسید، آنگاه الگوریتم متوقف خواهد شد و کوچکترین مجموعه جواب کاهش‌یافته را دریافت خواهیم کرد. اگر هیچ یک از این دو شرط برقرار نشد، فرومون به روز خواهد شد و گروه جدید از مورچه‌ها ساخته می‌شود و فرآیند بار دیگر ادامه می‌یابد. این فرآیند برگرفته از روش Jensen(JSACO) و Shen می‌باشد که دارای چهار مرحله است[۳۸]:

۱. نمایش و ارائه مسأله
۲. اطلاعات اکتشافی
۳. ساخت و ایجاد جواب‌های ممکن

۴. به روزرسانی فرومون

۳-۵- الگوریتم جستجوی ممنوعه TS

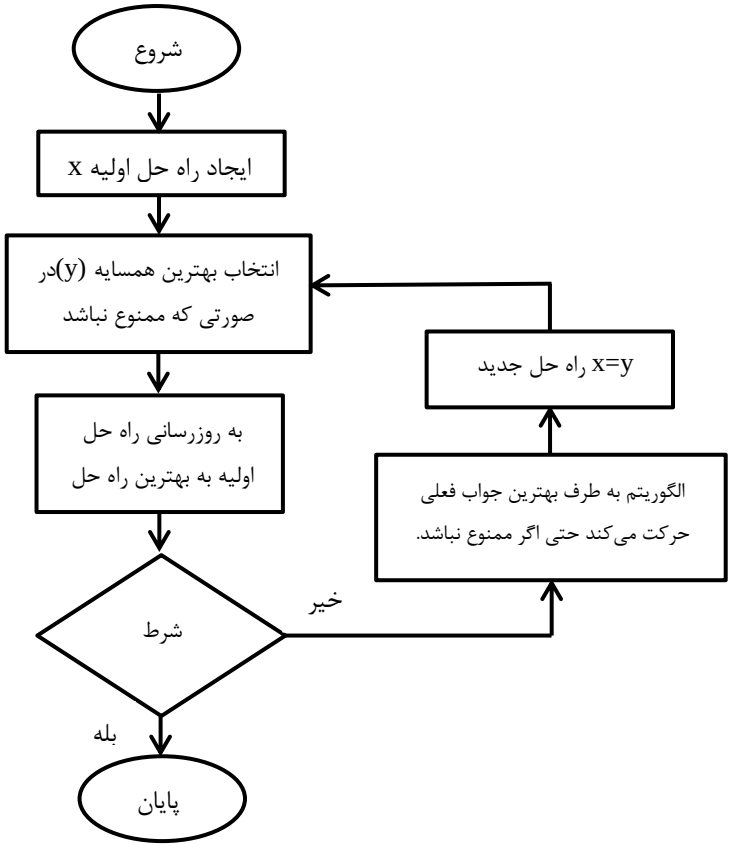
الگوریتم جستجوی ممنوعه (TS)، یک الگوریتم بهینه‌سازی فرااکتشافی است که برای اولین بار در سال ۱۹۸۶ توسط Glover معرفی شد[۳۹]. واژه تابو از تُنگان، زبان مردم جزایر پلینزی در اقیانوس آرام گرفته شده است. این واژه به معنای شیء مقدسی است که به دلیل قداست نباید آن را لمس کرد. بر اساس واژه‌نامه وِبستر، امروزه این واژه در معنای (ممنوعیت چیزی که دارای ریسک است) به کار می‌رود. معنای اخیر واژه تابو، با تکنیک جستجوی ممنوعه کاملاً سازگار است. ریسکی که در الگوریتم جستجوی ممنوعه از آن اجتناب می‌شود، خطر مسیرهای نامناسب است[۴۰]. برای رسیدن به جواب بهینه در یک مسأله بهینه‌سازی، الگوریتم جستجوی ممنوعه ابتدا از یک جواب اولیه، شروع به حرکت می‌کند. در هر دور تکرار الگوریتم، یک همسایگی برای جواب جاری تعریف می‌شود. سپس الگوریتم، بهترین جواب همسایه را از میان همسایه‌های جواب فعلی انتخاب می‌کند. در صورتی که این جواب در لیست ممنوعه[۱] (TL) قرار نداشته باشد، الگوریتم به سوی جواب همسایه حرکت می‌کند، در غیر این صورت، الگوریتم معیاری به نام معیار تنفس[۲] را بررسی خواهد کرد[۴۱]. بر اساس معیار تنفس، اگر جواب همسایه از بهترین جواب یافت شده تاکنون بهتر باشد، الگوریتم به آن حرکت خواهد کرد، حتی اگر آن جواب در لیست ممنوعه باشد. پس از حرکت الگوریتم به جواب همسایه، لیست ممنوعه بروزرسانی می‌شود، به این معنا که حرکت قبل که به وسیله آن به جواب همسایه حرکت کردیم در لیست ممنوعه قرار داده می‌شود تا از بازگشت مجدد الگوریتم به آن جواب و ایجاد سیکل جلوگیری شود. در واقع لیست ممنوعه ابزاری در الگوریتم جستجوی ممنوعه است که توسط آن از قرار گرفتن الگوریتم در بهینه محلی جلوگیری می‌شود. پس از قرار دادن حرکت قبلی در لیست ممنوعه، تعدادی از حرکت‌هایی که قبلا در لیست ممنوعه قرار گرفته بودند، از لیست خارج می‌شوند. مدت زمانی که حرکت‌ها در لیست ممنوعه قرار می‌گیرند، توسط یک پارامتر که زمان ممنوعه نام دارد تعیین می‌شود. حرکت از جواب فعلی به جواب همسایه تا جایی

[1] Tabu List
[2] Aspiration Criterion

ادامه می‌یابد که شرط خاتمه دیده شود. شرط‌های خاتمه متفاوتی می‌توان برای الگوریتم در نظر گرفت. بطور مثال محدودیت تعداد حرکت به جواب همسایه می‌تواند یک شرط خاتمه باشد.

الگوریتم‌های فرا اکتشافی بسیاری برای دستیابی به حداقل یک جواب خوب (نه لزوما بهترین) برای یک مسأله به وجود آمده است. بسیاری از این روش‌ها، از مکانیزم[1] (LS) بهره می‌گیرند. LS را می‌توان یک روال جستجوی تکرارشونده دانست که از یک جواب ممکن و شدنی شروع می‌کند و با انجام اصلاحات انتقالی جزئی، آنرا تا رسیدن به یک بهینه موضعی ادامه می‌دهد. با در نظر داشتن این نکته که در حالت معمول این بهینه موضعی، چیزی بیش از یک جواب متوسط نیست. در LS معمولا کیفیت جواب به دست آمده، تا حد زیادی بستگی به غنای انتقال‌های تعریف شده دارد و این مسأله اساسی در رویکردهای مبتنی بر LS است. اصل اولیه در الگوریتم جستجوی ممنوعه، مجاز دانستن انتقال‌هایی که بهبودی به همراه ندارند، برای ادامه دادن جستجو در LS است، وقتی که به یک بهینه موضعی بر می خوریم. البته در این روش برای اجتناب از دور زدن و رسیدن به جواب‌هایی که پیش از این به دست آمده، از حافظه‌ای به نام Tabu List استفاده می‌کنیم. این حافظه جواب‌های اخیر و یا انتقال‌های اخیر را در خود ضبط می‌کند. در واقع یک TS ساده را می‌توان ترکیبی از یک حافظه کوتاه مدت با LS دانست. انتقال‌های ممنوع، در حافظه‌ای کوتاه مدت[2] ذخیره می‌شوند تا برای مدتی معین انجام مجموعه ای معین از انتقال‌ها را ممنوع سازند. این مدت معین، یا اصطلاحاً Tabu Tenure بنابر الگوریتم حل و نوع مسأله و ماهیت انتقال‌ها متغیر است. حافظه مورد استفاده برای نگهداری انتقال‌های ممنوع، معمولا گردشی و دارای طول ثابت است. در **شکل ۳-۷** فلوچارت الگوریتم جستجوی ممنوعه آمده است.

[1] Local Search
[2] Short Term Memory



شکل ۳-۷: فلوچارت الگوریتم جستجوی ممنوعه Tabu Search

۳-۵-۱- روش جستجو در الگوریتم جستجوی ممنوعه

۱. از یک جواب اولیه آغاز کرده

۲. در هر تکرار همسایه های جواب را می یابد

۳. سپس بهترین همسایه را انتخاب کرده

۴. در صورتی که در لیست ممنوعه نباشد کل الگوریتم به سوی جواب همسایه حرکت می کند

۵. در غیر این صورت الگوریتم معیار تنفس را می گذراند.

۳-۵-۲- معیار تنفس

بر اساس معیار تنفس، اگر جواب همسایه از بهترین جواب یافت شده تاکنون بهتر باشد، الگوریتم به آن حرکت خواهد کرد، حتی اگر در لیست ممنوعه باشد.

فصل ۴

ماشین بردار پشتیبان

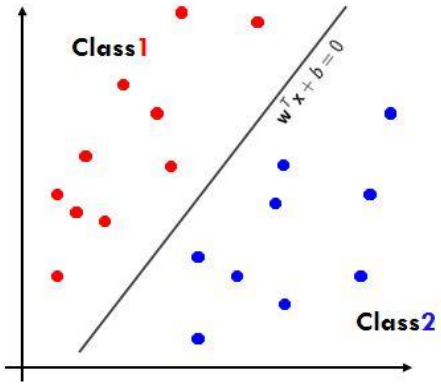
۴-۱- مقدمه

ماشین بردار پشتیبان یکی از روشهای یادگیری با نظارت است که از آن برای طبقه‌بندی و رگرسیون استفاده می‌کنند. این روش از جمله روشهای جدیدی است که در سالهای اخیر کارایی خوبی نسبت به روشهای قدیمی‌تر برای طبقه‌بندی از جمله شبکه عصبی[1] پرسپترون نشان داده است. الگوریتم ماشین بردار پشتیبان اولیه در سال ۱۹۶۳ توسط ولادیمیر وپنیک ابداع شد و در سال ۱۹۹۵ توسط وپنیک و گروهش در آزمایشگاه AT&T Bell پیشنهاد، و برای حالت غیرخطی تعمیم داده شد[۴۲]. این طبقه‌بند در واقع یک طبقه‌بند دیجیتال(دوتایی) است[۴۳]، که توانایی طبقه‌بندی نمونه‌های چند کلاس را نیز داراست. همچنین برای حل مسائل خطی و غیر خطی به کار می‌رود. مبنای کاری طبقه‌بندی کننده SVM دسته بندی خطی داده ها است و در تقسیم خطی داده‌ها سعی می‌کنیم خطی را انتخاب کنیم که حاشیه اطمینان بیشتری داشته باشد. ویژگی اصلی ماشین بردار پشتیبان در یافتن ابر صفحه بهینه برای طبقه‌بندی داده‌های دو کلاس می‌باشد. از لحاظ تئوری قوی‌ترین طبقه‌بند محسوب می‌شود. الگوریتم SVM جزء الگوریتم‌های تشخیص الگو دسته بندی می‌شوند. از SVM در هر جایی که نیاز به طبقه‌بندی الگو، یا دسته‌بندی اشیاء در کلاس خاص باشد استفاده می‌شود. در حال حاضر در بسیاری از زمینه‌های پردازش صوت و تصویر کاربرد دارد و نتایج بسیار خوبی ارائه داده است. الگوریتم ماشین بردار پشتیبان آموزش نسبتا ساده‌ای نسبت به شبکه‌های عصبی دارد و برخلاف شبکه عصبی در ماکزیمم محلی‌گیر نمی‌کند. و برای داده‌ها با ابعاد بالا تقریبا خوب جواب می‌دهد.

۴-۲- ماشین بردار پشتیبان خطی

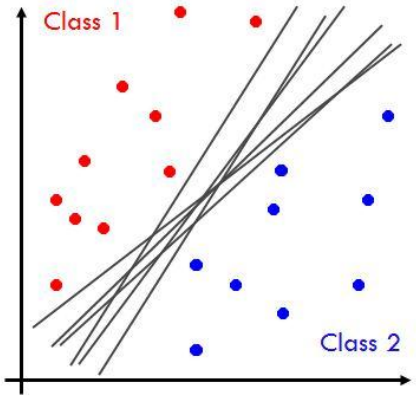
یک مسأله طبقه‌بندی خطی برای جدا سازی داده های آموزشی دو کلاس در نظر بگیرید. فرض کنید $X_i , i=1,2,\ldots,N$ مجموعه بردار های ویژگی مربوط به داده های آموزشی باشد که به صورت خطی از هم جداپذیرند و در دو کلاس ۱ و ۲ به صورت **شکل ۴-۱** طبقه‌بندی شده‌اند.

[1] Neural Network



شکل ۴-۱: داده‌های آموزشی در دو کلاس جدایی پذیر خطی

هدف بدست آوردن ابر صفحه[1] $g(x)=W^TX+b=0$ به نحوی است که تمامی داده‌های آموزشی بطور صحیح طبقه‌بندی شوند. به صورت کلی این ابر صفحه یکتا نبوده و می‌توان مقادیر مختلفی برای w وb بدست آورد. **شکل ۴-۲** چند نمونه از ابر صفحه هایی را که می‌توان به منظور طبقه‌بندی صحیح خطی داده‌های دو کلاس در نظر گرفت، نشان می‌دهد.

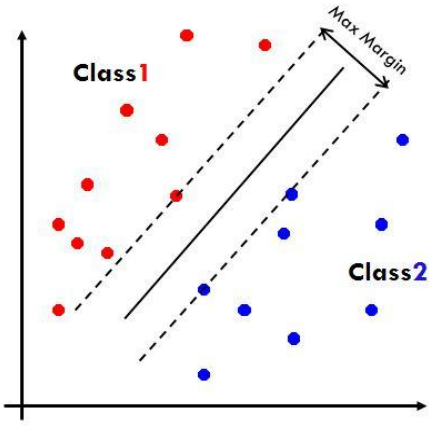


شکل ۴-۲: ابر صفحه‌های مجزا ساز

همه ابر صفحه‌های نشان داده شده در شکل ۴-۲ عمل جداسازی را به درستی انجام می‌دهند، اما تنها یک ابر صفحه وجود دارد که بیشترین فاصله نسبت به داده‌های هر دو کلاس دارد، که به آن ابر صفحه مجزا ساز بهینه[2] گفته می‌شود و انتظار می‌رود بتواند مرز بدست آمده را به تمام محدوده‌های ممکن تعمیم دهد[۴۴]. ابر صفحه مجزاساز بهینه در شکل ۴-۳ نشان داده شده است. این ابر صفحه به دلیل این که بیشترین فضا تا نزدیکترین داده از هر کلاس را در دو طرف خود ایجاد می‌کند، بهترین طبقه‌بندی با

[1] Hyperplane
[2] Optimal Separating hyperplane

کمترین میزان خطا را ارائه می‌دهد. بنابراین این نوع کلاسه‌بندی، در زمان اجرا بر روی نمونه‌های آزمایشی نتایج بهتری را از خود نشان می‌دهد. این نتیجه موضوعی مهم در طراحی طبقه‌بندها می‌باشد که آن را قدرت تعمیم طبقه‌بند می‌نامند.



شکل ۴-۳: ابر صفحه مجزاساز بهینه با حداکثر مقدار حاشیه

فرض می‌شود مسأله‌ای برای جداسازی مجموعه نمونه‌های آموزشی که متعلق به دو کلاس جداگانه هستند، وجود دارد. طبق معادله (۴-۱) بردار شاخص y_i شامل مقادیر ۱+ و۱- را به نحوی تعریف می‌کنیم، که برای کلاس ۱ مقدار ۱+ و برای کلاس ۲ مقدار ۱- داشته باشد.

$$y_i = \begin{cases} 1 & if \;\; x_i \; in \; class \; 1 \\ -1 & if \;\; x_i \;\; in \; class \; 2 \end{cases} \qquad (۱-۴)$$

هر ابر صفحه دقیقا به وسیله جهت و مکانش در فضا مشخص می‌شود، که w جهت ابر صفحه و b مکان آن را در فضا مشخص می‌کند. تابع تصمیم‌گیری را می‌توان به صورت معادله (۴-۲) و (۴-۳) تعریف نمود:

$$d(x) = sign(W^T X + b) \qquad (۲-۴)$$

یعنی:

$$\begin{cases} (W^T X_i + b) > 0 & if \;\; y_i = +1 \\ (W^T X_i + b) < 0 & if \;\; y_i = -1 \end{cases} \qquad (۳-۴)$$

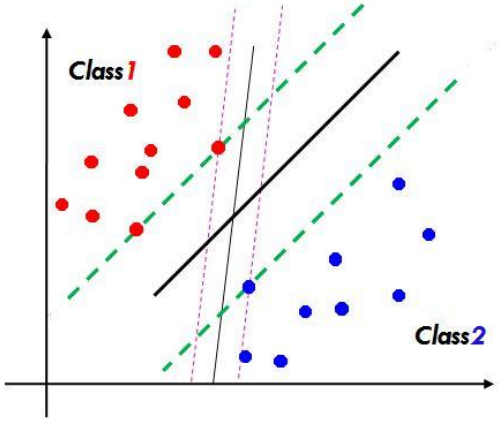
ابر صفحه‌ای که در این گونه از مسائل برای طبقه بندی داده ها استفاده می‌شود باید دارای دو ویژگی خاص باشد: اول اینکه دارای کمترین میزان خطای ممکن باشد، و از سویی دیگر از داده‌های هر کلاس

بیشترین فاصله ممکن را داشته باشد. در این حالت اگر رابطه (۴-۳) برای صفحه مجزاساز در نظر گرفته شود، داده‌های آموزشی در بالا و پایین این صفحه قرار خواهند گرفت. که به ترتیب برای $y_i=+1$ ، $(W^TX_i+b)>0$ و برای $y_i=-1$، $(W^TX_i+b)<0$ خواهد بود. بر اساس شرایط بیان شده، زمانی مجموعه‌ای از نقاط به صورت بهینه با یک صفحه جداسازی می‌شوند که:

۱. بدون اشتباه در کلاس مربوط به خود قرار گرفته باشند.

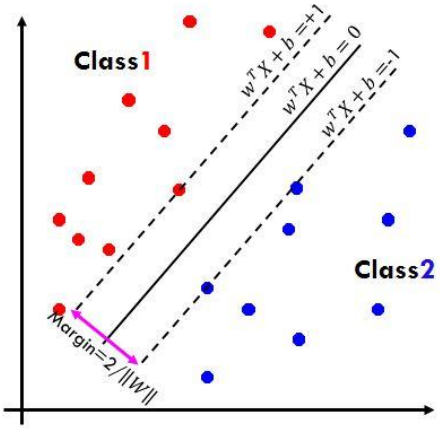
۲. فاصله بین نزدیکترین نقاط هر کلاس داده تا صفحه جدا کننده بیشینه باشد.

بر این اساس، پارامترهای w و b باید به گونه‌ای محاسبه گردند که دو شرط ذکر شده برقرار باشد. بنابر این در طبقه‌بند ماشین بردار پشتیبان خطی قصد بر این است که ابر صفحه‌ای را بدست آوریم که علاوه بر طبقه‌بندی صحیح داده‌های آموزشی، حاشیه بین دو کلاس را نیز بیشینه کند. **شکل ۴-۴** تفاوت بین دو ابر صفحه مجزاساز را نشان می‌دهد.



شکل ۴-۴: ابر صفحه‌های مجزاساز: حاشیه ایجاد شده توسط خط ضخیم‌تر بیشتر از حاشیه ایجاد شده

ابر صفحه‌ای که در **شکل ۴-۴** با خط ضخیم‌تر نشان داده شده است ابر صفحه مورد نظر جهت جداسازی دو کلاس می‌باشد، زیرا علاوه بر جداسازی دو کلاس، حاشیه بین ابر صفحه و دو کلاس را هم بیشینه کرده است. فرض کنیم معادله ابر صفحه مجزاساز بهینه به صورت $W^TX_i+b=0$ باشد، بنابراین معادله ابرصفحه‌های حاشیه‌ای دردو طرف ابرصفحه مجزاساز به صورت $W^TX_i+b=+1$ و $W^TX_i+b=-1$خواهد بود. **شکل ۴-۵** معادلات در نظر گرفته شده برای حاشیه‌ها و صفحه مجزاساز بهینه را نشان می‌دهد.



شکل ۴-۵: ابر صفحه مجزاساز بهینه و حاشیه‌های آن

چون داده‌ها به صورت خطی جدایی‌پذیر هستند پس هیچ کدام از داده‌ها روی صفحه $W^TX+b=0$ قرار نمی‌گیرند. بنابراین می‌توان بجای رابطه (۴-۳)، از رابطه (۴-۴) استفاده کرد.

$$(W^TX_i + b) = \begin{cases} \geq 1 \ if \ y_i = class1 \\ \leq -1 \ if \ y_i = class2 \end{cases} \quad (۴-۴)$$

و می‌توان آن را به فرم ساده شده رابطه (۴-۵) بازنویسی نمود.

$$y_i(w^Tx_i + b) \geq 1 \quad , i = 1,2, \dots , N \quad (۴-۵)$$

جهت معرفی بهترین صفحه مجزاساز بایستی صفحه بیشترین فاصله را از کلاسها داشته باشد و این فاصله مقدار ماکزیمم شود. حال فاصله بین این دو حاشیه بصورت رابطه (۴-۶) بدست می‌آید.

$$\frac{|(w^Tx + b - 1) - (w^Tx + b + 1)|}{\|w\|} = \frac{2}{\|w\|} \quad (۴-۶)$$

بر اساس رابطه (۴-۶) اگر مقدار $\frac{2}{\|w\|}$ ماکزیمم گردد مقدار حاشیه مورد نظر در بیشترین مقدار خود قرار دارد. برای سادگی کار بجای ماکزیمم کردن $\frac{2}{\|w\|}$ می‌توان مقدار $\|w\|^2$ مخرج را مینیمم نمود. یا به عبارتی می‌توان جمله را بصورت $\frac{1}{2}w^Tw$ نوشت. بنابراین بر اساس شزایط بیان شده، مسأله ماشین بردار پشتیبان به فرم زیر تبدیل می‌شود که به آن یک مسأله درجه دو گفته می‌شود.

$$minimize \quad J(w) = \frac{1}{2}w^Tw \quad (۴-۷)$$

$$subject\ to \quad y_i(w^Tx_i + b) \geq 1, \quad i = 1,2, \dots , N$$

چون فرم رابطه (۴-۷) یک مسأله درجه دو با شرط نامساوی است، بنابراین مقدار تابع هدف یکتا خواهد بود، یعنی فقط یک نقطه اکسترمم دارد. این یکی از ویژگی‌های مهم ماشین بردار پشتیبان نسبت به شبکه های عصبی چند لایه است که در آن تعداد زیادی نقاط مینیمم محلی وجود دارد[۴۴].

در مسأله درجه دو فوق برای مینیمم سازی و بدست آوردن مقادیر بهینه w و b با توجه به قید نامساوی موجود بایستی از ضرایب لاگرانژ استفاده کرد. ضرایب لاگرانژ که گاهی ضرایب نامعین نیز خوانده می‌شود، جهت شناسایی نقاط خاص تابعی که دارای چندین متغیر و محدودیت است، مورد استفاده قرار می‌گیرد. در واقع اگر هدف مینیمم‌سازی تابع F(x) با توجه به محدودیت $g(x)\geq 0$ باشد، باید تابع لاگرانژ که به صورت رابطه (۴-۸) تعریف می‌شود، مینیمم شود.

$$L(x,\alpha) \equiv f(x) + \alpha g(x) \qquad (۴-۸)$$

که در آن $\alpha \geq 0$ ضریب لاگرانژ خوانده می‌شود.

با در نظر گرفتن تابع لاگرانژ، فرم با قید نامساوی مسأله درجه دو (۴-۷) به فرم بدون قید رابطه (۴-۹) تبدیل می‌شود که به آن تابع لاگرانژ اولیه گفته می‌شود.

$$L_p(w,b,\alpha) = \frac{1}{2}w^T w - \sum_{i=1}^{N} \alpha_i [y_i(w^T x_i + b) - 1] \qquad (۴-۹)$$

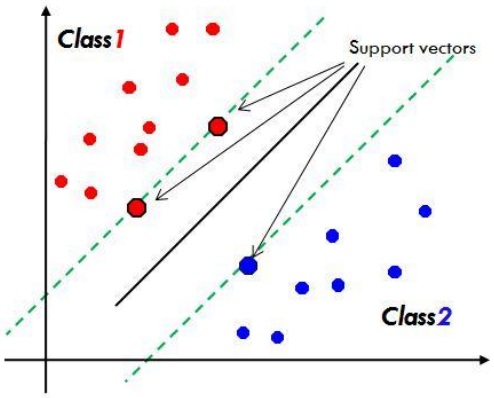
برای مینیمم کردن معادله (۴-۹) باید نقاط ایستان تابع لاگرانژ را بدست آورد. برای رسیدن به این هدف شرایط KKT اعمال می‌شود که بصورت رابطه (۴-۱۰) است.

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w_0 = \sum_{i=1}^{N} \alpha_i x_i y_i$$
$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N} \alpha_i y_i \qquad (۴-۱۰)$$

حال اگر مقدار W بدست آمده از مشتقات جزئی رابطه (۴-۱۰) در خود رابطه (۴-۹) قرار داده شود، معادله اساسی ماشین های بردار بصورت رابطه (۴-۱۱) معرفی می‌شود که فرم دوگان معادله لاگرانژ (۴-۹) است.

$$Max \qquad L_d(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} y_i y_j \alpha_i \alpha_j x_i^T x_j$$
$$S.T. \begin{cases} \alpha_i \geq 0 \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases} \qquad (۱۱-۴)$$

با حل معادله بالا مقادیر ضرایب لاگرانژ یعنی α برای هر یک از بردارهای آموزشی بدست خواهد آمد، که با توجه به محدودیت ذکر شده در رابطه (۴-۱۱) مقدار بزرگتر یا مساوی صفر خواهد بود. از میان بردارهای آموزشی کلاس ۱و۲، آنهایی که مقدار α متناظر با آنها بزرگتر از صفر است، بردارهای پشتیبان نامیده می‌شوند. در محاسبات برای بدست آوردن ابر صفحه بهینه فقط این بردارها استفاده می‌شوند، بنابراین حجم محاسبات تا حد زیادی کاهش می‌یابد. بردارهای پشتیبان درست بر روی دو ابر صفحه $w^Tx+b=\pm1$ قرار می‌گیرند. این بردار ها در **شکل ۴-۶** برجسته تر نشان داده شده‌اند.



شکل ۴-۶: بردارهای پشتیبان کلاسهای ۱و۲ بر روی ابر صفحه‌های حاشیه‌ای

از رابطه (۴-۱۰) مقدار مطلوب w_0 به دست آمد. مقدار بهینه b نیز از طریق رابطه $b = y_i - w^T x_i$ و میانگین‌گیری از تمامی مقادیر به دست آمد، محاسبه می‌شود. معادله کلی محاسبه مقدار بهینه b را می‌توان به صورت رابطه (۴-۱۲) بیان نمود.

$$b_0 = \frac{1}{SV} \sum_{i=1}^{SV} (y_i - w^T x_i) \qquad (۱۲-۴)$$

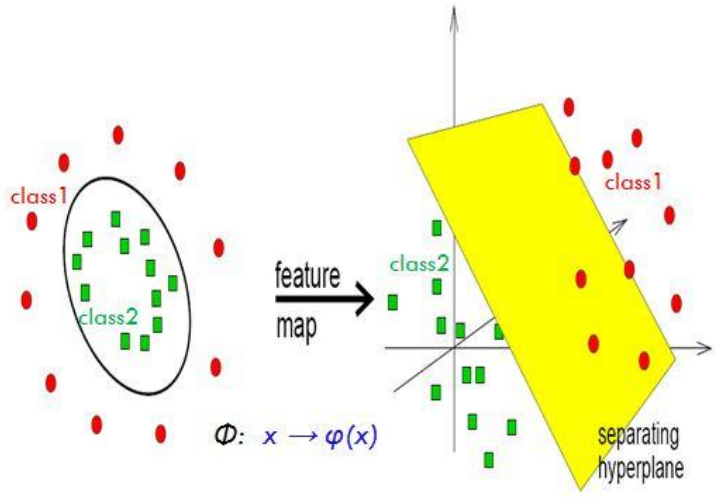
که در آن SV مشخص کننده تعداد بردارهای پشتیبان است. با بدست آوردن مقادیر w_0 و b_0 می‌توان به ابرصفحه مجزاساز بهینه دست یافت. با داشتن این مقادیر و با توجه به رابطه تابع تصمیم ماشین بردار پشتیبان، یعنی رابطه (۴-۲) برای طبقه‌بندی یک داده ناشناخته مثل x ، اگر مقدار تابع تصمیم بزرگتر از صفر باشد، داده در کلاس ۱ طبقه بندی خواهد شد، و اگر مقدار تابع تصمیم کوچکتر از صفر باشد، داده در کلاس ۲ دسته بندی می‌شود.

۴-۳- ماشین بردار پشتیبان غیرخطی

در بخش پیشین ماشین بردار پشتیبان را به عنوان یک طبقه‌بند خطی بهینه معرفی کردیم. سوالی که در اینجا مطرح است این است که اگر داده‌ها به صورت خطی جداپذیر نباشد چه باید کرد؟ آیا می‌توان ایده ماشین بردار پشتیبان خطی را به سیستم‌های غیرخطی تعمیم داد؟

ماشین بردار پشتیبان در ابتدای کار، تنها برای سیستم‌های خطی به کار گرفته می‌شد و صفحه مجزاساز بهینه تنها برای حالت خطی وجود داشت. این در حالی بود که در بسیاری از مسائل طبقه بندی و رگرسیون، راه حل خطی جواب مناسبی را ارائه نمی‌کرد. مطالعات گسترده انجام شده در این زمینه تئوری مرسر را جهت حل این مشکل ارائه داد تا ماشین بردار پشتیبان بتواند مسائلی که به صورت خطی جداپذیر نیستند را نیز پشتیبانی نماید[۴۵]. ایده اصلی این تئوری انتقال برداری مانند X از فضای محدود (فضای ورودی) به فضای بالاتر (فضای ویژگی یا فضای هیلبرت) با استفاده از تبدیل هیلبرت[1] و طبقه بندی آن در فضای بالا بود. در این شرایط برداری مانند X در فضای بالاتر به صورت $\varphi(x)$ نوشته می‌شود. **شکل ۴-۷** نحوه انتقال از فضای ورودی به فضای ویژگی را نشان می‌دهد.

[1] Hilbert Transform



شکل ۴-۷: انتقال فضای ورودی (سمت چپ) به فضای ویژگی (سمت راست) با تبدیل هیلبرت

با استفاده از این قضیه با تبدیل X به $\varphi(x)$ در معادلات (۴-۱۰) و (۴-۱۲)، به راحتی می‌توان روابط بردار وزن و بایاس ماشین بردار پشتیبان غیرخطی را نیز به صورت رابطه (۴-۱۳) بدست آورد.

$$w_0 = \sum_{i=1}^{N} \alpha_i \varphi(x_i) y_i$$
$$b_0 = \frac{1}{SV} \sum_{i=1}^{SV} (y_i - w^T \varphi(x_i)) \qquad (۱۳-۴)$$

و در نهایت با بکارگیری روابط بالا، تابع تصمیم ماشین بردار پشتیبان غیرخطی به صورت رابطه (۴-۱۴) خواهد بود.

$$d(x) = sign(W^T \varphi(x_i) + b) \qquad (۱۴-۴)$$

۴-۴- ماشین بردار پشتیبان چندکلاسه

ماشین بردار پشتیبان اساساً یک طبقه‌بند دوتایی است. اما می‌توان با روشهای مختلفی از این طبقه‌بند برای حل مسائل چند کلاس نیز استفاده کرد. چهار روش برای طبقه‌بندی ماشین بردار پشتیبان چند کلاسه وجود دارد که عبارتند از:

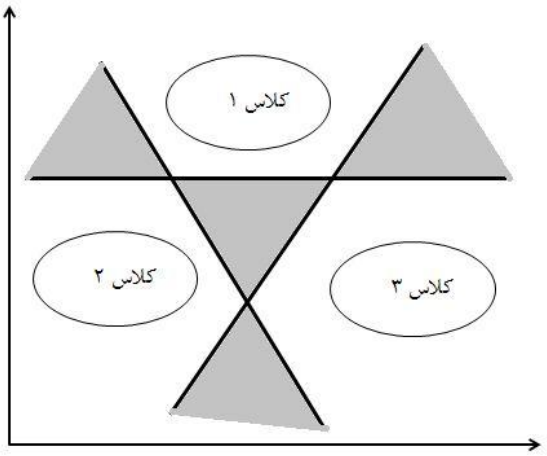
۱. روش یک کلاس در برابر همه

۲. روش طبقه‌بندهای دوتایی

۳. روش کدخروجی تصحیح کننده خطا

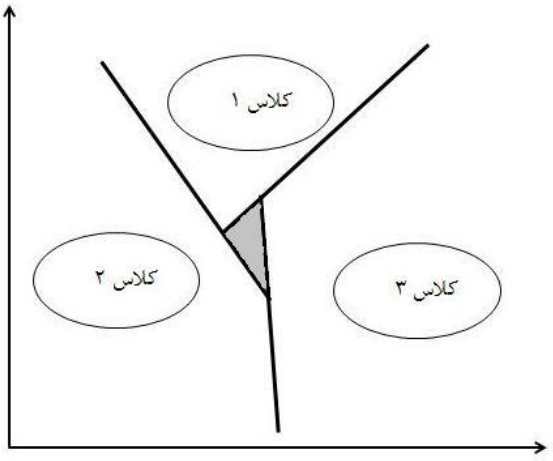
۴. روش همه کلاسها در یک کلاس

در روش یک کلاس در برابر همه، اگر n کلاس داشته باشیم، n تا طبقه بند ماشین بردار پشتیبان دوتایی تشکیل خواهد شد بطوری که برای طبقه بند دوتایی i ام، کلاس i از بقیه کلاسها جدا می‌شود. اما با این روش اگر از توابع تصمیم گسسته استفاده کنیم، نواحی طبقه بندی نشده‌ای طبق **شکل ۴-۸** به وجود خواهد آمد[۴۴].



شکل ۴-۸: نواحی طبقه‌بندی نشده در روش طبقه بند ی یک کلاس در برابر همه

برای حل این مشکل، کرِبل روش طبقه بندی‌های دوتایی را ارائه نمود [۴۶]. که در آن برای حل مسأله طبقه‌بندی n کلاس، از n(n-1)/2 مسأله طبقه بندی دوتایی استفاده می‌شود. این روش نسبت به روش یک کلاس در برابر همه، روش بهتری است اما باز هم نواحی طبقه بندی نشده‌ای طبق شکل ۴-۹ وجود خواهد داشت.



شکل ۴-۹: ناحیه طبقه‌بندی نشده درطبقه بندی دوتایی

البته نواحی طبقه بندی نشده با استفاده از یکی از روشهای: توابع عضویت، درخت‌های تصمیم، روش کد خروجی تصحیح کننده خطا و یا روش همه کلاسها در یکبار اصلاح خواهد شد.

۴-۵- کتابخانه LIBSVM

برای ماشین بردار پشتیبان پیاده‌سازی‌های گوناگونی در طول سالیان اخیر تهیه شده است .اغلب این نرم افزارها در محیط‌های دانشگاهی طراحی و پیاده سازی شده‌اند، اما در این میان می‌توان محصولات تجاری را نیز پیدا کرد. نرم افزارهای گوناگون از لحاظ شرایط استفاده نیز با یکدیگر تفاوت دارند به گونه-ای که برخی فقط برای استفاده‌های دانشگاهی و علمی قابل استفاده هستند و از برخی دیگر در کاربردهای تجاری نیز می‌توان استفاده کرد.

با در نظر گرفتن تمامی موارد فوق و همچنین مواردی مانند سهولت استفاده و استفاده از روشهای جدید و به روز در پیاده‌سازی و با توجه به تجربه شخصی استفاده از چندین نرم‌افزار موجود، نرم‌افزاری که برای استفاده در این پروژه انتخاب شده است، نرم افزار libsvm می‌باشد که توسط آقای جان چنگ و همکارانش در دانشگاه تایوان پیاده‌سازی شده است.

این نرم‌افزار در حقیقت با هدف طراحی یک کتابخانه برای انجام عملیات یادگیری و استفاده از ماشین بردار پشتیبان طراحی شده است. این نرم‌افزار و به عبارتی کتابخانه در اصل با استفاده از زبان C++

طراحی شده است، اگرچه در حال حاضر با استفاده از زبان Java نیز پیاده‌سازی شده است و همچنین رابط‌های گوناگونی برای استفاده از آن در محیط‌ها و زبان‌های مختلف از جمله Matlab وجود دارد[۴۷]. در مرجع ذکر شده اطلاعات کاملی در مورد این نرم افزار وجود دارد. یکی از مستندات موجود در این سایت یک راهنمای کوتاه برای آشنایی با SVM و همچنین استفاده از libsvm است[۴۸]. این مستند را می‌توان به عنوان خلاصه‌ای از مطالب ارائه شده در جلسه مربوط به ماشین بردارپشتیبان و همچنین برای یادآوری برخی مفاهیم استفاده شود.

هدف از این قسمت آشنایی دانشجویان با یکی از نرم‌افزارهای موجود برای ماشین بردار پشتیبان و چگونگی استفاده از آن است. به همین منظور یک سری از مجموعه داده‌ها برای عمل دسته‌بندی در نظر گرفته شده است و بعد از تبدیل به فرمت قابل استفاده توسط libsvm نتایج مختلف حاصل شده در جدولی آورده شده‌اند. ما این بررسی را به دو قسمت مجموعه داده‌های دوکلاسی و مجموعه داده‌های از دوکلاس به بالا تقسیم‌بندی نموده ایم.

مجموعه‌های که مورد استفاده قرارگرفته‌اند در جدول (۶-۱) آورده شده‌اند. این مجموعه داده‌ها از بین مجموعه‌های موجود در پایگاه UC Irvine Machine Learning Repository انتخاب شده‌اند. در انتخاب مجموعه‌ها سعی شده است تا مجموعه‌هایی انتخاب شوند که دارای تنها ویژگی‌های عددی هستند.

از آنجائیکه libsvm حالت چندکلاسه را نیز پشتیبانی می‌کند. با استفاده از توابع این کتابخانه و تغییر اندکی، از آن به عنوان طبقه‌بند چند کلاسه استفاده نمودیم. مجموعه‌های در نظر گرفته شده به صورت مسائل طبقه‌بندی دوکلاسه و یا چندکلاسه هستند. تنها تفاوت بین مجموعه‌ها این است که مجموعه های بزرگتر و یا حالت چندکلاسه ممکن است به زمان بیشتری برای آموزش و آزمایش نیاز داشته باشد.

برای به دست آوردن نتایج، ابتدا باید مجموعه داده‌ها را به دو بخش آموزش (Training) وآزمایش (Testing) تقسیم کرد .در مورد اکثر مجموعه‌ها این دسته‌بندی در مستندات همراه داده‌ها ذکر شده است .در این مجموعه‌ها باید از همین دسته‌بندی استفاده کرد. در مورد مجموعه‌هایی که این تقسیم بندی برای آن‌ها ذکر نشده است، با نظر خود و قبل از شروع به دست آوردن نتایج، داده‌ها را به دو دسته آموزش و آزمایش تقسیم کنید.

سعی شده است برای این کار از استدلالهای منطقی استفاده شود، مثلا اگر داده‌ها مربوط به تشخیص کلمه گفته شده است، نمونه‌های مربوط به برخی گوینده‌ها را برای آموزش و بقیه گویندگان را برای آزمایش استفاده کنید نه اینکه از گوینده چند نمونه را برای آموزش و چند نمونه را برای آزمایش استفاده کنید. دقت کنید که همانطور که در مستند آموزش با SVM که در بالا ذکر شد نیز آمده است، ابتدا باید داده‌ها به بازه [0,1] یا [-1,1] تبدیل شود. برای این کار می‌توانید از نرم‌افزاری که در بسته LIBSVM نیز موجود است استفاده کنید. این مسأله تاثیر فراوانی بر روی خروجی دارد .به علاوه در برخی موارد بهتر است که واریانس داده‌ها نیز 1 شود .این مورد را می‌توان به صورت اختیاری انجام داد (به عبارتی انجام آن اجباری نیست) و نتایج را درحالتی که واریانس 1 شده با حالتی که واریانس داده تغییر داده نشده مقایسه کرد.

فصل ۵

الگوریتم پیشنهادی جهت استخراج و انتخاب ویژگی

۵-۱- الگوریتم پیشنهادی جهت استخراج ویژگی

انتخاب روش استخراج ویژگی مناسب یکی از مهمترین مراحل در بازشناسی الگو محسوب می‌شود. در این پایان‌نامه از ترکیب دو روش هیستوگرام گرادیان و مکان مشخصه توسعه یافته برای استخراج ویژگی بر روی داده‌های ارقام دستنویس فارسی هدی استفاده شده است که در ادامه به شرح آنها می‌پردازیم.

۵-۱-۱- روش هیستوگرام گرادیان

ویژگی‌های هیستوگرام گرادیان برای اولین بار توسط آقای خسروی و همکارش برای شناسایی ارقام دستنویس فارسی استفاده شده است. نتایج بدست آمده نشان از دقت و سرعت بهتر این الگوریتم نسبت به فیلتر گابور دارد[۴۹]. مراحل کار در این روش به صورت زیر می‌باشد.

۱- ابتدا تصویر ورودی پایگاه داده را به ابعاد ۶۴×۶۴ نرمال‌سازی کرده و سپس تصویر نرمال شده را به ۱۶ بلوک غیر همپوشان ۱۶×۱۶ تقسیم می‌نماییم.

۲- برای هر پیکسل از هر بلوک، گرادیان با استفاده از عملگر سوبل که در جدول (۵-۱) نشان داده شده محاسبه می‌شود. روش کار بر اساس معادله‌های (۵-۱) تا (۵-۳) می‌باشد.

$$g(x,y) = [g_x, g_y]^T \quad where \tag{۵-۱}$$

$$\begin{aligned} g_x(x,y) &= f(x+1,y-1) + 2f(x+1,y) \\ &+ f(x+1,y+1) - f(x-1,y-1) \\ &- 2f(x-1,y) - f(x-1,y+1) \end{aligned} \tag{۵-۲}$$

$$\begin{aligned} g_y(x,y) &= f(x-1,y+1) + 2f(x,y+1) \\ &+ f(x+1,y+1) - f(x-1,y-1) \\ &- 2f(x,y-1) - f(x+1,y-1) \end{aligned} \tag{۵-۳}$$

جدول ۵-۱: الف و ب عملگرهای سوبل

۱	۲	۱
۰	۰	۰
-۱	-۲	-۱

الف

۱	۰	-۱
۲	۰	-۲
۱	۰	-۱

ب

عبارت $f(x,y)$ مقدار پیکسل در مکان (x,y) می‌باشد.

۳- برای هر پیکسل فاز و اندازه گرادیان با استفاده از معادله‌های (۵-۴) و (۵-۵) محاسبه می‌شود.

$$p(x,y) = tan^{-1}(g_x, g_y) \quad (۵-۴)$$

$$A(x,y) = \sqrt{{g_x}^2 + {g_y}^2} \quad (۵-۵)$$

۴- فاز به ۱۶ زاویه از صفر تا $۲\pi\frac{۱۵}{۱۶}$ کوانتیزه شده سپس برای هر بلوک ۱۶ ویژگی مطابق با ۱۶ فاز با استفاده از معادله (۵-۶) استخراج شده است.

$$F_\theta = \sum_{x_\theta, y_\theta} A(x,y) \quad (۵-۶)$$

عبارت F_θ ویژگی متحد با فاز θ می‌باشد. عبارت های x_θ و y_θ تمام مختصات‌هایی می‌باشند که فازشان برابر θ می‌باشد. در نتیجه برای ۱۶ چرخش به تعداد ۱۶×۴×۴=۲۵۶ ویژگی با استفاده از عملگر سوبل بدست آمده و نرمالیزه شده‌اند. می‌توان تعداد چرخش‌ها را به تعداد کمتری مانند ۴ یا ۸ جهت نیز کوانتیزه کرد. تعداد ویژگی‌ها برای چرخش‌های مختلف مطابق جدول (۵-۲) می‌باشد.

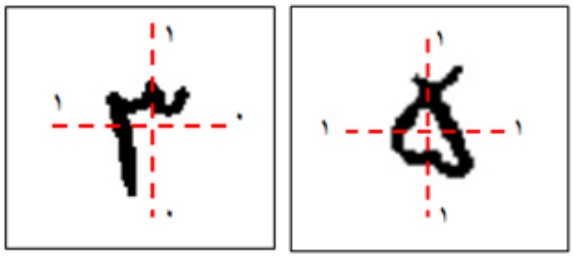
جدول ۵-۲: تعداد ویژگی‌ها برای چرخش‌های مختلف

تعداد ویژگی‌ها	تعداد چرخش
۱۶×۴×۴= ۲۵۶	۱۶
۸×۴×۴= ۱۲۸	۸
۴×۴×۴= ۶۴	۴

در این گزارش ما از ۱۶ چرخش استفاده نموده‌ایم پس تعداد ویژگی‌هایی که با استفاده از این روش بدست می‌آید طبق جدول (۵-۲) برابر ۲۵۶ ویژگی می‌باشد.

۵-۱-۲- روش مکان مشخصه توسعه یافته

مکان مشخصه توسعه یافته روش خوبی برای ترکیب با سایر روش‌های استخراج ویژگی است چون اطلاعاتی می‌دهد که سایر روش‌های استخراج ویژگی چنین اطلاعاتی در اختیار قرار نمی‌دهد. مراحل کار در این روش این گونه است که به ازای ۶۴×۶۴ پیکسل در ۴ جهت بالا، پایین، چپ و راست تعداد گذرها شمارش می‌شود. گاهی اوقات برای غلبه بر نویز تعداد گذرها محدود می‌گردد همچنین با این عمل تعداد ویژگی‌ها نیز محدود می‌شوند. سپس اعداد بدست آمده در جهت‌های مختلف را به ترتیب به صورت (راست، بالا، چپ، پایین) چیده و این رشته به عددی بر مبنای یک واحد بیشتر از ماکزیمم عدد بدست آمده در رشته بالا برده می‌شود. **شکل ۵-۱** جزئیات این روش را نمایش داده است. رشته تولیدی برای عدد ۳ برابر (۰۱۱۰) و برای عدد ۵ برابر (۱۱۱۱) می‌باشد.



شکل ۵-۱: جزئیات روش مکان مشخصه توسعه یافته

اگر رشته بدست آمده به صورت $(N_3N_2N_1N_0)$ نشان داده شود عدد خروجی از انتقال رشته فوق بر مبنای ۳ و۴ به صورت معادله (۵-۷) بدست می‌آید.

$$(N_3N_2N_1N_0) = N_3\,(۴\times۳\times۴\times۱) + N_2\,(۳\times۴\times۱) + N_1\,(۴\times۱) + N_0\,(۱) \qquad (۵-۷)$$

در ادامه کار بایستی نمودار فراوانی اعداد بدست آمده از انتقال رشته اعداد به مبنای مشخص شده را بدست آورده و به عنوان بردار ویژگی استفاده گردد. روش کار بدین صورت است که یک بردار به ابعاد ۱×۱۴۴ ایجاد نموده و به درایه عدد بدست آمده که مقداری بین ۰ تا ۱۴۳ است یک واحد اضافه می-

نماییم در ابتدا مقدار کل درایه‌های بردار برابر صفر می‌باشند. این کار را تا جایی ادامه می‌دهیم که تا کل پیکسلهای تصویر ورودی جاروب شوند. همان گونه که بیان شد در این مقاله گذرهای افقی به مبنای ۴ و گذرهای عمودی به مبنای ۳ منتقل شده است. این بدین معنی است که در جهت‌های افقی بایستی تعداد گذرها به ۳ و در جهت عمودی تعداد گذرها به مقدار ۲ محدود گردد. بنابراین بیشترین تعداد بردار ویژگی زمانی رخ می‌دهد که با شمارش تعداد گذرها رشته (۲۳۲۳) تولید شود. در مرحله بعد این رشته بدست آمده به مبنای ذکر شده در دو جهت عمودی و افقی منتقل می‌شود. طبق جدول (۵-۳) کمترین و بیشترین مقدار طول بردار ویژگی از مجموع انتقال یافته اعداد به پایه‌های مربوطه از رشته (۰۰۰۰) تا (۲۳۲۳) برابر[۰ تا ۱۴۳] است. حال طبق توضیحات داده شده بایستی رشته اعداد فوق به مبنای ۳ و۴ برده شود. جدول (۵-۳) این انتقال را نمایش داده است.

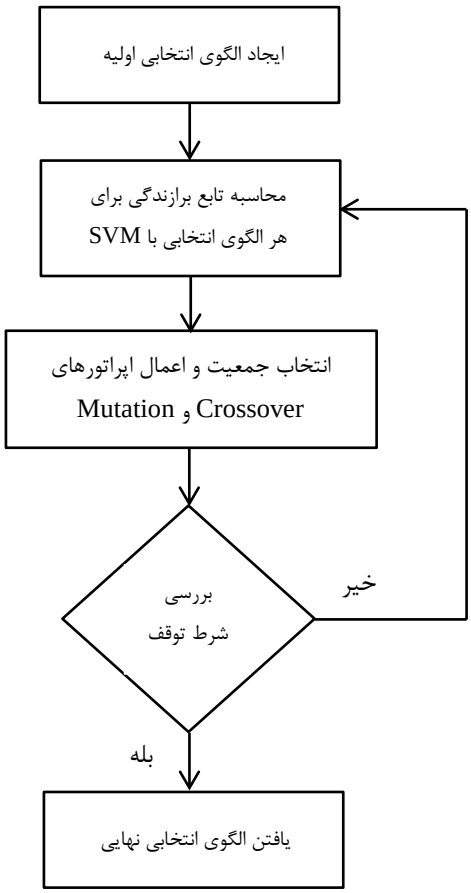
جدول ۵-۳: انتقال رشته اعداد به مبنای ۳ و۴

رشته اعداد	تعداد ویژگی‌ها
۰۰۰۰	۰ × (۳×۴×۱) + ۰ × (۴×۳×۴×۱) + ۰ × (۱) + ۰ × (۴×۱) + ۰ =۰
۲۳۲۳	۲ × (۳×۴×۱) + ۳ × (۴×۳×۴×۱) + ۳ × (۱) + ۲ × (۴×۱) + ۲ =۱۴۳

در نتیجه طول بردار ویژگی که در این روش بدست می‌آید برابر ۱۴۴ ویژگی می‌باشد. نهایتاً طول بردار ویژگی اصلی از مجموع طول بردار ویژگی ۲ روش فوق برابر ۴۰۰ (۲۵۶+۱۴۴) است.

۵-۲- انتخاب ویژگی با استفاده از الگوریتم ژنتیک GA

در این بخش ما روش انتخاب ویژگی با الگوریتم ژنتیک را که پیاده سازی نموده‌ایم را شرح می‌دهیم. با توجه به توضیحات داده شده می‌توان مراحل انتخاب ویژگی با استفاده از الگوریتم ژنتیک را به صورت چارت نشان داده شده در **شکل ۵-۲** نشان داد.



شکل ۵-۲: فلوچارت انتخاب ویژگی با استفاده از الگوریتم GA

مرحله ۱: در این مرحله با استفاده از توابع موجود در نرم‌افزار متلب ماتریسی تصادفی شامل فقط اعداد ۰و۱ را تولید می‌نماییم که ۰ نشانه عدم انتخاب ویژگی و۱ نشان دهنده انتخاب ویژگی متناظر می-باشد. تعداد سطر این ماتریس برابر تعداد طول بردار ویژگی و تعداد ستونها آن برابر تعداد الگوهای انتخابی می‌باشد که ما برای حل مسأله در نظر می‌گیریم. این تعداد نه می‌تواند بزرگ در نظر گرفته شود چون باعث می‌شود الگوریتم زمان زیادی را برای رسیدن به پاسخ بهینه صرف کند و همچنین نمی‌تواند خیلی کوچک در نظرگرفته شود تا تنوع الگوی انتخابی اولیه کم نباشد. پس این مقدار دست برنامه‌نویس بوده و عموما سعی می‌شود بنابر مقدار زمانی که برای محاسبه مقدار برازندگی هر الگو صرف می‌شود این تعداد انتخاب شود. ما از این به بعد این مقدار را بزرگی توده[1] می‌نامیم.

[1] Swarm Size

مرحله ۲: به ترتیب با استفاده از هر الگوی انتخابی ویژگی‌های ورودی را کاهش داده و سپس با استفاده از تابع برازندگی که در نظر گرفته‌ایم مقدار شایستگی این الگو را بدست می‌آوریم. تابع برازندگی مورد استفاده در اینجا طبقه‌بند ماشین بردار پشتیبان (SVM) می‌باشد. این مقدار طبق معادله (۵-۸) که برای آن در نظر گرفتیم از ترکیب وزنی تعداد ویژگی‌های انتخاب شده نسبت به طول بردار ویژگی اصلی و مقدار بازشناسی که طبقه‌بند مورد استفاده برای هر الگو بدست می‌آورد محاسبه می‌شود. سپس بعد از اینکه برای کل الگوها مقدار برازندگی بدست آمد بایستی برای هر الگو i طبق معادله (۵-۹) مقدار احتمال شایستگی در بین کل j الگو بدست آید.

$$Fitness(i) = \frac{K_1}{|F|} + K_2 \times Accuracy \qquad (۵-۸)$$

$$P(h_i) = \frac{fittness(h_i)}{\sum_j fittness(h_j)} \qquad (۵-۹)$$

در رابطه (۵-۸) مقدار K_1 و K_2 اعداد ثابت اختیاری و $|F|$ برابر تعداد ویژگی‌های انتخابی است. بر اساس رابطه (۵-۹) فوق احتمال انتخاب فرضیه i از میان کل فرضیه‌ها محاسبه می‌شود.

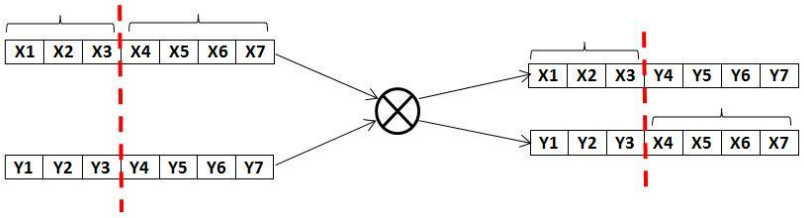
مرحله ۳: حال در این قسمت بایستی جمعیت را بر اساس مقدار شایستگی و احتمال بدست آمده برای هر الگو به روزرسانی نماییم. ابتدا کارکرد اپراتورهای Crossover و Mutation را بطور خلاصه بیان می‌نماییم. سپس به چگونگی استفاده از این اپراتورها در به روز رسانی الگوها می‌پردازیم.

۵-۲-۱- اپراتور Crossover

این اپراتور با استفاده از ۲ والد۲ فرزند را تولید می‌کند. که برای این کار قسمتی از بیت‌های والدین در فرزند کپی می‌شود. انتخاب این بیت‌ها که از والدین به فرزند منتقل می‌شود به ۳ روش صورت می‌گیرد که در ادامه آورده شده است.

۵-۲-۱-۱- روش اول Single-Point Crossover:

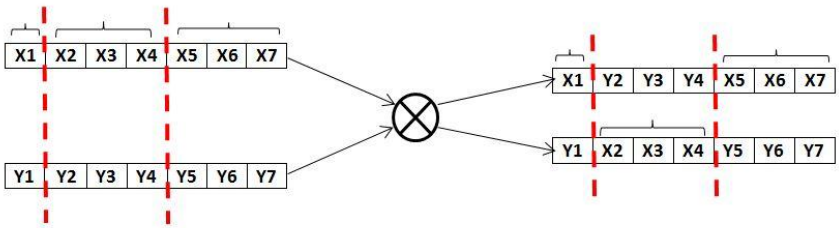
در این روش ما جفت جفت الگوها را انتخاب کرده و به عنوان والدین استفاده می‌نماییم. سپس یک نقطه تصادفی اما برابر برای هر دو رشته در طول رشته والدین انتخاب نموده و سپس رشته والدین را در این نقطه به ۲ رشته تبدیل می‌نماییم. تولید رشته فرزندان در این روش این گونه است که هر فرزند از ترکیب تکه اول از والد اول و تکه دوم از والد دوم بدست می‌آید. در **شکل ۵-۳** مراحل این روش نشان داده شده است.



شکل ۵-۳: مراحل روش Single-Point Crossover

۵-۲-۱-۲- روش دوم Two-Point Crossover:

در این روش نیز مانند قبل عمل می‌کنیم اما فرقی که با روش اول دارد این است که بجای انتخاب ۱ نقطه، در این روش ۲ نقطه را برای تولید فرزندان انتخاب می‌نماییم. این عمل نسبت به حالت اول بهتر بوده و تغییرات در نسلها آهسته‌تر و نرم‌تر صورت می‌پذیرد که نسبت به قبل مناسب‌تر می‌باشد. در **شکل ۵-۴** مراحل این روش نشان داده شده است.



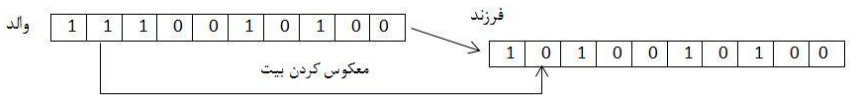
شکل ۵-۴: مراحل روش Two-Point Crossover

۵-۲-۱-۳- روش سوم Uniform Crossover:

در این روش چند نقطه را در طول رشته والدین انتخاب نموده و سپس عمل تولید فرزندان نسل بعد صورت می‌پذیرد. برای این عمل در محیط نرم‌افزار متلب می‌توانید چند ماسک ثابت[1] برای خود معرفی کنید و با استفاده از این ماسک‌ها عملیات انتقال و تولید را انجام دهید.

۵-۲-۲- اپراتور Mutation

این اپراتور برای ایجاد فرزند فقط از یک والد استفاده می‌کند که با ایجاد تغییرات کوچکی در رشته اولیه نسل بعد را ایجاد می‌کند. در این روش با استفاده از یک توزیع یکنواخت یک بیت به صورت تصادفی انتخاب و مقدار آن را معکوس می‌نماید. روش کار در شکل **۵-۵** نشان داده شده است.



شکل ۵-۵: مراحل روش Mutation

اگر از هر دو اپراتور در تولید نسل جدید استفاده شود بایستی اپراتور Mutation را بعد از اعمال اپراتور Crossover اعمال نماییم. حال سوال اینجاست که از کدام اپراتور استفاده شود؟ در حالت کل بهتر است از هر دو اپراتور استفاده نماییم و هر کدام نقش مخصوص به خود را دارند. همچنین می‌توان الگوریتمی داشت که فقط از Mutation استفاده کرده باشد اما الگوریتمی که فقط از Crossover استفاده کرده باشد کار نمی‌کند چون در اپراتور Crossover پرش‌ها و تغییرات بزرگ می‌باشند خصوصاً زمانی که طول بردار ویژگی ما بزرگ باشد. اما Crossover خاصیت جستجو گرانه[2] دارد که می‌تواند با پرش‌های بزرگ نواحی جدیدی را کشف نماید. اپراتور Mutation خاصیت گسترشی[3] دارد که با اعمال تغییرات کوچک تصادفی به نواحی کشف شده وسعت می‌بخشد. از همه مهمتر اپراتور Crossover اطلاعات والدین را ترکیب نموده در صورتی که اپراتور Mutation می‌تواند اطلاعات جدیدی را در جهت بهبود نسل جدید اضافه نماید.

[1] Crossover Mask
[2] Explorative
[3] Exploitive

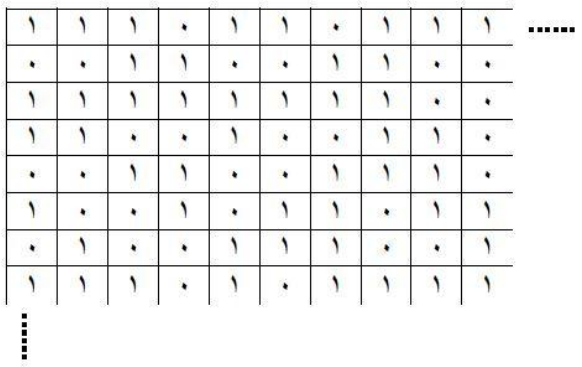
۵-۲-۳- انتخاب جمعیت

در ادامه بعد از اینکه مقدار شایستگی جمعیت اولیه بدست آمد بایستی نحوه تولید نسل در جمعیتی که مقدار شایستگی بهتری دارد رااز جمعیتی که مقدار شایستگی کمتر دارد جدا نمود. برای این عمل بعد از محاسبه مقدار برازندگی برای هرالگو مقدار احتمال یک الگو برای شرکت در نسل بعدی را بسته به مقدار شایستگی بدست آمده برای آن الگو بدست می‌آوریم. ما در پیاده‌سازی خود فقط از اپراتور Mutation و در قسمت انتخاب جمعیت نیز از چرخ رولت استفاده نموده‌ایم. روش کار در چرخ رولت این گونه است که پس از محاسبه مقادیر شایستگی مقدار احتمال هر فرضیه یا الگو را به استفاده از رابطه (۵-۹) بدست می‌آوریم. سپس بر اساس مقدار احتمال الگوها را به صورت نزولی از چپ به راست قرار می‌دهیم و α درصد الگوهایی که مقدار شایستگی بالاتری نسبت با سایر الگوها دارند را به یک صورت به نسل بعد انتقال می‌دهیم و سایر الگوهای باقی مانده را با جهشی بیشتراز سایر الگوها به نسل بعد می‌فرستیم. دلیل این امر واضح است الگویی که به مقدار ماکزیمم شایستگی محلی رسیده برای اینکه بتواند بهتر الگوی بهینه سراسری را بیابد بایستی حرکت آهسته‌تری نسبت به الگویی‌که ازاین مقدار فاصله دارد داشته باشد. لازم به توضیح است که مقدارα دست کاربربوده وهمچنین تعداد بیتی که قراراست در این α درصد تغییر کند نیز دست کاربر می‌باشد. پس تغییرات تعداد بیتها دردو ناحیه‌ای که بر اساس مقدار احتمال بدست آمده از شایستگی بدست‌آمد متفاوت و قابل کنترل می‌باشد[20].

در ادامه بعد از بروزرسانی الگوهای انتخابی بنابر توضیحات داده شده بار دیگر مقدار شایستگی هر الگو را بدست آورده و این مراحل را تکرار می‌نماییم. شرط خاتمه نیز می‌تواند مقدار تکرار مشخص یا مقدار خطا و مقدار شایستگی فرزند تولیدی باشد که دست کاربر و قابل تغییر می‌باشد. در نهایت بهترین الگوی انتخابی بنابر تابع شایستگی که ما تعیین نموده بودیم بدست آمده و با انتخاب ویژگی بر اساس این الگو به پاسخ بهینه شده دست می‌یابیم.

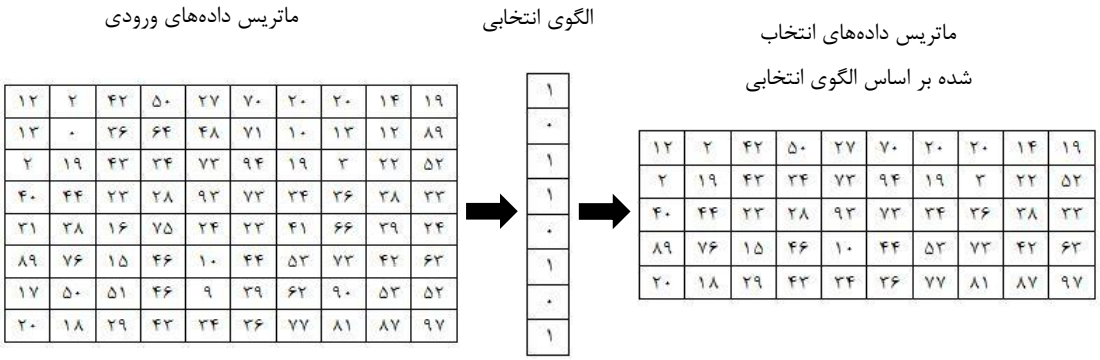
۵-۳- انتخاب ویژگی با استفاده از الگوریتم باینری PSO

در این قسمت مراحل انتخاب ویژگی با استفاده از الگوریتم بهینه سازی توده ذرات باینری شرح داده می‌شود. در این روش ابتدا مقادیر اولیه PSO داده شده و همانند مرحله اول در الگوریتم GA ماتریس تصادفی به ابعاد شرح داده شده، ایجاد و از آن به عنوان جمعیت اولیه در الگوریتم PSO استفاده می‌شود. در این ماتریس هر ستون به عنوان یک الگو انتخاب ویژگی مورد استفاده قرار می‌گیرد.



شکل ۵-۶: ماتریس تصادفی برای انتخاب ویژگی

تعداد سطرهای این ماتریس برابر تعداد ویژگی‌های داده ورودی است و تعداد ستونها برابر بزرگی ذره یا همان (Swarm Size) می‌باشد که مقدار آن قابل تغییر می‌باشد و بنابر اندازه داده ورودی تغییر می-نماید. مراحل کار در این روش این گونه است: مانند روش قبل با استفاده از هر الگو کل داده‌های آموزش ورودی انتخاب و سپس مقدار برازندگی برای هر الگو بدست می‌آید یعنی اگر مقدار Swarm Size=100 باشد طول بردار برازندگی ما نیز برابر ۱۰۰ می‌باشد در ادامه با استفاده از یک ماتریس پیش فرض برای داده ورودی این مراحل را تشریح می‌نماییم.



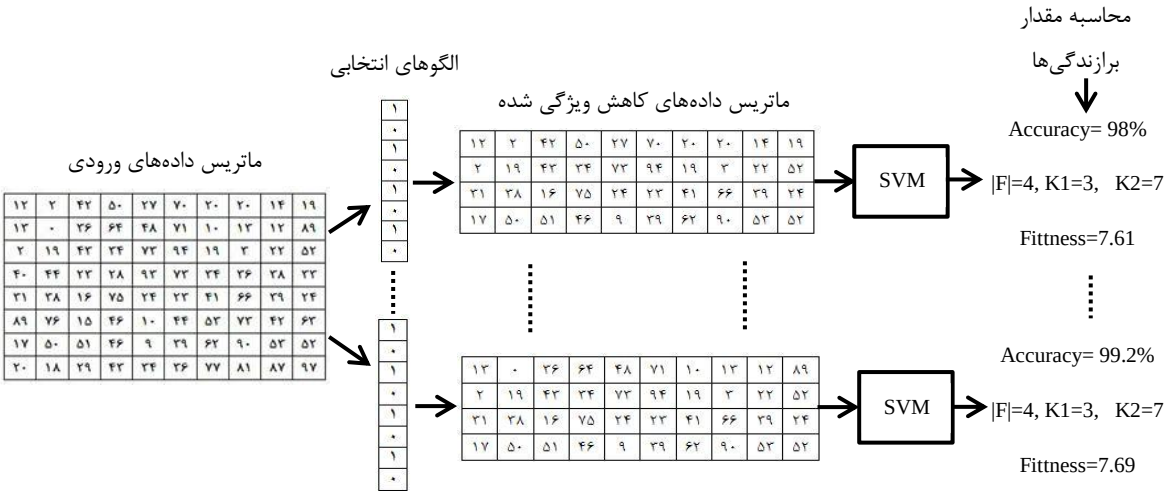
شکل ۵-۷: نحوه انتخاب ویژگی بر اساس الگو مورد نظر

پس از بدست آوردن مقدار برازندگی برای هر الگو مقدار بهترین شخصی P_best و بهترین سراسری G_best را بدست می‌آوریم و در حالت دقیقتر از L_best استفاده می‌نماییم و سپس ماتریس اولیه سرعت را نیز که به اندازه ماتریس جمعیت اولیه می‌باشد ایجاد نموده در این مورد ما مقدار ماکزیمم و مینیمم ماتریس را به بازه [۱,۱-] محدود نموده‌ایم.

۵-۴- تابع برازندگی

از تابع برازندگی در الگوریتم‌های بهینه سازی برای هدایت کل الگوریتم به پاسخ بهینه کل مورد استفاده قرار می‌گیرد. برای این مورد از توابع مختلفی تا به امروز استفاده شده است از فاصله اقلیدوسی ویژگی‌ها تا مقدار خروجی تابعی که برای آن مد نظر گرفته شده است. ما در این قسمت از طبقه‌بند ماشین بردار پشتیبان SVM کتابخانه LIBSVM برای محاسبه تابع برازندگی استفاده نموده‌ایم. بدین گونه که برای هر الگوی انتخابی با استفاده از طبقه‌بند SVM مقدار بازشناسی محاسبه شد و سپس ترکیبی از مقدار نرخ بازشناسی و تعداد ویژگی را بر اساس رابطه (۵-۸) برای مقدار برازندگی استفاده نمودیم.

در این رابطه همان گونه که قبلا گفتیم مقدار $|F|$ برابر تعداد ویژگی‌های انتخابی و در ضمن K_1 و K_2 اعداد ثابت می‌باشند. مقادیر K_1 و K_2 بر اساس اولویتی که ما برای نیاز به کاهش ویژگی یا نرخ بازشناسی بالا داریم انتخاب می‌شود. در شکل **۵-۸** مراحل محاسبه مقدار برازندگی نشان داده شده است.



شکل ۵-۸: مراحل محاسبه مقدار برازندگی بر اساس الگوها

همچنین در این مرحله پارامترهای طبقه‌بند SVM مربوط به هر الگو نیز ذخیره می‌شود. در ادامه الگویی که بهترین مقدار برازندگی را کسب کرده را به عنوان الگوی بهینه سراسری G_best و کل الگوها را به عنوان بهینه شخصی P_best برداشته و برای تکرار بعدی مورد استفاده قرار می‌گیرند. در تکرار بعد مقدار P_best بدست آمده در تکرار قبلی به عنوان جمعیت اولیه یا در اینجا الگوی انتخابی اولیه مورد استفاده قرار می‌گیرد. سپس الگوها بر اساس معادلات (۳-۲) و (۳-۳) به روزرسانی شده و در این مرحله کل الگوها بر اساس سرعت و پارامترهای اینرسی و C_1 و C_2 انتخابی به سمت الگوی بهتر یعنی G_best حرکت می‌کنند. همان گونه که بیان شد ما برای انتخاب ویژگی ازالگوریتم BPSO استفاده نمودیم در این الگوریتم اساس کار به این صورت است که پس از به روزرسانی سرعت قبلی مقدار سرعت را با استفاده از تابع سیگموید بین ۰و۱ محدود نموده و سپس جمعیتی بر اساس این سرعت به روز شده با درایه‌های صفر و یک ایجاد می‌نماییم. اساس کار در این روش بر پایه ماتریس با درایه‌های ۰و۱ است اما در PSO پایه این گونه نمی‌باشد. در هر تکرار پس از بدست آوردن P_best و مقدار برازندگی، مقدار برازندگی جدید را با برازندگی قبلی مقایسه می‌نماییم برای هر درایه از بردار برازندگی اگر برازندگی بدست آمده بزرگتر از درایه متناظر قبلی خود باشد الگوی مربوط به آن درایه رااز ماتریس الگوی P_best جدید انتخاب نموده و باالگوی متناظر خود در P_best قبلی جایگزین می‌نماییم. همچنین پارامتر SVM جدید متناظر با آدرس این الگو را نیز انتخاب و با پارامتر متناظرش تعویض می‌نماییم. این مراحل تا زمانی که شرط توقف برقرار

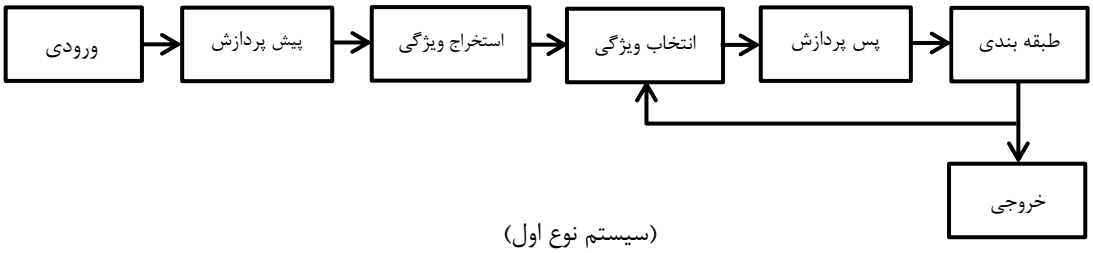
نشده ادامه می‌یابد. قابل توجه است که امکان دارد الگوریتم ویژگی‌ها را بیش از حد مورد نیاز کاهش دهد وکاهش ما قابل توجهی در نرخ بازشناسی داشته باشیم که بایستی با انتخاب دقیق مقادیر K_1 وK_2 در تابع برازندگی و شرط توقف به این مورد توجه کنیم. در نهایت پس از پایان الگوریتم و بدست آوردن بهترین الگوی انتخابی از داده‌های ورودی، در مرحله آزمایش داده‌ها را با این الگو انتخاب نموده و سپس عمل بازشناسی را با ویژگی‌های کاهش یافته بدست می‌آوریم.

فصل ۶

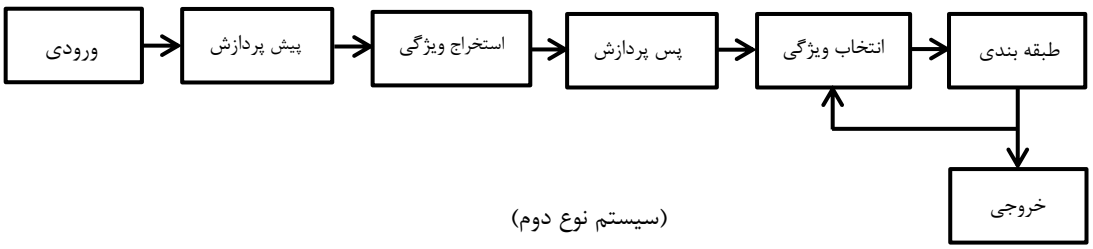
نتایج شبیه‌سازی و بررسی کلی

۶-۱- سیستم پیشنهادی جهت بازشناسی الگو

شناسایی الگو شاخه‌ای از هوش مصنوعی است که با طبقه‌بندی و توصیف مشاهدات سروکار دارد. شناسایی الگو به ما کمک می‌کند داده‌ها (الگوها) را با تکیه بر دانش قبلی یا اطلاعات آماری استخراج شده از الگوها، طبقه‌بندی نماییم. الگوهایی که می‌بایست طبقه‌بندی شوند، معمولا گروهی از سنجش‌ها یا مشاهدات هستند که مجموعه نقاطی را در یک فضای چند بعدی مناسب تعریف می‌نمایند. یک سیستم شناسایی الگو کامل متشکل است ازیک حسگر، که مشاهداتی را که می‌بایست توصیف یا طبقه-بندی شوند جمع‌آوری می‌نماید، یک سازوکار برای استخراج ویژگی‌ها که اطلاعات عددی یا نمادین را از مشاهدات، محاسبه می‌کند(این اطلاعات عددی را با یک بردار به نام بردار ویژگی نمایش می‌دهند)، و یک نظام طبقه‌بندی یا توصیف که وظیفه اصلی طبقه‌بندی یا توصیف الگوها را با تکیه بر ویژگی‌های استخراج شده عهده‌دار است. شکل **۶-۱** نمودار بلوکی نمونه‌ای از یک سیستم بازشناسی الگو را نشان می‌دهد[۵۱]. در اینجا قصد داریم سیستمی را برای سهولت پیگیری عمل بازشناسی پیشنهاد نماییم تا خواننده به راحتی بتواند بر اساس این سیستم الگوریتم پیاده سازی شده را دنبال و عیب‌یابی نماید.



(سیستم نوع اول)



(سیستم نوع دوم)

شکل ۶-۱: انواع سیستم های بازشناسی الگو استاندارد

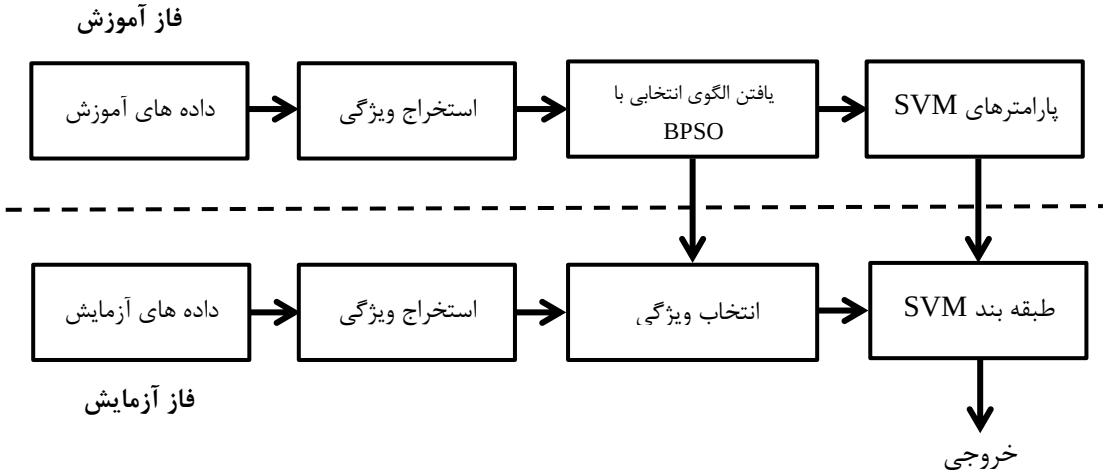
که هر کدام از دو نوع نشان داده شده به نوبه خود مزیتهایی نسبت به دیگری دارند. در این پایان‌نامه ما سیستم بازشناسی را به دو فاز تقسیم نموده‌ایم: فاز آموزش و فاز آزمایش. در فاز آموزش از سیستم نوع دوم و به جهت اینکه در فاز آزمایش الگوی انتخابی در دست است ابتدا انتخاب ویژگی صورت گرفته سپس پس پردازش اعمال می شود یعنی از سیستم نوع اول استفاده شده است.

۶-۱-۱- پیش پردازش:

کلیه اعمالی که روی تصویر انجام می‌شود تا موجب تسهیل در روند اجرای فازهای بعدی گردد، مانند دودویی کردن تصویر، حذف نویز، هموارسازی، نازک سازی، تشخیص زبان و فونت کلمات و نظیر اینها و از مجموعه این پردازش ها اهداف زیر بدست می‌آید[۵۲].

کاهش نویز، نرمال کردن داده ها و فشرده سازی میزان اطلاعاتی که می‌بایست محفوظ بماند

در شکل **۶-۲** سیستم پیشنهادی استفاده شده برای این منظور نشان داده شده است.



شکل ۶-۲: سیستم پیشنهادی جهت بازشناسی الگو

۶-۱-۲- فاز آموزش:

در این فاز ما با استفاده از داده های مخصوص آموزش و الگوریتم BPSO پیشنهاد شده بهترین الگوی انتخاب ویژگی را پیدا می‌نماییم و همچنین در این فاز ما پارامترهای طبقه بند SVM مخصوص بهترین

الگوی انتخابی را ذخیره کرده تا در فاز آزمایش عمل طبقه بندی را بر اساس این پارامترها انجام دهیم.

۶-۱-۳- فاز آزمایش:

در این فاز ما داده های تست را بر اساس الگوی بدست آمده در فاز آموزش انتخاب نموده و با پارامتر ذخیره شده برای طبقه بند SVM عمل بازشناسی نهایی صورت می‌گیرد.

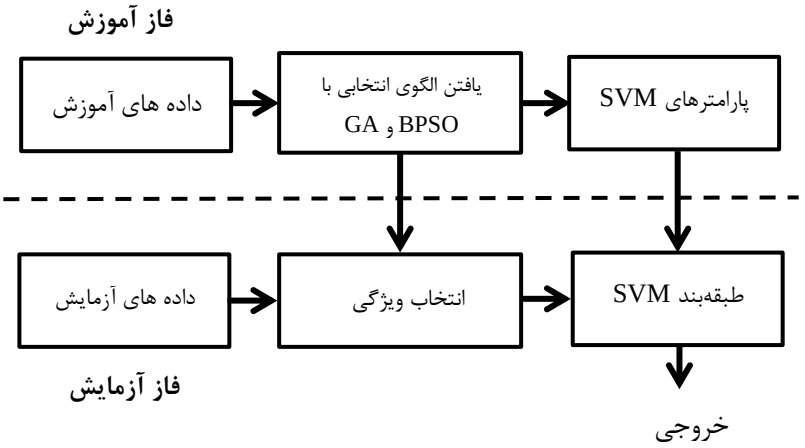
۶-۱-۴- کاربردهای بازشناسی الگو:

بازشناسی الگو در بسیاری از زمینه‌ها نقش کاربردی دارد[۵۳]. بازشناسی حروف، بازشناسی نویسنده، تصدیق امضاء، طبقه‌بندی اثر انگشت، چهره، ارقام و بازشناسی گفتار و ... نمونه‌هایی از این کاربردها هستند. در این پایان‌نامه از دیتا بیس‌هایی از پایگاه داده UCI که به صورت فایل text. می‌باشند و همچنین از داده ارقام دستنویس فارسی هدی و داده‌های چهره پایگاه داده ORL برای پیاده‌سازی استفاده نموده‌ایم که در ادامه به آنها می‌پردازیم.

۶-۲- بازشناسی داده های پایگاه داده UCI

۶-۲-۱-سیستم پیشنهادی برای بازشناسی داده های پایگاه UCI

داده‌های مربوط به این پایگاه UCI از دو کلاسه گرفته تا چند کلاسه موجود می‌باشند. با استفاده از الگوریتم های بحث شده جهت انتخاب ویژگی با BPSO و GA عمل کاهش ویژگی صورت گرفت و چون ویژگی داده های این پایگاه به صورت استخراج شده در دسترس می باشند، دیگر نیازی به استخراج ویژگی نبوده، سیستم نشان داده در شکل ۶-۳ برای عمل بازشناسی به صورت زیر تغییر می‌نماید.



شکل ۶-۳: سیستم پیشنهادی جهت بازشناسی داده های پایگاه UCI

در این پایگاه داده‌های مختلفی در ابعاد و حوزه‌های متفاوت وجود دارد ما چند نمونه پرکاربرد از این داده ها را انتخاب نموده‌ایم و الگوریتم پیشنهاد شده را بر روی آنها پیاده نموده‌ایم و نتایج بدست آمده برای تک تک داده‌ها به ترتیب آورده شده است. در جدول (۶-۱) مشخصات کامل داده ها شرح داده شده است. مقادیر زمان آورده شده در جدول نتایج هر داده برای مقایسه الگوریتم GA و PSO می‌باشد. در تمام جوابها از نرم افزار MATLAB با مشخصات (Version 7.11.0.584(R2010b -64bit(win64) و سیستمی با مشخصات (64-bit)-intel Core(TM)i3 CPU-330M @2.13GHz 2.13GHz استفاده شده است.

جدول ۶-۱: مشخصات داده های پایگاه داده UCI

نام دیتا بیس	تعداد ویژگی‌ها	تعداد نمونه‌ها	تعداد کلاس‌ها
Glass	۱۰	۲۱۴	۶
WDBC	۳۰	۵۶۹	۲
Sonar	۶۰	۲۰۸	۲
Wine	۱۳	۱۷۸	۳
Pen digits	۱۶	۱۰۹۹۲	۱۰
Ionosphere	۳۲	۳۵۱	۲
Image Segmentation	۱۹	۲۱۰	۷
Spect	۲۲	۲۶۷	۲
Australian	۱۴	۶۹۰	۲

۶-۲-۲- نتایج بدست آمده برای داده های UCI

نتایج بدست آمده بر روی داده‌های پایگاه داده UCI که در جدول ۶-۱ آمده است به ترتیب بر اساس ترتیب جدول ۶-۱ در زیر از جدول ۶-۴ تا جدول ۶-۲۱ آورده شده است. با استفاده از سیستم پیشنهادی و الگوریتم GA و BPSO به ترتیب نتایج بررسی شده است. برای همه جداول در این قسمت آخرین ستون بازشناسی بدون کاهش ویژگی و سایر ستونها بازشناسی با کاهش ویژگی است. همچنین منحنی تغییرات نرخ بازشناسی نسبت به تعداد ویژگی برای الگوریتم BPSO و GA برای هر داده در ادامه جدول بازشناسی مربوط به آن داده آمده است. در این بررسی پارامتر های مربوط به الگوریتم بهینه سازی توده ذرات بر اساس جدول ۶-۲ می باشد.

جدول ۶-۲: پارامتر های انتخاب شده برای الگوریتم PSO

تعداد ذرات	۱۰۰
ضریب c_1 و c_2	۲
ضریب K_1	۳
ضریب K_2	۶
تعداد تکرار	۵۰

همچنین در بررسی این داده ها با استفاده از الگوریتم بهینه سازی توده ذرات و الگوریتم ژنتیک شرط توقف را تعداد تکرار در نظر گرفته ایم وپارامتر های الگوریتم ژنتیک را نیز مانند جدول ۶-۳ در نظرگرفته ایم. در این آزمایش با استفاده از تابع crossvalind موجود در نرم افزار Matlab کل داده ها را ۱۰ برابر نمودیم و ۳۰ درصد کل را برای آزمایش و ۷۰ درصد باقی مانده را برای آموزش استفاده نمودیم.

جدول ۶-۳: پارامتر های انتخاب شده برای الگوریتم GA

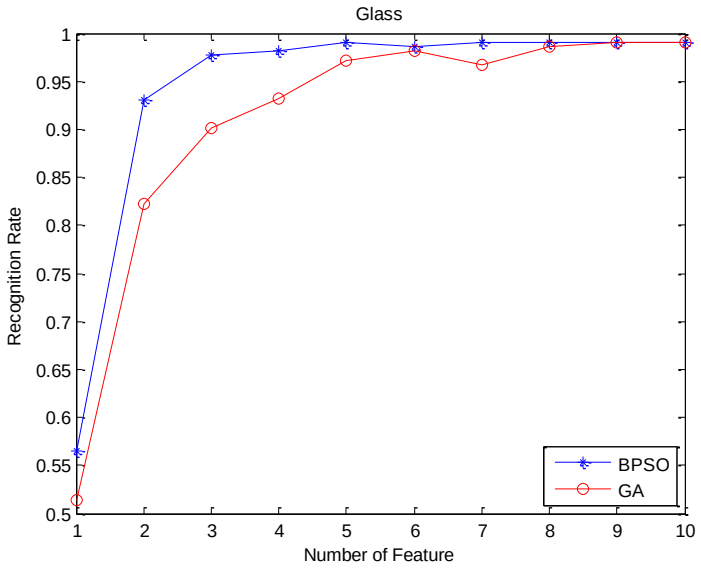
تعداد ذرات	۱۰۰
ضریب α	٪۱۰
ضریب K_1	۳
ضریب K_2	۶
تعداد تکرار	۵۰

جدول ۶-۴: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از GA+SVM بر روی داده Glass

تعداد ویژگی	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰
نرخ بازشناسی	٪۵۱,۴۰	٪۸۲,۲۴	٪۹۰,۱۹	٪۹۳,۱۸	٪۹۷,۲۰	٪۹۸,۱۳	٪۹۶۷۳	٪۹۸,۶۰	٪۹۹,۰۷	٪۹۹,۰۷
زمان	۰,۰۲۰۵	۰,۰۲۶۱	۰,۰۲۹۰	۰,۰۳۱۲	۰,۰۲۱۱	۰,۰۲۳۰	۰,۰۲۳۸	۰,۰۲۵۰	۰,۰۲۴۳	۰,۰۲۹۴

جدول ۶-۵: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از BPSO+SVM بر روی داده Glass

تعداد ویژگی	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰
نرخ بازشناسی	٪۷۵,۷۰	٪۹۲,۹۹	٪۹۷,۶۶	٪۹۸,۱۳	٪۹۹,۰۷	٪۹۹,۰۷	٪۹۹,۰۷	٪۹۹,۰۷	٪۹۹,۰۷	٪۹۹,۰۷
زمان	۰,۰۱۵۲	۰,۰۱۷۱	۰,۰۲۱۸	۰,۰۲۲۸	۰,۰۲۰۲	۰,۰۲۱۰	۰,۰۲۹۲	۰,۰۲۷۱	۰,۰۲۸۱	۰,۰۲۹۱



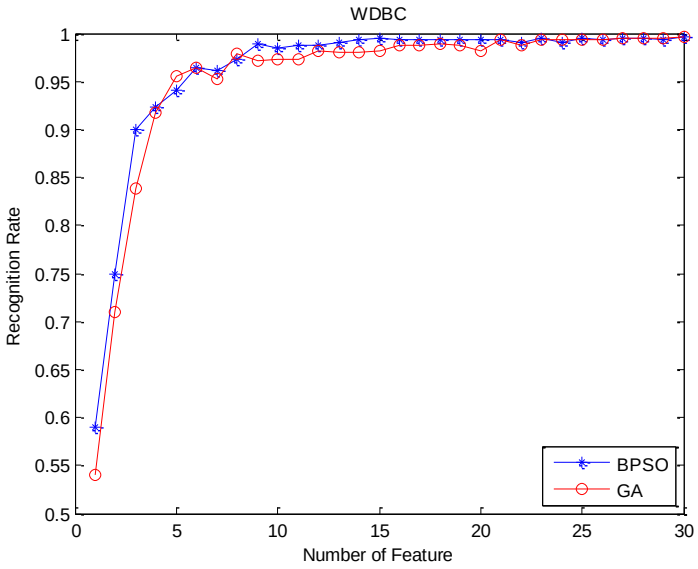
شکل ۶-۴: تغییرات نرخ بازشناسی بر اساس تعداد ویژگی های مختلف در داده Glass

جدول ۶-۶: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از GA+SVM بر روی داده WDBC

تعداد ویژگی	۴	۵	۷	۹	۱۲	۱۴	۱۶	۲۰	۲۵	۳۰
نرخ بازشناسی	٪۹۱,۷۴	٪۹۵,۶۱	٪۹۵,۲۵	٪۹۷,۱۹	٪۹۸,۲۴	٪۹۸,۰۷	٪۹۸,۰۷	٪۹۸,۲۴	٪۹۹,۳۰	٪۹۹,۶۵
زمان	۰,۰۴۴۶	۰,۰۴۷۸	۰,۰۵۰۷	۰,۰۵۶۹	۰,۰۵۲۲	۰,۰۶۰۰	۰,۰۶۰۴	۰,۰۷۰۲	۰,۰۷۳۳	۰,۰۸۶۴

جدول ۶-۷: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از BPSO+SVM بر روی داده WDBC

تعداد ویژگی	۴	۵	۷	۹	۱۲	۱۴	۱۶	۲۰	۲۵	۳۰
نرخ بازشناسی	٪۹۲,۲۷	٪۹۴,۰۲	٪۹۶,۱۳	٪۹۸,۹۵	٪۹۸,۸۷	٪۹۹,۳۰	٪۹۹,۳۰	٪۹۹,۳۰	٪۹۹,۴۷	٪۹۹,۶۵
زمان	۰,۰۴۳۳	۰,۰۴۹۲	۰,۰۵۰۹	۰,۰۴۹۸	۰,۰۵۴۳	۰,۰۵۵۲	۰,۰۵۶۱	۰,۰۶۱۳	۰,۰۶۷۷	۰,۰۷۹۰



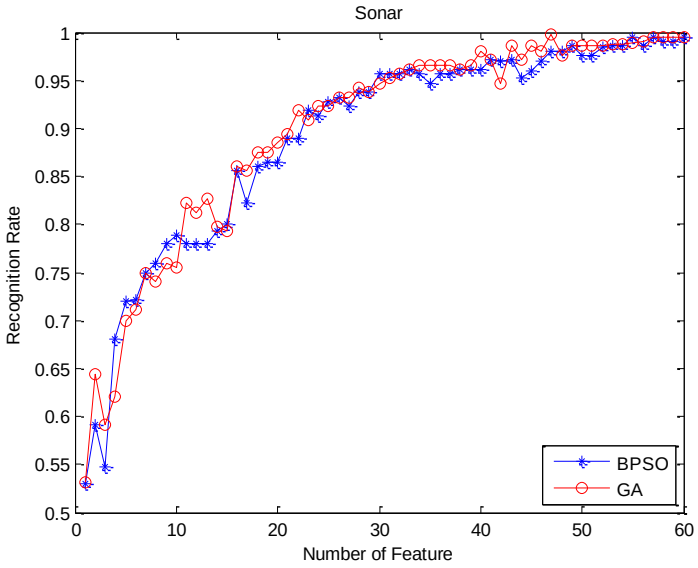
شکل ۶-۵: تغییرات نرخ بازشناسی بر اساس تعداد ویژگی های مختلف در داده WDBC

جدول ۶-۸: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از GA+SVM بر روی داده Sonar

تعداد ویژگی	۶	۱۰	۱۹	۲۶	۳۱	۳۸	۴۳	۴۹	۵۷	۶۰
نرخ بازشناسی	٪۷۱,۱۵	٪۷۵,۴۸	٪۸۷,۵۰	٪۹۳,۲۷	٪۹۵,۱۹	٪۹۶,۱۵	٪۹۸,۵۶	٪۹۸,۵۶	٪۹۹,۵۲	٪۹۹,۵۲
زمان	۰,۰۳۵۲	۰,۰۳۹۱	۰,۰۴۹۱	۰,۰۵۴۰	۰,۰۶۲۵	۰,۰۶۶۸	۰,۰۶۹۱	۰,۰۶۹۳	۰,۰۷۱۱	۰,۰۷۸۹

جدول ۶-۹: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از BPSO+SVM بر روی داده Sonar

تعداد ویژگی	۶	۱۰	۱۹	۲۶	۳۱	۳۸	۴۳	۴۹	۵۷	۶۰
نرخ بازشناسی	٪۷۲,۱۲	٪۷۸,۸۵	٪۸۶,۵۴	٪۹۳,۲۷	٪۹۵,۶۷	٪۹۶,۱۵	٪۹۷,۱۲	٪۹۸,۵۶	٪۹۹,۵۲	٪۹۹,۵۲
زمان	۰,۰۳۰۶	۰,۰۳۵۰	۰,۰۴۹۶	۰,۰۵۳۳	۰,۰۵۴۲	۰,۰۵۸۸	۰,۰۵۹۹	۰,۰۶۶۱	۰,۰۷۲۲	۰,۰۸۰۱



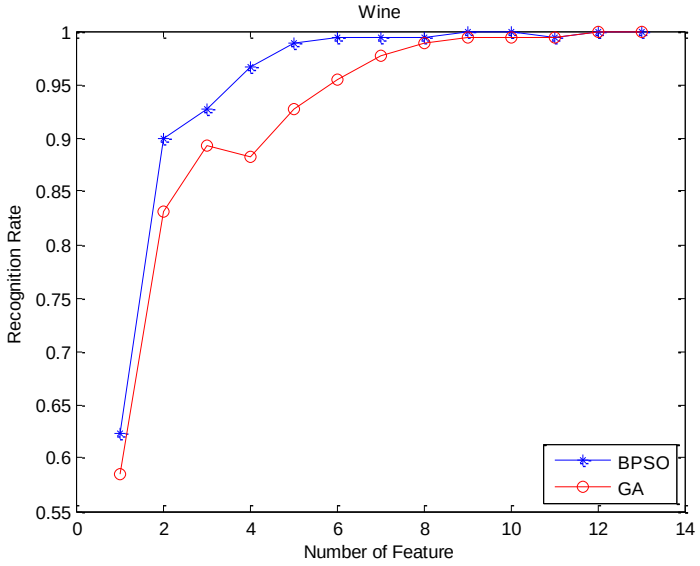
شکل ۶-۶: تغییرات نرخ بازشناسی بر اساس تعداد ویژگی های مختلف در داده Sonar

جدول ۶-۱۰: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از GA+SVM بر روی داده Wine

تعداد ویژگی	۱	۲	۳	۴	۵	۶	۸	۱۰	۱۲	۱۳
نرخ بازشناسی	٪۵۸,۴۳	٪۸۳,۱۵	٪۸۹,۳۳	٪۸۸,۲۰	٪۹۲,۷۰	٪۹۵,۵۱	٪۹۸,۸۸	٪۹۹,۴۴	٪۱۰۰	٪۱۰۰
زمان	۰,۰۱۲۸	۰,۰۱۶۳	۰,۰۱۷۸	۰,۰۲۵۳	۰,۰۲۷۵	۰,۰۲۹۹	۰,۰۳۰۴	۰,۰۳۳۷	۰,۰۳۷۶	۰,۰۳۷۸

جدول ۶-۱۱: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از BPSO+SVM بر روی داده Wine

تعداد ویژگی	۱	۲	۳	۴	۵	۶	۸	۱۰	۱۲	۱۳
نرخ بازشناسی	٪۶۲,۳۶	٪۸۹,۸۹	٪۹۲,۷۰	٪۹۶,۶۳	٪۹۸,۸۸	٪۹۹,۴۴	٪۹۹,۴۴	٪۱۰۰	٪۱۰۰	٪۱۰۰
زمان	۰,۰۱۰۴	۰,۰۱۳۹	۰,۰۱۷۸	۰,۰۲۴۹	۰,۰۲۷۶	۰,۰۳۰۴	۰,۰۳۳۹	۰,۰۳۴۹	۰,۰۳۵۳	۰,۰۳۴۷



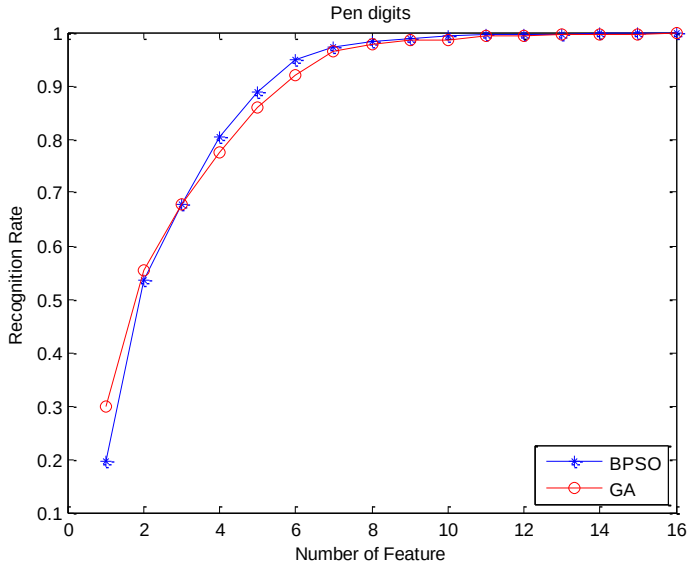
شکل ۶-۷: تغییرات نرخ بازشناسی بر اساس تعداد ویژگی های مختلف در داده Wine

جدول ۶-۱۲: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از GA+SVM بر روی داده Pen digits

تعداد ویژگی	۲	۳	۴	۵	۷	۹	۱۰	۱۲	۱۳	۱۶
نرخ بازشناسی	٪۵۵,۴۳	٪۶۷,۷۶	٪۷۷,۳۷	٪۸۵,۹۹	٪۹۶,۵۱	٪۹۸,۵۰	٪۹۸,۵۰	٪۹۹,۴۰	٪۹۹,۷۰	٪۹۹,۸۲
زمان	۲۳,۷۷	۲۴,۵۶۷	۳۴,۶۳۴	۲۱,۳۷۰	۱۱,۲۵۳	۸,۹۱۷۴	۸,۳۸۸۳	۶,۲۷۵۷	۵,۵۷۴۳	۵,۱۰۳۳

جدول ۶-۱۳: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از BPSO+SVM بر روی داده Pen digits

تعداد ویژگی	۲	۳	۴	۵	۷	۹	۱۰	۱۲	۱۳	۱۶
نرخ بازشناسی	٪۵۳,۵۵	٪۶۷,۷۳	٪۸۰,۳۹	٪۸۸,۸۹	٪۹۷,۲۲	٪۹۸,۹۴	٪۹۹,۲۸	٪۹۹,۶۵	٪۹۹,۷۶	٪۹۹,۸۲
زمان	۴۱,۳۱	۳۸,۳۴	۳۰,۳۷	۳۰,۵۴	۱۴,۸۳	۸,۳۸۵	۶,۷۵۵	۷,۵۶۸	۵,۴۳۱	۴,۳۸۵۴



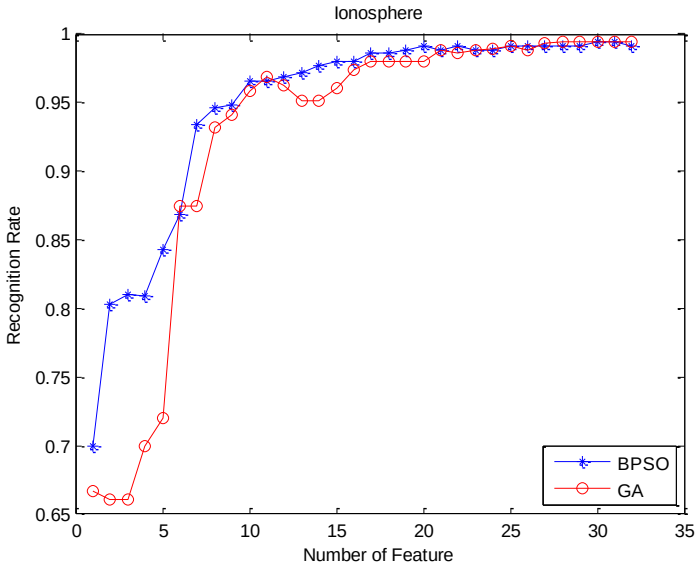
شکل ۶-۸: تغییرات نرخ بازشناسی بر اساس تعداد ویژگی های مختلف در داده Pen digits

جدول ۶-۱۴: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از GA+SVM بر روی داده Ionosphere

تعداد ویژگی	۲	۵	۷	۸	۱۰	۱۴	۱۷	۲۲	۲۶	۳۲
نرخ بازشناسی	٪۶۶,۱۰	٪۷۲,۰۱	٪۸۷,۴۶	٪۹۳,۱۶	٪۹۵,۸۷	٪۹۵,۱۶	٪۹۸,۰۱	٪۹۸,۵۸	٪۹۸,۸۶	٪۹۹,۴۳
زمان	۰,۰۷۹۱	۰,۰۸۲۰	۰,۰۹۲۸	۰,۰۶۸۹	۰,۰۴۸۰	۰,۰۴۷۹	۰,۰۵۴۴	۰,۰۶۱۹	۰,۰۴۷۱	۰,۰۳۶۳

جدول ۶-۱۵: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از BPSO+SVM بر روی داده Ionosphere

تعداد ویژگی	۲	۵	۷	۸	۱۰	۱۴	۱۷	۲۲	۲۶	۳۲
نرخ بازشناسی	٪۸۰,۳۴	٪۸۴,۳۳	٪۹۳,۴۵	٪۹۴,۵۹	٪۹۶,۵۸	٪۹۷,۷۲	٪۹۸,۵۸	٪۹۹,۱۵	٪۹۹,۱۵	٪۹۹,۴۳
زمان	۰,۰۳۷۷	۰,۰۳۱۹	۰,۰۲۸۶	۰,۰۳۹۴	۰,۰۳۱۸	۰,۰۳۳۴	۰,۰۳۷۲	۰,۰۳۳۵	۰,۰۳۰۶	۰,۰۲۸۸



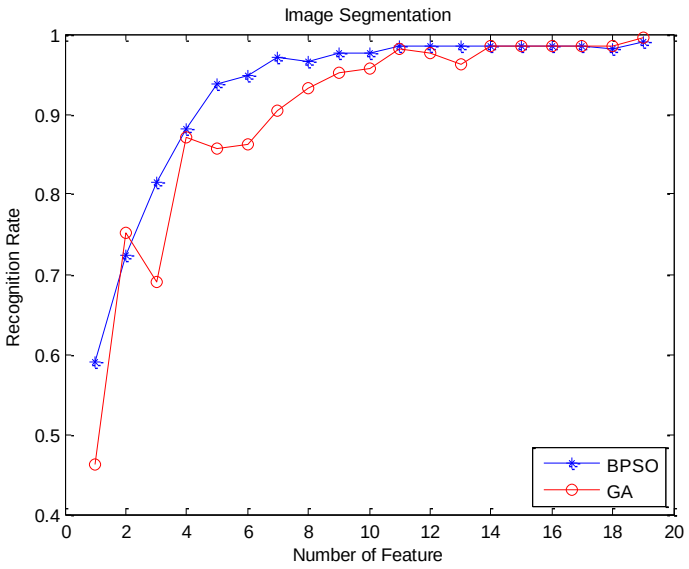
شکل ۶-۹:تغییرات نرخ بازشناسی بر اساس تعداد ویژگی های مختلف در داده Ionosphere

جدول ۶-۱۶:نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از GA+SVM بر روی داده Image segmentation

تعداد ویژگی	۲	۳	۴	۵	۷	۹	۱۱	۱۵	۱۷	۱۹
نرخ بازشناسی	٪۷۵,۲۴	٪۶۹,۰۵	٪۸۷,۱۴	٪۸۵,۷۱	٪۹۰,۴۸	٪۹۵,۲۴	٪۹۸,۱۰	٪۹۸,۵۷	٪۹۸,۵۷	٪۹۹,۵۲
زمان	۰,۰۳۸۶	۰,۰۵۵۳	۰,۰۳۴۵	۰,۰۴۷۰	۰,۰۲۸۴	۰,۰۳۰۱	۰,۰۲۲۵	۰,۰۳۰۳	۰,۰۲۱۵	۰,۰۲۰۱

جدول ۶-۱۷: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از BPSO+SVM بر روی داده Image segmentation

تعداد ویژگی	۲	۳	۴	۵	۷	۹	۱۱	۱۵	۱۷	۱۹
نرخ بازشناسی	٪۷۲,۳۸	٪۸۱,۴۳	٪۸۸,۱۰	٪۹۳,۸۱	٪۹۷,۱۴	٪۹۷,۶۲	٪۹۸,۵۷	٪۹۸,۵۷	٪۹۸,۵۷	٪۹۹,۵۲
زمان	۰,۰۳۱۱	۰,۰۲۱۵	۰,۰۱۹۳	۰,۰۱۷۶	۰,۰۱۴۰	۰,۰۱۵۳	۰,۰۱۳۴	۰,۰۱۴۸	۰,۰۱۵۲	۰,۰۱۵۵



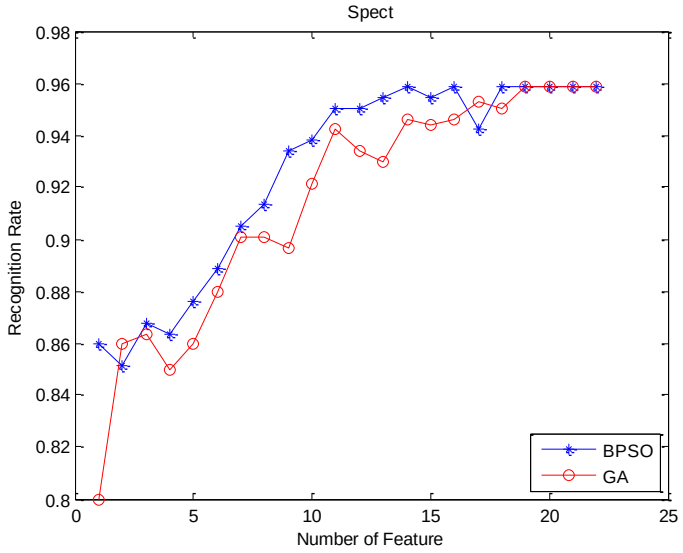
شکل ۶-۱۰: تغییرات نرخ بازشناسی بر اساس تعداد ویژگی های مختلف در داده Image Segmentation

جدول ۶-۱۸:نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از GA+SVM بر روی داده Spect

تعداد ویژگی	۱	۲	۳	۵	۷	۱۰	۱۳	۱۶	۱۹	۲۲
نرخ بازشناسی	٪۸۰	٪۸۵,۹۵	٪۸۶,۳۶	٪۸۵,۹۵	٪۹۰,۰۸	٪۹۲,۱۵	٪۹۲,۹۸	٪۹۴,۶۳	٪۹۵,۸۷	٪۹۵,۸۷
زمان	۰,۰۱۶۷	۰,۰۱۹۵	۰,۰۲۰۴	۰,۰۲۷۹	۰,۰۲۶۵	۰,۰۳۶۱	۰,۰۲۳۹	۰,۰۲۶۸	۰,۰۲۴۴	۰,۰۳۳۰

جدول ۶-۱۹: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از BPSO+SVM بر روی داده Spect

تعداد ویژگی	۱	۲	۳	۵	۷	۱۰	۱۳	۱۶	۱۹	۲۲
نرخ بازشناسی	٪۸۵,۹۵	٪۸۵,۱۲	٪۸۶,۷۸	٪۸۷,۶۰	٪۹۰,۵۰	٪۹۳,۸۰	٪۹۵,۴۵	٪۹۵,۸۷	٪۹۵,۸۷	٪۹۵,۸۷
زمان	۰,۰۰۸۷	۰,۰۱۰۶	۰,۰۱۲۶	۰,۰۱۵۳	۰,۰۱۷۴	۰,۰۱۷۹	۰,۰۱۵۷	۰,۰۱۷۰	۰,۰۱۹۲	۰,۰۲۱۱



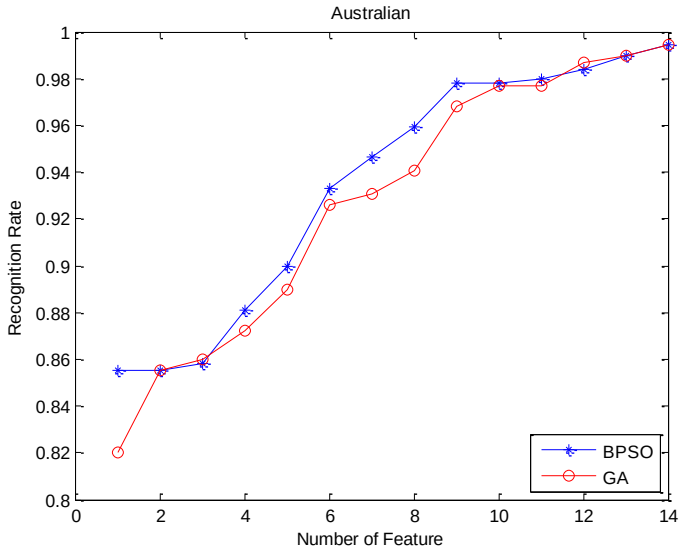
شکل ۶-۱۱: تغییرات نرخ بازشناسی بر اساس تعداد ویژگی های مختلف در داده Spect

جدول ۶-۲۰:نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از GA+SVM بر روی داده Australian

تعداد ویژگی	۲	۳	۴	۵	۶	۷	۸	۱۰	۱۲	۱۴
نرخ بازشناسی	٪۸۵,۵۱	٪۷۳,۷۷	٪۸۷,۲۵	٪۷۹,۱۳	٪۹۲,۶۱	٪۹۳,۰۴	٪۹۴,۰۶	٪۹۷,۶۸	٪۹۸,۷۰	٪۹۹,۴۲
زمان	۰,۱۴۳۵	۰,۲۷۸۲	۰,۱۴۹۳	۰,۳۳۴۷	۰,۲۰۱۸	۰,۱۵۷۸	۰,۱۴۷۸	۰,۱۲۷۱	۰,۱۴۲۲	۰,۱۰۹۸

جدول ۶-۲۱: نرخ بازشناسی برای تعداد ویژگی‌های مختلف با استفاده از BPSO+SVM بر روی داده Australian

تعداد ویژگی	۲	۳	۴	۵	۶	۷	۸	۱۰	۱۲	۱۴
نرخ بازشناسی	٪۸۵,۵۱	٪۸۵,۸۰	٪۸۸,۱۲	٪۹۰	٪۹۳,۳۳	٪۹۴,۶۴	٪۹۵,۹۴	٪۹۷,۸۳	٪۹۸,۴۱	٪۹۹,۴۲
زمان	۰,۰۵۰۵	۰,۰۶۱۵	۰,۰۸۵۹	۰,۰۶۹۸	۰,۰۶۳۳	۰,۰۷۰۶	۰,۰۷۹۴	۰,۱۰۶۶	۰,۰۸۳۸	۰,۰۹۰۶



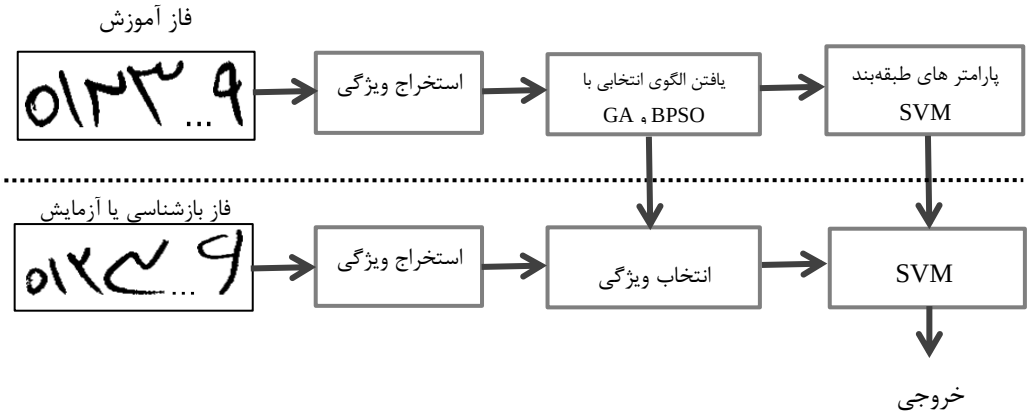
شکل ۶-۱۲: تغییرات نرخ بازشناسی بر اساس تعداد ویژگی های مختلف در داده Australian

۶-۳- بازشناسی ارقام دستنویس فارسی پایگاه داده هدی

بازشناسی ارقام دستنویس کاربردهای زیادی در شناسایی کدهای درج شده بر فرمهای مختلف و چکهای بانکی و کدپستی دارد[۵۴]. استفاده از یک سیستم بازشناسی ارقام دستنویس، در عمل با چالش-هایی مواجه است که مهمترین آنها ضرورت بالا بودن نرخ بازشناسی است. در حوزه زبان فارسی به دلیل شباهت زیادی که ارقام به هم دارند و همچنین تفاوت در شیوه رسم آنها، ایجاد یک سیستم بازشناسی با دقت قابل قبول برای استفاده عملی با مشکلاتی مواجه است. از این رو توسعه روشها برای بالا بردن نرخ بازشناسی در آنها ضروری است.

۶-۳-۱- سیستم پیشنهادی برای بازشناسی ارقام دستنویس

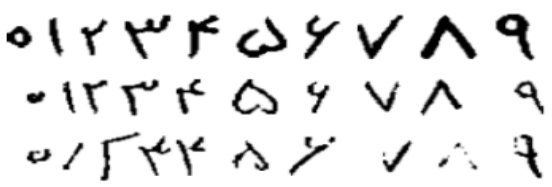
در شکل **۶-۱۳** سیستم پیشنهادی برای بازشناسی ارقام دستنویس نشان داده شده است.



شکل ۶-۱۳: سیستم پیشنهادی جهت بازشناسی ارقام دستنویس فارسی هدی

روش‌های مختلفی برای شناسایی ارقام دستنویس خصوصاً در زبان فارسی ارائه شده است که از جمله آنها می‌توان به روشهای آماری و ساختاری و روشهای مبتنی بر تبدیلات اشاره نمود. انتخاب روش استخراج ویژگی مهمترین عامل در بازشناسی الگو مطرح است[۲۵]. برای بازشناسی ارقام و حروف از ویژگی‌های ناحیه‌ای[۲۶]، گشتاورهای هندسی[۲۷]، گشتاورهای زرنیکی[۵۸]، توصیفگرهای فوریه[۵۹]،

پایاهای گشتاوری[۶۰]، هیستوگرام‌نما و ویژگی‌های مکان مشخصه [۶۱] استفاده شده است. یکی دیگر از کارهای مؤثر در بالا بردن نرخ بازشناسی، تفکیک ویژگی‌هایی است که علاوه بر اینکه در بالا بردن نرخ بازشناسی نقشی ندارند به کاهش نرخ بازشناسی نیز منجر می‌شوند. با انتخاب زیر مجموعه مناسب از ویژگی‌های استخراجی می‌توان در بالا نگه داشتن نرخ بازشناسی گام مناسبی برداشت. در تحقیقاتی که در زمینه بازشناسی ارقام دستنویس فارسی صورت گرفته پایگاه داده استانداردی ارائه شده است. این پایگاه از ۱۰۲۳۵۲تصاویر ارقام باینری که از حدود ۱۱۹۴۲ از دو نوع فرم ثبت نام آزمون سراسری، که توسط دانشجویان مقاطع کارشناسی و کارشناسی ارشد پرشده استخراج شده است. از این فرمها دونوع پایگاه داده ارقام و حروف تهیه شده است که در این مقاله از پایگاه داده ارقام آن استفاده شده است. تعداد کل ارقام این پایگاه داده ۱۰۲۳۵۲ رقم است که ۶۰۰۰۰ رقم آن به عنوان داده های آموزش و ۲۰۰۰۰ رقم آن به عنوان نمونه‌های آزمایش استفاده شده و ۲۲۳۵۲ رقم آن هم به عنوان ارقام باقی مانده در نظر گرفته شده است[۶۲]. شکل **۶-۱۴** تعدادی از این ارقام را نشان می دهد.



شکل ۶-۱۴: نمونه ای از اعداد پایگاه داده هدی

در این قسمت با استفاده از الگوریتم پیشنهادی در فصل ۵ ابتدا با استفاده از ترکیب الگوریتم‌های هیستوگرام‌گرادیان و مکان مشخصه توسعه یافته ویژگی‌های ارقام دستنویس فارسی پایگاه داده هدی استخراج شد و سپس در دو فاز عمل استخراج الگوی انتخابی و انتخاب ویژگی و بازشناسی طبق شکل (۶-۴) انجام گردید. ابتدا در فاز آموزش با استفاده از ترکیب روشهای پیشنهاد شده، استخراج ویژگی بر روی داده‌های آموزش صورت گرفته و با استفاده از الگوریتم بهینه سازی توده ذرات باینری بهترین الگوی انتخابی از ترکیب ویژگی‌ها در حالتی که بتواند بهترین مقدار تابع برازندگی معادله (۵-۸) را ارضا نماید، بدست آمد و همچنین در این فاز پارامترهای طبقه‌بند ماشین بردار پشتیبان SVM نیز ذخیره شد تا در فاز آزمایش دیگر داده های فاز آزمایش توسط طبقه‌بند فوق آموزش نبینند. در فاز آزمایش نیز مانند فاز

آموزش با استفاده از ترکیب الگوریتم‌های پیشنهادی ویژگی های داده‌های آزمایش استخراج شد و سپس با استفاده از بهترین الگوی انتخابی بدست آمده در فاز آموزش ویژگی‌های مجموعه داده آزمایش انتخاب و کاهش داده شده و سپس با استفاده از پارامترهای ذخیره شده برای طبقه‌بند عمل طبقه‌بندی داده‌های آزمایشی که کاهش ویژگی داده شده‌اند صورت می‌گیرد. الگوریتم پیشنهادی برای بازشناسی ارقام دستنویس فارسی منجر به مقاله ای تحت عنوان ((کاهش ویژگی توسط BPSO روی ترکیب ویژگی‌های هیستوگرام گرادیان و مکان مشخصه توسعه یافته برای بازشناسی ارقام دستنویس فارسی)) شد که در ادامه نتایج بدست آمده در این مقاله آورده شده است. در این مقاله از پایگاه داده ارقام دستنویس فارسی هدی استفاده شده است[۶۲]. با استفاده از ترکیب الگوریتم‌های هیستوگرام گرادیان و مکان مشخصه توسعه یافته ویژگی‌ها برای ۶۰۰۰۰ داده آموزش و ۲۰۰۰۰ داده تست استخراج شد و با کنار هم قرار دادن بردارهای ویژگی استخراجی از دو روش ذکر شده بردار ویژگی اصلی با طول ۴۰۰ بدست آمد. دقت بازشناسی برای ۲۰۰۰۰ داده تست با ۱۶ چرخش در روش گرادیان با استفاده از طبقه‌بند SVM بدون کاهش ویژگی در تعداد نمونه‌های آموزش مختلف بصورت جدول (۶-۲۰) می‌باشد. در بین داده‌ها بازشناسی ارقام ۲ و۳ نسبت به سایر ارقام بازشناسی پایینی دارند و ارقام ۰ و۱ و۸ بیشترین بازشناسی را به خود اختصاص داده‌اند.

جدول ۶-۲۲: دقت بازشناسی برای ۲۰۰۰۰ نمونه تست و تعداد نمونه های آموزش مختلف بدون کاهش ویژگی با استفاده از طبقه‌بند SVM

تعداد نمونه های آموزش	۵۰۰۰	۱۰۰۰۰	۲۰۰۰۰	۳۰۰۰۰	۴۰۰۰۰	۵۰۰۰۰	**۶۰۰۰۰**
دقت بازشناسی	٪۹۸,۴۶	٪۹۸,۸۴	٪۹۹,۱۱	٪۹۹,۲۲	٪۹۹,۲۹	٪۹۹,۳۳	**٪۹۹,۴۰**

درادامه در جدول (۶-۲۱) نتیجه بدست آمده با کارهایی که قبلاً بر روی این دیتابیس صورت گرفته مورد بررسی قرار گرفته است. سپس تعداد ویژگی‌ها را با استفاده از الگوریتم بهینه سازی توده ذرات باینری بر اساس انتخاب ویژگی کاهش داده شده است بطوریکه دقت بازشناسی بالا نگه داشته شود. برای این کار در تابع برازندگی معادله (۵-۸) مقدار K_2 بزرگتر از K_1 انتخاب شده است. پارامترهای انتخاب شده برای الگوریتم PSO باینری بصورت جدول (۶-۲۲) می‌باشد.

مقدار $w_{initial}$ برابر با ۰,۹ در نظر گرفته شده که پس از هر تکرار با استفاده از معادله (۳-۵) بروزرسانی شده است. نتایج بدست آمده از ترکیب ویژگی های استخراج شده بـا اسـتفاده از دو روش بـالا و کـاهش ویژگی در جدول (۶-۲۳) نشان داده شده است.

جدول ۶-۲۳: مقایسه کارهای قبلی با روش پیشنهادی بر روی دیتابیس هدی

الگوریتم های پیاده شده	اندازه دیتا بیس		درصد بازشناسی	
	آموزش	آزمایش	آموزش	آزمایش
Harifi., Aghagolzadeh [63]	230	500	-	97.60
Hosseini, Bouzerdum [64]	480	480	-	92
Mowlaei et al. [65]	2240	1600	99.29	91.88
Mozaffari et al. [66]	2240	1600	98.00	91.37
Mozaffari et al. [67]	2240	1600	100	94.44
Mowlaei, Faez [68]	2240	1600	100	92.44
Shirali-Shahreza et al.[69]	2600	1300	-	97.80
Soltanzadeh, Rahmati [70]	4979	3939	-	99.57
Dehghan, Faez [71]	6000	4000	-	97.01
Ziaratban et al. [72]	6000	4000	100	97.65
Sadri et al. [73]	7390	3035	-	94.14
A. Alaei et al. [74]	60000	20000	99.99	99.37
A. Alaei et al. [75]	60000	20000	99.99	99.02
H. Parvin et al. [76]	60000	20000	-	98.27
H.Parvin et al. [77]	60000	20000	-	98.89
نحوی، کیانی، ابراهیم پور[۸]	60000	20000	-	99.15
الگوریتم پیشنهادی	**60000**	**20000**	**100**	**99.40**

جدول ۶-۲۴: پارامتر های انتخاب شده برای الگوریتم PSO

تعداد ذرات	۳۰
ضریب c_1 و c_2	۲
ضریب K_1	۳
ضریب K_2	۶
تعداد تکرار	۱۵

جدول ۶-۲۵: دقت بازشناسی برای ۲۰۰۰۰ نمونه تست و تعداد نمونه های آموزش مختلف با کاهش ویژگی با استفاده از طبقه‌بند SVM

تعداد نمونه های آموزش	۵۰۰۰	۱۰۰۰۰	۲۰۰۰۰	۳۰۰۰۰	۴۰۰۰۰	۵۰۰۰۰	**۶۰۰۰۰**
تعداد ویژگی	۱۶۹	۱۱۰	۱۰۷	۱۸۰	۱۷۸	۲۰۱	**۱۹۳**
دقت بازشناسی	٪۹۸,۲۰	٪۹۸,۱۳	٪۹۸,۱۷	٪۹۸,۸۷	٪۹۸,۶۰	٪۹۸,۹۱	**٪۹۹,۱۱**

با مقایسه جدول (۶-۲۰) و جدول (۶-۲۳) می‌توان دید که دقت‌های بدست آمده از روش بدون کاهش ویژگی و با کاهش ویژگی به وسیله الگوریتم بهینه سازی توده ذرات باینری، تفاوت چندانی دربازشناسی با هم ندارند و با کاهش ویژگی تا تعداد قابل قبولی هنوز دقت بازشناسی به مقدار قابل قبولی در حد بالا قرار دارد. جدول (۶-۲۴) مقدار بازشناسی را برای تعداد ویژگی‌های مختلف با استفاده از الگوریتم BPSO نشان می‌دهد. در همه حالات تعداد نمونه آموزش برابر ۶۰۰۰۰ می‌باشد.

جدول ۶-۲۶: دقت بازشناسی برای ۲۰۰۰۰ نمونه تست و تعداد ویژگی های مختلف با استفاده از طبقه‌بند SVM

تعداد ویژگی	۹۸	۱۲۸	۱۴۱	۱۵۱	۱۶۵	۱۸۸	**۱۹۳**
دقت بازشناسی	٪۹۷,۹۵	٪۹۸,۶۲	٪۹۸,۸۸	٪۹۸,۵۳	٪۹۸,۶۹	٪۹۸,۹۰	**٪۹۹,۱۱**

۶-۳-۲- نتیجه گیری

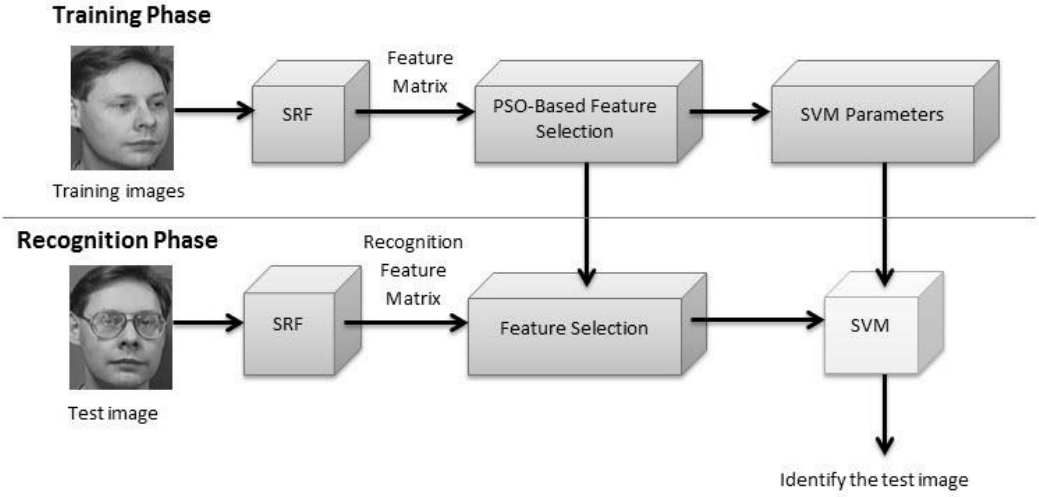
در این گزارش ویژگی‌های ارقام دستنویس فارسی با استفاده ازترکیب دو الگوریتم هیستوگرام گرادیان و مکان مشخصه توسعه یافته استخراج گردید و توسط طبقه‌بند SVM بازشناسی با ۴۰۰ ویژگی بدست آمده صورت گرفت و دقت ٪۹۹,۴۰ حاصل شد. توسط الگوریتم بهینه سازی توده ذرات باینری و تعریف تابع برازندگی خاص ۱۹۳ ویژگی انتخاب گردید و دقت بازشناسی ٪۹۹,۱۱ حاصل گردید. در تابع برازندگی ارائه شده دقت بازشناسی و تعداد ویژگی‌های مورد استفاده لحاظ شده است. بازشناسی در دو فاز آموزش و تشخیص صورت گرفت. در فاز آموزش که برای ۶۰۰۰۰ نمونه صورت گرفت پارامترهای طبقه‌بند SVM تعیین، و در فاز تشخیص مورد استفاده قرار گرفت. مقایسه نتایج با کار محققان دیگر حاکی از آن است روش ارائه شده در استخراج ویژگی و انتخاب ویژگی از کارایی مناسبی برخوردار است.

۶-۴-بازشناسی چهره، پایگاه داده ORL

۶-۴-۱- سیستم پیشنهادی برای بازشناسی چهره

در سیستم پیشنهاد شده برای شناسایی چهره مطابق شکل ۶-۱۵ مجموعه داده به دو مجموعه تست و آموزش تقسیم شد. برای مجموعه آموزش در فاز آموزش ابتدا ویژگی های تصاویر با استفاده از الگوریتم

سوبل – رابرتز استخراج شد سپس با استفاده از الگوریتم باینری PSO الگویی از نحوه انتخاب ویژگی‌ها برای داشتن بهترین درصد و کمترین تعداد ویژگی بدست آمد و نیز پارامتر های مربوط به طبقه‌بند SVM که برای محاسبه درصد استفاده شد، در این حالت ذخیره شد. برای مجموعه تست در فاز تشخیص ابتدا ویژگی‌های تصاویر با استفاده از الگوریتم سوبل – رابرتز استخراج شد سپس با استفاده از الگوی بدست آمده در فاز آموزش، ویژگی ها برای هر تصویر انتخاب شد و در نهایت با استفاده از پارامتر مربوط به طبقه‌بند SVM که در فاز آموزش ذخیره شد، تشخیص انجام گرفت و درصد تشخیص برای مجموعه تست بدست آمد.



شکل ۶-۱۵: سیستم پیشنهادی جهت بازشناسی چهره

ویژگی های سوبل – رابرتز توسط آقای خسروی و همکارش برای شناسایی فونت‌های فارسی استفاده شد. [۷۸]. در این قسمت ما از الگوریتم سوبل – رابرتز برای استخراج ویژگی از تصاویر پایگاه داده ORL استفاده کرده‌ایم.

۶-۴-۲- الگوریتم سوبل – رابرتز SRF

در این روش پس خواندن تصویر ابتدا تصویر با اندازه ۱۱۲×۹۲ به ۴ بلوک غیر همپوشان ۲۸×۲۳ تقسیم شده و سپس همانند روش هیستوگرام گرادیان با استفاده از ماسک سوبل در ۱۶ و ۸ و ۴ جهت به ترتیب تعداد ۲۵۶ و ۱۲۸ و ۶۴ ویژگی تولید می‌شود. همچنین با استفاده از ماسک رابرتز نیز همین مراحل را تکرار کرده و دقیقا همین تعداد ویژگی استخراج می‌نماییم سپس ویژگی‌های بدست آمده از این دو روش

را کنار هم قرار داده و طبق جدول (۶-۲۵) اندازه بردار ویژگی برای چرخش های متفاوت بدست می‌آید که این ویژگی ها را مورد استفاده قرار می‌دهیم.

جدول ۶-۲۷: تعداد ویژگی های استخراجی به روش SRF در چرخش های مختلف

ویژگی های سوبل + ویژگی های رابرتز	تعداد چرخش
۲۵۶+۲۵۶	۱۶
۱۲۸+۱۲۸	۸
۶۴+۶۴	۴

در این قسمت از پایگاه داده ORL استفاده شده است [۷۹]. این پایگاه داده بین سال ۱۹۹۲ تا ۱۹۹۴ در آزمایشگاه و توسط گروه رباتیک، ماشین بینایی و گفتار دانشکده مهندسی دانشگاه کمبریج جمع‌آوری شده است. این پایگاه داده شامل تصویرهای هشت بیتی خاکستری رنگ با ابعاد ۱۱۲×۹۲ می باشد که دارای ۱۰ عکس مختلف از ۴۰ نفر می باشد. برای تعدادی از افراد عکس ها در زمان های مختلف، نورپردازی مختلف، حالت های چهره خاص (چشمان باز و بسته، با لبخند و بدون لبخند)، جزئیات خاص (با عینک و بدون عینک) گرفته شده است نمونه‌ای از تصاویر موجود در این پایگاه داده بصورت شکل ۶-۱۶ می باشد.

شکل ۶-۱۶: نمونه ای از تصاویر پایگاه داده ORL

پس با استفاده از الگوریتم سوبل- رابرتز برای ۴، ۸ و ۱۶ جهت به ترتیب ۱۲۸، ۲۵۶ و ۵۱۲ ویژگی برای هر تصویر استخراج شد. درصد تشخیص برای تعداد تصاویر مختلف از هر شخص با استفاده از طبقه‌بند

SVM و بصورت جدول (۶-۲۶) می باشد. با توجه به جدول (۶-۲۶) بهترین درصدهای تشخیص بصورت جدول (۶-۲۷) می باشد.

جدول ۶-۲۸: درصد تشخیص برای تعداد تصاویر مختلف از هر شخص با استفاده از طبقه‌بند SVM

تعداد تصاویر	۱	۲	۳	۴	۵	۶	۷	۸	۹
۴ چرخش	٪ ۶۰	٪ ۸۷,۸۱	٪ ۹۲,۱۴	٪ ۹۸,۳	٪ ۹۸,۳	٪ ۹۸,۱	٪ ۹۹,۱۶	٪ ۱۰۰	٪ ۱۰۰
۸ چرخش	٪ ۶۸,۰۵	٪ ۸۵,۹۳	٪ ۹۶,۰۷	٪ ۹۷,۰۸	٪ ۹۶,۵	٪ ۹۹,۳	٪ ۹۹,۱۶	٪ ۱۰۰	٪ ۱۰۰
۱۶ چرخش	٪ ۶۸,۰۵	٪ ۸۸,۴۳	٪ ۹۵	٪ ۹۷,۵	٪ ۹۸,۵	٪ ۱۰۰	٪ ۱۰۰	٪ ۱۰۰	٪ ۱۰۰

جدول ۶-۲۹: بهترین درصد تشخیص برای تعداد تصاویر مختلف از هر شخص با توجه به جدول (۶-۲۶)

تعداد تصاویر	۱	۲	۳	۴	۵	۶	۷	۸	۹
بهترین درصد	٪ ۶۸,۰۵	٪ ۸۸,۴۳	٪ ۹۶,۰۷	٪ ۹۸,۳۳	٪ ۹۸,۵	٪ ۱۰۰	٪ ۱۰۰	٪ ۱۰۰	٪ ۱۰۰
کمترین تعداد ویژگی	۲۵۶	۵۱۲	۲۵۶	۱۲۸	۵۱۲	۵۱۲	۵۱۲	۱۲۸	۱۲۸

در ادامه تعداد ویژگی‌ها را با استفاده از الگوریتم بهینه‌سازی توده ذرات بر اساس انتخاب ویژگی کاهش دادیم بطوریکه درصد تشخیص بالا نگه داشته شد. برای این کار در تابع برآزندگی معادله (۵-۸) مقدار K_2 را بزرگتر از K_1 انتخاب کردیم. پارامتر های انتخاب شده برای الگوریتم باینری PSO بصورت جدول (۶-۲۲) می باشد. مقدار $W_{initial}$ اینجا نیز برابر با ۰,۹ در نظر گرفته شد. درصدهای تشخیص و تعداد ویژگی‌ها بصورت جدول (۶-۲۸) می‌باشد. بهترین درصدها در بین تعداد چرخش‌های مختلف همراه با کمترین تعداد ویژگی‌ها با توجه به جدول (۶-۲۸) بصورت جدول (۶-۲۹) می باشد.

جدول ۶-۳۰: درصد های تشخیص و تعداد ویژگی ها با استفاده از الگوریتم بهینه سازی توده ذرات بر اساس انتخاب ویژگی

تعداد چرخش	تعداد تصاویر	۱	۲	۳	۴	۵	۶	۷	۸	۹
۴	درصد	٪۶۸,۰۵	٪۸۳,۷۵	٪۸۹,۶۴	٪۹۵	٪۹۶,۵	٪۹۸,۱۲	٪۱۰۰	٪۱۰۰	٪۱۰۰
	تعداد ویژگی	۵۸	۵۲	۴۸	۵۲	۵۱	۴۸	۴۳	۵۳	۴۳
۸	درصد	٪۷۱,۶۶	٪۸۶,۲۵	٪۹۵,۳۷	٪۹۸,۳۳	٪۹۷	٪۹۹,۳۷	٪۱۰۰	٪۱۰۰	٪۱۰۰
	تعداد ویژگی	۱۱۲	۱۰۴	۱۱۱	۹۹	۹۰	۱۰۲	۱۰۷	۱۱۲	۱۰۳
۱۶	درصد	٪۷۰,۵۵	٪۹۰	٪۹۵,۷۱	٪۹۷,۵	٪۹۸,۵	٪۱۰۰	٪۱۰۰	٪۱۰۰	٪۱۰۰
	تعداد ویژگی	۲۰۵	۲۱۵	۲۲۱	۲۳۱	۲۰۷	۱۸۷	۲۱۲	۲۰۸	۲۰۶

جدول ۶-۳۱: بهترین درصدها در بین تعداد چرخش‌های مختلف همراه با کمترین تعداد ویژگی‌های با توجه به جدول (۶-۲۸)

تعداد تصاویر	۱	۲	۳	۴	۵	۶	۷	۸	۹
بهترین درصد	٪۷۱,۶۶	٪۹۰	٪۹۵,۷۱	٪۹۸,۳۳	٪۹۸,۵	٪۱۰۰	٪۱۰۰	٪۱۰۰	٪۱۰۰
کمترین تعداد ویژگی	۱۱۲	۲۱۵	۲۲۱	۹۹	۲۰۷	۱۸۷	۴۳	۵۳	۴۳
تعداد چرخش	۸	۱۶	۱۶	۸	۱۶	۱۶	۴	۴	۴

در ادامه تعداد ویژگی‌ها را بااستفاده از الگوریتم ژنتیک بر اساس انتخاب ویژگی کاهش دادیم بطوریکه درصد تشخیص بالا نگه داشته شد. نتایج بدست آمده بصورت جدول (۶-۳۰) می‌باشد. بهترین درصدها در بین تعداد چرخش‌های مختلف همراه با کمترین تعداد ویژگی بصورت جدول (۶-۳۱) می‌باشد.

جدول ۶-۳۲: درصدهای تشخیص و تعداد ویژگی‌ها با استفاده از الگوریتم ژنتیک بر اساس انتخاب ویژگی

تعداد چرخش	تعداد تصاویر	۱	۲	۳	۴	۵	۶	۷	۸	۹
۴	درصد	٪۶۶,۹۴	٪۸۶,۸۷	٪۹۰,۷۱	٪۹۶,۶۶	٪۹۸	٪۹۸,۱۲	٪۱۰۰	٪۱۰۰	٪۱۰۰
	تعداد ویژگی	۶۴	۷۹	۶۶	۷۰	۶۳	۷۱	۵۵	۵۳	۴۳
۸	درصد	٪۷۱,۶۶	٪۸۶,۸۷	٪۹۵,۳۵	٪۹۸,۳۳	٪۹۸	٪۹۹,۳۷	٪۱۰۰	٪۱۰۰	٪۱۰۰
	تعداد ویژگی	۱۴۱	۱۳۰	۱۳۰	۱۳۵	۱۳۲	۱۲۰	۱۲۳	۱۱۱	۱۱۱
۱۶	درصد	٪۶۶,۹۴	٪۹۰,۹۳	٪۹۵,۷۱	٪۹۸,۳۳	٪۹۸,۵	٪۱۰۰	٪۱۰۰	٪۱۰۰	٪۱۰۰
	تعداد ویژگی	۲۶۱	۲۴۹	۲۶۳	۲۵۸	۲۵۹	۲۴۱	۲۳۲	۲۳۸	۲۳۱

جدول ۶-۳۳: بهترین درصدها در بین تعداد چرخش‌های مختلف همراه با کمترین تعداد ویژگی‌ها با توجه به جدول (۶-۳۰)

تعداد تصاویر	۱	۲	۳	۴	۵	۶	۷	۸	۹
بهترین درصد	٪۷۱,۶۶	٪۹۰,۹۳	٪۹۵,۷۱	٪۹۸,۳۳	٪۹۸,۵	٪۱۰۰	٪۱۰۰	٪۱۰۰	٪۱۰۰
کمترین تعداد ویژگی	۱۴۱	۲۴۹	۲۶۳	۱۳۵	۲۵۹	۲۴۱	۵۵	۵۳	۴۳
تعداد چرخش	۸	۱۶	۱۶	۸	۱۶	۸	۴	۴	۴

با مقایسه جدول (۶-۳۱) و جدول (۶-۲۹) می‌توان دید که درصدهای بدست آمده از هر دو الگوریتم بهینه سازی توده ذرات بر اساس انتخاب ویژگی و الگوریتم ژنتیک بر اساس انتخاب ویژگی نزدیک هم

بوده ولی تعداد ویژگی‌ها در الگوریتم ژنتیک بر اساس انتخاب ویژگی بیشتر می‌باشد. برای مقایسه این دو الگوریتم تعداد ویژگی‌ها در الگوریتم ژنتیک را کاهش دادیم تا با تعداد ویژگی‌های الگوریتم باینری PSO برابر شود سپس درصدها را با هم مقایسه کردیم. جدول (۶-۳۲) درصد تشخیص به همراه تعداد ویژگی برابر با BPSO با استفاده از الگوریتم ژنتیک می‌باشد.

جدول ۶-۳۴: درصد تشخیص به همراه تعداد ویژگی برابر با الگوریتم BPSO با استفاده از الگوریتم ژنتیک

تعداد تصاویر	۱	۲	۳	۴	۵	۶	۷	۸	۹
بهترین درصد	٪۶۷,۲۲	٪۸۷,۱۸	٪۹۵,۳۵	٪۹۵,۴۱	٪۹۶,۵۵	٪۹۸,۱۲	٪۹۹,۱۶	٪۱۰۰	٪۱۰۰
کمترین تعداد ویژگی	۱۱۲	۲۱۵	۲۲۱	۹۹	۲۰۷	۱۸۷	۴۳	۵۳	۴۳
تعداد چرخش	۸	۱۶	۱۶	۸	۱۶	۱۶	۴	۴	۴

با مقایسه جدول (۶-۳۲) و جدول (۶-۲۹) می‌توان دید که با تعداد ویژگی برابر، الگوریتم بهینه سازی توده ذرات بر اساس انتخاب ویژگی در بیشتر موارد دارای درصدهای بالاتری نسبت به الگوریتم ژنتیک بر اساس انتخاب ویژگی می‌باشد. جدول (۶-۳۳) درصد تشخیص برای روش‌های مختلف بازای ۴ تصویر از هر شخص برای مجموعه آموزش را نشان می‌دهد. درصد تشخیص برای روش‌های مختلف در جدول (۶-۳۳) از مقاله [۱] آورده شده است. همانطور که در جدول (۶-۳۳) مشاهده می‌شود روش پیشنهادی دارای درصد تشخیص بالاتری نسبت به سایر روش‌ها بازای ۴ تصویر از هر شخص برای مجموعه آموزش می‌باشد. جدول (۶-۳۴) درصد تشخیص برای روش‌های مختلف بازای ۵ تصویر از هر شخص برای مجموعه آموزش را نشان می‌دهد. درصد تشخیص برای روش‌های مختلف در جدول (۶-۳۴) از مقاله [۸۰] آورده شده است. همانطور که در جدول (۶-۳۴) مشاهده می‌شود روش پیشنهادی دارای درصد تشخیص بالاتری نسبت به سایر روش‌ها بازای ۵ تصویر از هر شخص برای مجموعه آموزش می‌باشد.

جدول ۶-۳۵: درصد تشخیص برای روش‌های مختلف بازای ۴ تصویر از هر شخص برای مجموعه آموزش

انواع روش های تشخیص	درصد تشخیص
NCC	٪۸۴,۸۶
Hidden Markov model (HMMS)	٪۸۷
Eigenface	٪۹۰
SVM	٪۹۱,۲۱
DCT+GA	٪۹۴,۴
DCT+PSO FS	٪۹۴,۷
2D+HMM	٪۹۵
Hybrid NN: SOM+CNN	٪۹۶,۲
DWT+PSO FS	٪۹۶,۸
DWT+GA	٪۹۷
SRF+PSO FS+SVM	**٪۹۸,۳۳**

جدول ۶-۳۶: درصد تشخیص برای روش‌های مختلف بازای ۵ تصویر از هر شخص برای مجموعه آموزش

انواع روش های تشخیص	درصد تشخیص
Contourlet+LDA	٪۷۸,۵
ICA	٪۸۰,۵
LDA	٪۸۷,۵
PCA	٪۸۸,۵
DWT+LDA	٪۸۹,۵
Standard Eigenface	٪۹۲,۲
Waveletface	٪۹۲,۵
Waveletface + PCA	٪۹۴,۵
Waveletface +LDA	٪۹۴,۷
Waveletface +LDA+NFL	٪۹۵,۲
Waveletface+KAM	٪۹۶,۶
Curveletface+PCA	٪۹۶,۶
Curveletface+PCA+LDA	٪۹۷,۷
Curvelet+PCA	٪۹۸
SRF+PSO FS+SVM	**٪۹۸,۵**

مدت زمان برای کامپیوتری با مشخصاتی به صورت Core(TM)i3 CPU M370 @2.40GHz 2.39GHz برای فاز تشخیص بصورت جدول (۶-۳۵) می‌باشد.

جدول ۶-۳۷: مدت زمان لازم برای فاز تشخیص برای هر تصویر

تعداد چرخش	مدت زمان فاز تشخیص برای هر تصویر
۴	۹,۴۷۶ ms
۸	۹,۵۱۹ ms
۱۶	۹,۵۶۷ ms

همانطور که در جدول (۶-۳۵) مشاهده می‌شود با دو برابر شدن تعداد چرخش تقریباً ۰,۰۴۵ میلی ثانیه به زمان فاز تشخیص اضافه شده است. سرعت بالای این روش نسبت به روش‌های دیگر از مزیت‌های دیگر این روش است.

۶-۴-۳- نتیجه‌گیری

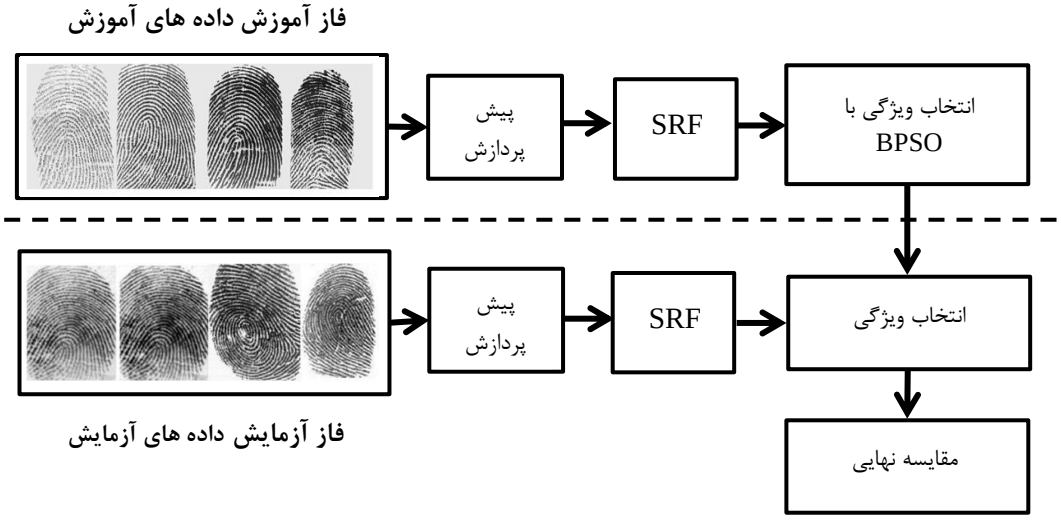
در این قسمت تشخیص چهره با استفاده از الگوریتم سوبل – رابرتز و بهینه سازی توده ذرات بر اساس انتخاب ویژگی صورت گرفت. ویژگی های چهره‌ای که با استفاده از الگوریتم سوبل-رابرتز در سه جهت ۴، ۸ و ۱۶ توسط مهدی پرهیزکاری و همکارش استخراج شده، استفاده شد. تابع برآزندگی مورد استفاده برای BPSO شامل درصد و تعداد ویژگی بود که درصد را با استفاده از طبقه بند SVM بدست آوردیم و برای فاز تشخیص نیز از طبقه بند SVM و پارامترهای ذخیره شده در فاز آموزش استفاده شد. با استفاده از BPSO الگویی از ویژگی‌ها در فاز آموزش بدست آمد که در فاز تشخیص اعمال شد. در بازشناسی چهره از مجموعه داده ORL استفاده شد. با استفاده از روش پیشنهادی برای ۴ و ۵ تصویر از هر شخص برای مجموعه آموزش به ترتیب به درصد تشخیص ۹۸,۳۳٪ و ۹۸,۵٪ رسیدیم. مدت زمان فاز تشخیص با استفاده از این روش نیز تقریباً ۹,۵ میلی ثانیه شد. نتایج بدست آمده نشان می‌دهد که این روش دارای دقت و سرعت بالا ست.

پیشنهاد می‌شود که قبل از استفاده از الگوریتم سوبل-رابرتز ابتدا از تبدیل موجک روی تصویر اصلی استفاده شود و سپس از تصویر تقریب با استفاده از الگوریتم سوبل –رابرتز ویژگی‌ها استخراج شود. اینکار باعث افزایش سرعت برنامه خواهد شد.

۶-۵- بازشناسی اثر انگشت :

۶-۵-۱- سیستم بازشناسی برای بازشناسی اثر انگشت :

در این بخش ویژگی های اثر انگشت داده های استاندارد دیتا بیس FVC2004 که شامل تصاویر ۸ بیتی خاکستری رنگ در ۴ بخش DB1 تا DB4 می باشد [۸۳] استفاده شده و با استفاده از الگوریتم SRF گفته شده در بخش تشخیص چهره توسط مهدی پرهیزکاری و همکارش استخراج شد، مورد استفاده قرار گرفته است. اما در این بخش از الگوریتم مقایسه اثر انگشت‌ها (فاصله اقلیدوسی) برای مقایسه بردارهای ویژگی استفاده شده و مقدار خطای خروجی مقایسه EER[1] در محاسبه تابع برازندگی بکار رفته است. در شکل **۶-۱۷** سیستم پیشنهادی جهت بازشناسی اثر انگشت نشان داده شده است.



شکل ۶-۱۷: سیستم پیشنهادی جهت بازشناسی اثر انگشت

ویژگی‌ها برای بازشناسی اثر انگشت در فاز آموزش برای هر یک از تصاویر مجموعه آموزش ابتدا پس زمینه حذف شد و کیفیت تصویر را بهبود داده سپس در تصویر بهبود یافته مرکز اثر انگشت را پیدا کرده و تصویر بهبود یافته را حول مرکز پیدا شده ۵ بار با گام ۲۲,۵ درجه از ۴۵- تا ۴۵+ درجه چرخاندیم و در هر بار چرخش ناحیه‌ای به مرکز اثر انگشت جدا شد. تصویر ورودی به ابعاد ۳۰۰×۳۰۰ تغییر اندازه داده

[1] Equal Error Rate

شده است. به دلیل اینکه تصاویر پایگاه داده اصلی در چرخش‌های مختلف اسکن شده بود طبق توضیحات در فایل همراه دیتا بیس این چرخش را از ۱۵- درجه تا ۱۵ درجه بیان کرده است. برای حاشیه اطمینان تصاویر را از ۴۵- درجه تا ۴۵ درجه چرخاندیم. سپس تصویر جدا شده به ۱۶ بلوک هم اندازه غیر همپوشان تقسیم و در هر یک از این نواحی جدا شده با استفاده از الگوریتم سوبل-رابرتز ویژگی‌ها استخراج شد. برای هر ماسک (سوبل و رابرتز) بردار ویژگی به ابعاد ۲۵۶ بدست آمد که در مجموع ۲۵۶+۲۵۶ ویژگی بدست آمد. پس با این اوصاف ما برای هر تصویر ۵ بردار با ۲۵۶+۲۵۶ ویژگی خواهیم داشت. این بردارها در یک جدول ذخیره شد. جدول ۶-۳۶ تعداد ویژگی‌های استخراجی برای ۴ نوع دیتا را نشان می دهد.

جدول ۶-۳۸: تعداد ویژگی های استخراجی برای ۴ نوع دیتا اثر انگشت

دیتا بیس	تعداد ویژگی‌ها
DB1	۱۶×۱۶×۲
DB2	۱۶×۱۶×۲
DB3	۱۶×۱۶×۲
DB4	۱۶×۱۶×۲

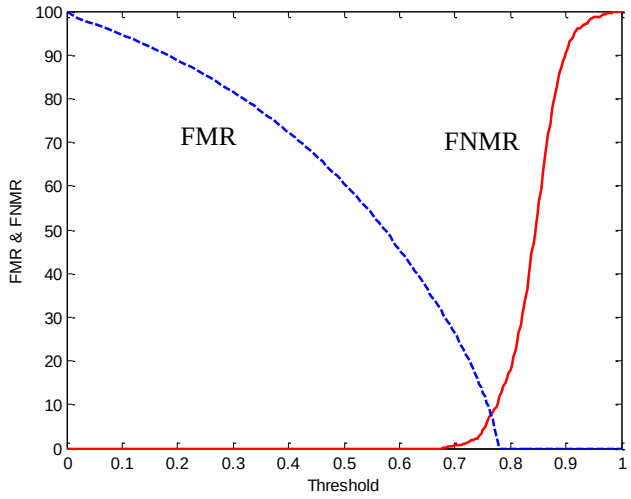
با استفاده از الگوریتم بهینه سازی توده ذرات براساس انتخاب ویژگی، ویژگی‌های نامطلوب برای داشتن کمترین مقدار EER حذف شد و الگویی از انتخاب ویژگی‌ها بدست آمد سپس بردارهای کاهش یافته در جدولی جدید ذخیره شد. شکل **۶-۱۸** نمونه ای از داده‌های اثر انگشت را نشان می‌دهد.

شکل ۶-۱۸: نمونه‌هایی از داده اثر انگشت

در فاز تشخیص برای هر یک از تصاویر مجموعه تست ابتدا پس زمینه حذف شد و کیفیت تصویر را بهبود داده سپس در تصویر بهبود یافته مرکز اثر انگشت را پیدا کرده و ناحیه ای به مرکز اثر انگشت جدا

شد. این ناحیه جدا شده نرمالیزه شد و سپس با استفاده از روش ذکر شده در بخش آموزش ویژگی های تست بدست آمد. سپس براساس الگوی بدست آمده در فاز تشخیص برای انتخاب ویژگی، ویژگی‌ها کاهش داده شد.

سیستم تطابق اثر انگشت می‌تواند دو نوع خطا تولید کند. نوع نخست تطابق اشتباه[1] نامیده می‌شود که در این خطا، واحد مقایسه، تصاویر دو اثر انگشت متفاوت را یکسان اعلام می‌کند. خطای دوم عدم تطابق اشتباه[2] نامیده می‌شود که در آن دو تصویر یک اثر انگشت واحد متفاوت اعلام می‌شوند. نرخ تطابق اشتباه یا (FMR) و عدم تطابق اشتباه(FNMR) به آستانه عملیاتی سیستم بستگی دارند. امتیاز آستانه خیلی بالا موجب کاهش شدید FMR و در عوض افزایش FNMR می‌شود. برای یک سیستم تشخیص اثر انگشت واحد، کاهش همزمان نرخ هر دوی این خطاها غیرممکن است. شکل **۶-۱۹**نمایی از FMR و FNMR را در حالتی که ویژگی کاهش داده نشده نشان می‌دهد.



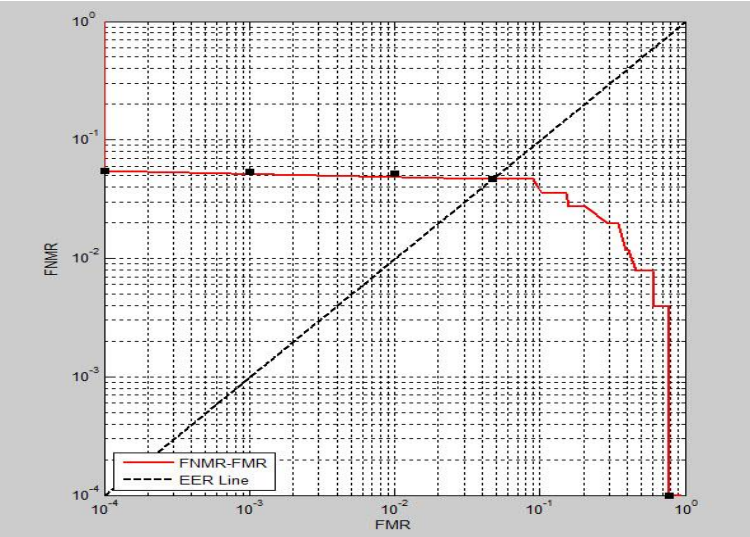
شکل ۶-۱۹: نمودار FMR و FNMR

در سیستم های اثر انگشت نقطه ای که این دو نمودار همدیگر را قطع می‌کنند را EER نامیده و نمایانگر کارایی سیستم است و هر چقدر این مقدار کمتر باشد کارایی سیستم بازشناسی اثر انگشت بالا می‌باشد[۸۱و۸۲].

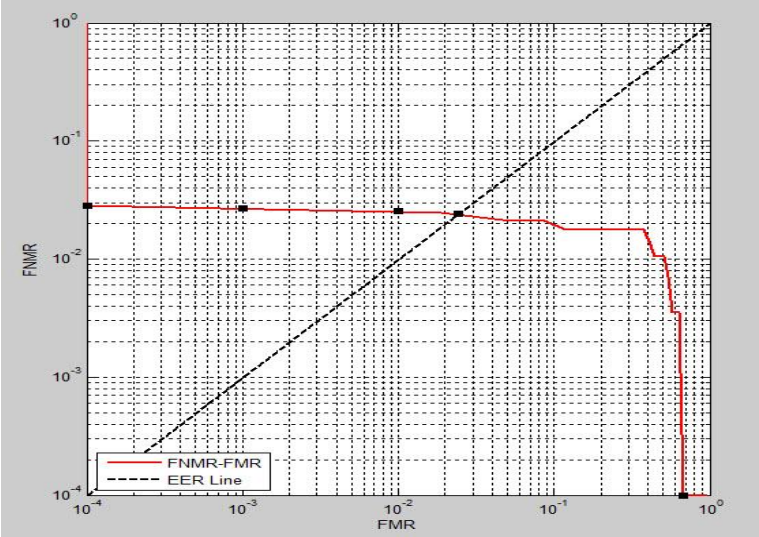
[1] False Match Rate
[2] False NonMatch Rate

۶-۵-۲- نتیجه گیری :

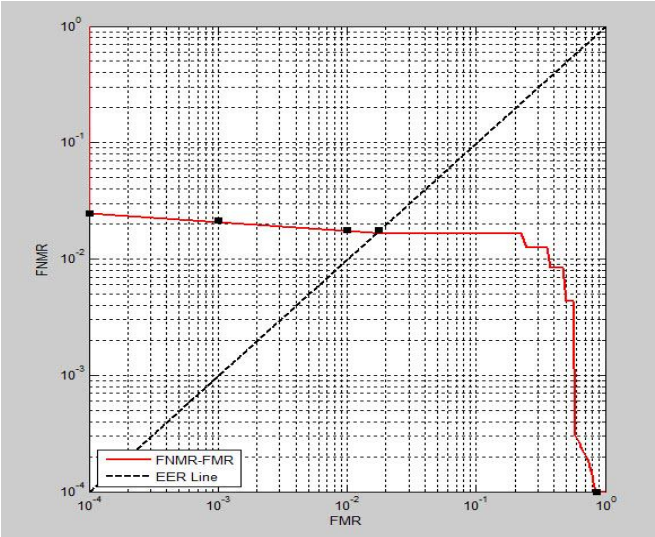
در این بخش نمودار خطای سیستم برای سیستم پیشنهادی نمودار FMR بر حسب FNMR برای هر یک از ۴ مجموعه داده DB1، DB2، DB3 و DB4 در شکل های ۶-۲۰، ۶-۲۱، ۶-۲۲ و ۶-۲۳ رسم شده است.



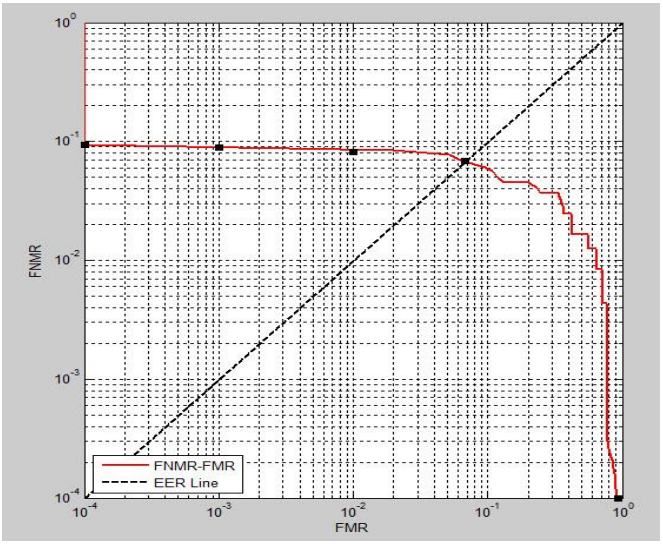
شکل ۶-۲۰: نمودار **FMR** بر حسب **FNMR** برای مجموعه داده **DB1**



شکل ۶-۲۱ : نمودار **FMR** بر حسب **FNMR** برای مجموعه داده **DB2**



شکل ۶-۲۲ : نمودار **FMR** بر حسب **FNMR** برای مجموعه داده **DB3**



شکل ۶-۲۳ : نمودار **FMR** بر حسب **FNMR** برای مجموعه داده **DB4**

روی هر یک از نمودارهای رسم شده ۵ نقطه با رنگ مشکی مشخص شده است که مقادیر عددی این ۵ نقطه برای ۴ مجموعه داده بصورت جدول ۶-۳۷ و میانگین آنها بصورت جدول ۶-۳۸ می باشد. در جدول ۶-۳۷ مدت زمان ثبت نام یک شخص در سیستم و نیز مدت زمان شناسایی یک شخص توسط سیستم برای ۴ مجموعه داده آورده شده است و میانگین این زمان‌ها در جدول ۶-۳۸ قرار داده شده است.

جدول ۶-۳۹: مقادیر نقاط روی نمودارهای **FMR** بر حسب **FNMR**

	خطا	FMR100	FMR1000	ZeroFMR	ZeroFNMR	میانگین زمان عضویت	میانگین زمان تطبیق
DB1	٪۴,۷۰	٪۵,۲۰	٪۵,۳۷	٪۵,۴۹	٪۷۷,۷۱	۱,۷۱	۰,۳۳
DB2	٪۲,۴۱	٪۲,۵۴	٪۲,۷۰	٪۲,۸۲	٪۶۷,۲۹	۱,۱۵	۰,۲۷
DB3	٪۱,۷۶	٪۱,۷۷	٪۲,۱۳	٪۲,۴۷	٪۸۶	۱,۰۷	۰,۲۱
DB4	٪۶,۸۰	٪۸,۲	٪۸,۹۲	٪۹,۳۲	٪۹۲,۸	۱,۰۲	۰,۱۹

جدول ۶-۴۰: میانگین نتایج جدول ۶-۳۷ برای ۴ مجموعه داده

	میانگین خطا	میانگین FMR100	میانگین FMR1000	میانگین ZeroFMR	میانگین ZeroFNMR	میانگین زمان عضویت	میانگین زمان تطبیق
روش پیشنهادی	٪۳,۹۱	٪۴,۴۲	٪۴,۷۸	٪۵,۰۲	٪۸۰,۹۵	۱,۲۳	۰,۲۵

برای مقایسه روش پیشنهادی با روش های دیگر از نظر میانگین زمان تطبیق و میانگین EER جدول ۶-۳۹ آورده شده است. نتایج سایر روشها که در جدول ۶-۳۹ از مرجع [۸۴] آورده شده است.

جدول ۶-۴۱: مقایسه روش های مختلف و روش پیشنهاد شده

الگوریتم ها	میانگین زمان تطبیق	میانگین خطا
P101	1.48 s	2.07%
P047	2.07 s	2.10%
P071	0.67 s	2.30%
P004	0.71 s	2.45%
P039	1.19 s	2.90%
P097	0.51 s	3.13%
P049	0.38 s	3.24%
P009	0.25 s	3.31%
P113	0.48 s	3.71%
الگوریتم پیشنهادی	**0.25 s**	**3.91%**
P068	0.47 s	4.03%
P108	0.32 s	4.04%
P016	0.34 s	4.27%
P103	0.14 s	4.33%
P048	0.39 s	4.41%
P075	0.43 s	4.79%
P041	0.17 s	4.89%
P111	0.43 s	5.66%
P052	0.24 s	6.12%
P050	0.61 s	6.23%
P027	2.63 s	6.42%
P120	1.30 s	6.56%
P083	0.42 s	6.80%
P067	0.05 s	7.17%
P026	3.78 s	7.27%
P011	0.47 s	7.48%
P072	0.11 s	7.64%

تعداد ویژگی های جدید با استفاده از بهینه سازی توده ذرات بر اساس انتخاب ویژگی برای ۴ مجموعه داده بصورت جدول ۶-۴۰ می باشد.

جدول ۶-۴۲ : تعداد ویژگی های جدید برای ۴ مجموعه داده با استفاده **BPSO**

مجموعه داده	تعداد ویژگی
DB1	۱۷۳
DB2	۱۹۴
DB3	۱۶۵
DB4	۱۴۳

مقادیر انتخاب شده برای الگوریتم بهینه سازی توده ذرات بر اساس انتخاب ویژگی بصورت جدول ۶-۴۱ می باشد.

جدول ۶-۴۳ : مقادیر انتخاب شده برای الگوریتم **BPSO**

تعداد ذرات	۳۰
ضریب C_1 و C_2	۲
ضریب K_1	۵
ضریب K_2	۶
تعداد تکرار	۵۰

۶-۵-۳- استفاده از طبقه بند ماشین بردار پشتیان در بازشناسی اثر انگشت

در این بخش ۴ تصویر از هر شخص برای مجموعه آموزش و ۴ تصویر از هر شخص برای مجموعه تست بطور تصادفی انتخاب شد. در فاز آموزش برای هر تصویر از مجموعه آموزش ۵ بردار ویژگی با طول ۵۱۲ استخراج شد که با استفاده از بهینه سازی توده ذرات بر اساس انتخاب ویژگی طول بردار مطابق جدول ۶-۴۰ کاهش داده شد. بردارهای ویژگی بدست آمده برای کل تصاویر مجموعه آموزش بصورت

یک جدول ذخیره شد. با استفاده از جدول بدست آمده طبقه بند ماشین بردار پشتیان آموزش داده شد و پارامترهای طبقه بند ذخیره شد.

در فاز تشخیص برای هر تصویر از مجموعه تست همانند فاز آموزش ویژگی ها استخراج شد و بردار ویژگی بدست آمد سپس با استفاده از پارامترهای بدست آمده در فاز آموزش برای طبقه بند ماشین بردار پشتیان شناسایی صورت گرفت. نتایج برای ۴ مجموعه داده بصورت جدول ۶-۴۲ می باشد.

جدول ۶-۴۴: درصد شناسایی با استفاده ازطبقه بند **SVM** روی ۴ مجموعه داده

مجموعه داده	درصد تشخیص
DB1	٪۹۴
DB2	٪۹۴
DB3	٪۹۶,۵
DB4	٪۹۳,۵
میانگین درصد تشخیص	**٪۹۴,۵**

فصل ۷

نتیجه‌گیری و پیشنهاد

در این تحقیق بهینه سازی توده ذرات باینری بر اساس انتخاب ویژگی با استفاده از طبقه‌بند ماشین-بردار پشتیبان به عنوان تابع برازندگی سیستمی برای تشخیص ارقام دستنویس فارسی با طول بردار ویژگی ۴۰۰ استخراج شده با استفاده از ترکیب روشهای هیستوگرام گرادیان و مکان مشخصه توسعه یافته معرفی شد.

طبق نتایج بدست آمده در فصل ۶ روش پیشنهادی نسبت به سایر روشها از کارایی و دقت خوبی برخوردار است. در بهترین حالت برای بازشناسی ارقام دستنویس فارسی بدون استفاده از بهینه سازی توده ذرات جهت انتخاب ویژگی به نرخ بازشناسی ۹۹,۴۰٪ دست یافتیم. سپس با استفاده از بهینه سازی توده ذرات باینری BPSO توانستیم در بهترین حالت ویژگی‌ها را تا مقدار ۱۹۳ کاهش داده و نرخ بازشناسی رادر سطح ۹۹,۱۱٪ که درصد خوبی محسوب می‌شود نگه داریم.

هر چند در این تحقیق ما از بهینه سازی توده ذرات باینری برای انتخاب ویژگی استفاده کردیم اما می‌توان از سایر الگوریتم های بهینه سازی بجای PSO استفاده کرد و نتایج را مقایسه نمود. و همچنین از تابع ارزیابی دیگری برای این منظور می‌توان استفاده کرد.

با در دست داشتن الگوریتم بهینه و کارا برای انتخاب ویژگی می‌توان بر روی داده‌های مختلف مانند گفتار و تصویر کارهای متفاوتی صورت داد. همچنین در روش استخراج ویژگی بیان شده در فصل ۵ می-توان ابتدا از تصویر ورودی تبدیل موجک گرفت سپس بر روی داده‌های تقریب بدست آمده از اعمال موجک روش استخراج ویژگی پیشنهادی را پیاده نماییم که این عمل به دلیل کاهش ابعاد ویژگی در مرحله تبدیل موجک گیری، باعث افزایش سرعت که امر مهم و حیاتی در سیستم‌های بلادرنگ است می‌باشد.

یکی از مهمترین کارهایی که می‌توان در بخش انتخاب ویژگی صورت داد کار بر روی تابع برازندگی است. تابع برازندگی نیز امر مهم و حیاتی در رسیدن الگوریتم های بهینه سازی به جواب بهینه است یعنی اگر بهترین روش را برای پیاده‌سازی الگوریتم انتخاب کرده باشید اما تابع برازندگی شما بد عمل نماید کل الگوریتم را به دور از جواب بهینه نگه می‌دارد.

پس بر اساس نتایج بدست آمده بر روی ۴ دیتا بیس استاندارد مختلف مشاهده نمودیم ویژگی هایی نیز وجود دارند که بر نرخ بازشناسی اثر منفی می گذارند یعنی ویژگی هایی می باشند که یا نویز بوده اند یا اینکه شامل نویز می باشند و یا ویژگی های نمونه هایی که نسبت به کلاس مورد نظر نامرتبط می باشند. همچنین بر این اساس شرایطی نیز گاهی اوقات پیش می آید که اضافه نمودن ویژگی ها باعث کاهش نرخ بازشناسی می شود که علتش وجود همان ویژگی هایی که ذکر شد است و می توان با حذف این ویژگی ها نرخ بازشناسی را بالا برد.

فهرست مراجع

[1] R. M. Ramadan and R. F. Abdel-Kader, "Face Recognition Using Particle Swarm Optimization-Based Selected Features," International Journal of Signal Processing, Image Processing and Pattern Recognition, vol. 2, pp. 51-66, June 2009.

[2] T. Chung-Jui, et al., "Feature Selection using PSO-SVM," International Journal of Computer Science, 33:1, IJCS_33_1_18, 13 February 2007.

[3] L.-Y. Chuang, et al., "Improved binary PSO for feature selection using gene expression data," Computational Biology and Chemistry, vol. 32, pp. 29–38, 2008.

[4] A. Abdulwahhab Ghidan and A. A. M. a. h. Al-Saedi, "Face Recognition Based On Mixed Between Selected Feature By Multiwavelet And Particle Swarm Optimization," Developments in E-systems Engineering, IEEE, 2010.

[5] A. Alaei, et al., "Using Modified Contour Features and SVM Based Classifier for the Recognition of Persian/Arabic Handwritten Numerals," Proceedings of 7th International Conference on Advances in Pattern Recognition, pp. 391-394, 2009.

[6] A. Alaei, et al., "Fine Classification of Unconstrained Handwritten Persian/Arabic Numerals by Removing Confusion amongst Similar Classes," Proceedings of 10th International Conference on Document Analysis and Recognition, pp. 601-605, 2009.

[7] H. Parvin, et al., "A New Approachto Improve the Vote-Based Classifier Selection," Proceedings of Fourth International Conference on Networked Computing and Advanced Information Management, pp. 91-95, 2008.

[8] محمد نحوی، کوروش کیانی و رضا ابراهیم پور، " بهبود روش استخراج ویژگی گرادیان مبتنی برتبدیل گسسته کسینوسی جهت بازشناسی ارقام دستنوشته فارسی " هجدهمین کنفرانس مهندسی برق ایران، صفحات ۳۰۶۷ الی ۳۰۷۱، ۲۱ الی ۲۳ اردیبهشت ۱۳۸۹.

[9] I. K. Fodor, "A survey of dimension reduction techniques," technical report, Lawrence Livemore National Laboratory, June 2002.

[10] صادق سلیمان پور، "معرفی روشهای مختلف انتخاب ویژگی"، گزارش تحقیق درس شناسائی آماری الگو، زمستان ۱۳۸۴.

[11] Y. Zhu, "High Performance Data Mining in Time Series," Techniques and Case Studies, New York University, January 2004.

[12] L. I. Smith, A tutorial on Principal Components Analysis, 2002.

[13] M. Dash and H. Liu, "Feature Selection for Classification," Intelligent Data Analysis vol. 1, pp. 131-156, 1997.

[14] J. C. Schlimmer, "Efficiently inducing determinations: A complete and systematic search algorithm that uses optimal pruning," Proceedings of Tenth International Conference on Machine Learning, pp. 284–290, 1993.

[15] http://www.swarmintelligence.org/tutorials.php.

[16] G. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. New York: Addison Wesley, 1989.

[17] M. Mitchell, An Introduction to Genetic Algorithm: MIT Press, 1996.

[18] C. Emmanouilidis, et al., "A multiobjective evolutionary setting for feature selection and a commonality-based crossover operator," In Congress on Evolutionary Computing, vol. 2, pp. 309-316, 2000.

[19] C. B. Congdon, "A comparison of genetic algorithm and other machine learning systems on a complex classi," Ph.D Thesis, University of Michigan, 2002.

[20] J. Kennedy and R. Eberhart, "Particle Swarm Optimization " vol. vol 4, 27 Nov-1 Dec ed: Presented at the IEEE International Conference on neural networks 1995.

[21] A. Unler and A. Murata, "Discrete particle swarm optimization method for feature selection in binary classification problems," European Journal of Operational Research, vol. 206, pp. 528–539, 2010.

[22] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," IEEE Transactions on Evolutionary Computation, vol. 6, pp. 58–73, 2002.

[23] K. E. Parsopoulos, et al., "Improving the Particle Swarm Optimizer by function "STRETCHING" " Kluwer Academic Publisher, Dordrecht, The Netherlands, vol. vol 54, pp. 28, pp.445-457., 2001.
[24] R.Hassan, et al., "A Comparison Of Particle Swarm Optimization and The Genetic Algorithm," ed: AIAA/ASME/AHS Adaptive Structures Conference 7t,Austin, Texas, Apr. 18-21, 2005.
[25] J. Kennedy and R. Eberhart, "A Discrete Binary Version of the Particle Swarm Algorithm," presented at the presented at the IEEE International Conference on Systems, Man, and Cybernetics, 1997.
[26] M.SETTLES and B.RYLANDER. Neural Network Learning using Particle Swarm Optimization. Available: http://upibm9.egr.up.edu/contrib/rylander/studpubs/ matt.pdf
[27] A.Stacey, et al., "Particle Swarm Optimization with Mutation," The 2003 Congress on Evolutionary Computation vol. vol 2, 8-12 Dec 2003.
[28] M.P.Wachowiak, et al., "An Approach to Multimodel Biomedical Image Registration Utilizing Particle Swarm Optimization," IEEE Transactions on Evolutionary Computation, vol. vol 8, pp. 3, JUNE 2004.
[29] W.JIAN, et al., "An Improvmented Particle Swarm Optimization Algorithm With Neighborhoods Topologies," presented at the Proceedings of the Third International Conference on Machine Larning and Cybemetics,Shanghai, 2004.
[30] P. N. Suganthan, "Particle Swarm Optimiser With Neighbourhood Operator," Proceedings of the 1999 Congress on Evolutionary Computation vol. vol3,6-9 July 1999.
[31] M. Dorigo and L. M. Gambardella, "Ant colony system: Acooperative learning approach to the traveling salesman problem," IEEE Transaction on Evolutionary Computation, vol. 1(1), 53–66, 1997.
[32] E. Bonabeau, et al., "Swarm Intelligence: From Natural to Artificial Systems," in Oxford Univ. Press, ed. New York, 1999.
[33] M. Dorigo, et al., "The Ant system: Optimization by acolony of cooperating agents," IEEE Trans. Syst. Man Cybernet, vol. Part B 26, pp. 29–41, 1996.
[34] V. Maniezzo and A. Colorni, "The ant system applied to the quadratic assignmentproblem," Knowledge Data Eng, vol. 11, pp. 769–778, 1999.
[35] A. Al-Ani, "An Ant Colony Optimization Based Approach for Feature Selection," presented at the Proceeding of AIML Conference, 2005.
[36] M. Goodarzi, et al., "Ant colony optimization as a feature selection method in the QSAR modeling of anti-HIV-1 activities of 3-(3,5-dimethylbenzyl)uracil derivatives using MLR, PLS and SVM regressions, Chemometrics and Intelligent Laboratory Systems," vol. 98, pp. 123–129, 2009.
[37] Y. Chen, et al., "A rough set approach to feature selection based on ant colony optimization," Pattern Recognition Letters vol. 31, pp. 226–233, 2010.
[38] R. Jensen and Q. Shen, "Finding rough set reducts with ant colony optimization," Proceeding of 2003 UK Workshop Computational Intelligence, pp. 15–22, 2003.
[39] F Glover, "Future paths for integer programming and links to artificial intelligence," Computers and Operations Research, vol. 5, pp. 533-549, 1986.
[40] F. Glover, et al., Tabu Search, In Handbook of AppliedOptimization, P.M Pardalos M .G .C Resende ed.: Oxford University Press, 2002.
[41] K. Hammouchea, et al., "A comparative study of various meta-heuristic techniques applied to the multilevel thresholding problem," Engineering Applications of Artificial Intelligence, vol. 23, pp. 676–688, 2010.
[42] V. Vapnik, The Nature of Statistical Learning Theory: Springer Verlag, 1995.
[43] C.Cortes and V.Vepnik, "Support Vector Network," Machine Learning, vol. 20, pp. 273-279, 1995.
[44] S. Abe, "Support Vector Machine for Pattern Classification," Springer-Verlog, Ed., ed. London, 2005.
[45] M. Martinez-Ramon and C.Christodoulou, Support Vector Machines for Antenna Array Processing and Electromagnetics. USA: Morgan & Claypool Publishers, 2006.

[46] U. H. G. Krebel, Pairwise Classification and Support Vector Machines. MIT Press, Cambridge, MA: B.Scholkopf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods: Support Vector Learning, 1999.
[47] This LIBSVM implementation document was created in 2001 and has been maintained at http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.pdf.
[48] This LIBSVM implementation document was created in 2001 and has been maintained at http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.
[49] حسین خسروی، احسان اله کبیر،" معرفی دو ویژگی سریع و کارآمد برای بازشناسی ارقام دستنویس فارسی " چهارمین کنفرانس ماشین بینایی و پردازش تصویر ایران، دانشگاه فردوسی مشهد، صفحات ۲۵ الی ۲۶، بهمن ماه ۱۳۸۵.
[50] J. Yang and V. Honavar, Feature Subset Selection using a Genetic Algorithm, Feature Eztraction Construction and Selection: A data Mining Perspective ed.: Kluwer Academic Publishers, 1998.
[51] T. Sergios, et al., Pattern Recognition: Academic Press, 1st edition, January 15, 1999.
[52] A. Nazif and T. Y.-V. Fatos, "An overview of character recognition based focused on off-line handwriting," IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews, vol. 31, 2001.
[53] عزمی رضا، "بازشناسی متون چاپی فارسی"، پایان نامه دکتری، به راهنمایی: کبیر احسان اله، دانشکده فنی و مهندسی، دانشگاه تربیت مدرس، ۱۳۷۸.
[54] S. Impedovo, et al., Automatic Bankcheck Processing. Singapore: World Scientific 1997.
[55] I. D. Trier and A. K. Jain, "Feature extraction Methods for Character Recognition- A Survey," Pattern Recognition, vol. 29, pp. 641-662, 1996.
[56] H. Takahashi, "A Neural Net OCR using geometrical and zonal pattern features," presented at the in proceedings of the first international Conference on Document Analysis and Recognition, 1991.
[57] L. O. Jimenez and A. Morales-Morell, "Classification of Hyperdimensional Data Based on Feature and Decision Fusion Approachs Using Projection Pursuit, Majority Voting, and Neural Networks," IEEE Trans. on Geoscience and Remote Sensing, vol. 37, May 1999.
[58] محمد حسن شیرعلی شهر رضا، علیرضا ختن زاده، "تشخیص اعداد چاپی مستقل از اندازه و جابجایی با استفاده از گشتاورهای زرنیکی و به کمک شبکه های عصبی" دومین کنفرانس مهندسی برق ایران، جلد۵، صفحات ۴۱۷ الی ۴۲۴، دانشگاه تربیت مدرس، تهران، ۲۴ الی ۳۰ اردیبهشت۱۳۷۳.
[59] رضا عزمی، احسان اله کبیر،کامبیز بدیع، " بازشناسی حروف چاپی با استفاده ازویژگی های منحنی پیرامونی " نشریه علمی پژوهشی انجمن کامپیوترایران، جلد۱، شماره ۱، صفحات ۲۹ الی ۳۷، بهار ۱۳۸۲.
[60] Y. Li, "Reforming the theory of invariant moments for Pattern recognition," Pattern Recognition Letters, vol. 25, pp. 723-730, July 1992.
[61] H. A. Glucksman, "Multicategory of Patterns Represented by High-Order Vectors of Multilevel Measurement," IEEE Transaction Computer, pp. 1593- 1598, Dec 1971.
[62] H. Khosravi and E. Kabir, "Introducing a very large dataset of handwritten Farsi digits and a study on their varieties," Pattern Recognition Letters, vol. 28, pp. 1133-1141, 2007.
[63] A. Harifi and A. Aghagolzadeh, "A New Pattern for Handwritten Persian/ Arabic Digit Recognition," Journal of Information Technology, vol. 3, pp. 249-252, 2004.
[64] H. M. M. Hosseini and A. Bouzerdoum, "A Combined Method for Persian and Arabic Handwritten Digit Recognition," presented at the Australian New Zealand Conference on Intelligent Information System, 1996.
[65] A. Mowlaei, et al., "Feature Extraction with Wavelet Transform for Recognition of Isolated Handwritten Farsi/Arabic Characters and Numerals," Digital Signal Processing, vol. 2, pp. 923-926, 2002.
[66] S. Mozaffari, et al., "Recognition of Isolated Handwritten Farsi/Arabic Alphanumeric Using Fractal Codes," Image Analysis and Interpretation, 6th Southwest Symposium, pp. 104-108, 2004.
[67] S. Mozaffari, et al, "Structural Decomposition and Statistical Description of Farsi/Arabic Handwritten Numeric Characters " Proceedings of the 8th Intl. Conference on Document Analysis and Recognition vol. 1, pp. 237- 241, 2005.

[68] A. Mowlaei and K. Faez, "Recognition Of Isolated Handwritten Persiawarabic Characters And Numerals Using Support Vector Machines " Proceedings of XIII Workshop on Neural Networks for Signal Processing, pp. 547-554, 2003.
[69] M. H. Shirali-Shahreza, et al., "Recognition of Hand-written Persian/Arabic Numerals by Shadow Coding and an Edited Probabilistic Neural Network," Proceedings of International Conference on Image Processing, vol. 3, pp. 436-439, 1995.
[70] H. Soltanzadeh and M. Rahmati, "Recognition of Persian handwritten digits using image profiles of multipleorientations," Pattern Recognition Letters vol. 25, pp. 1569–1576, 2004.
[71] M. Dehghan and K. Faez, "Farsi Handwritten Character Recognition With Moment Invariants," Proceedings of 13th International Conference on Digital Signal Processing, vol. 2, pp. 507-510, 1997.
[72] M. Ziaratban, et al., "Language-Based Feature Extraction Using Template-Matching in Farsi/Arabic Handwritten Numeral Recognition," Proceedings of 9th International Conference on Document Analysis and Recognition, vol. 1, pp. 297-301, 2007.
[73] J. Sadri, et al., "Application of Support Vector Machines for Recognition ofHandwritten Arabic/Persian Digits," Proceedings of the 2nd Conference on Machine Vision and Image Processing & Applications, vol. 1, pp. 300-307, 2003.
[74] A. Alaei, et al., "Using Modified Contour Features and SVM Based Classifier for the Recognition of Persian/Arabic Handwritten Numerals," Proceedings of 7th International Conference on Advances in Pattern Recognition, pp. 391-394, 2009.
[75] A. Alaei, et al., "Fine Classification of Unconstrained Handwritten Persian/Arabic Numerals by Removing Confusion amongst Similar Classes," Proceedings of 10 thInternational Conference on Document Analysis and Recognition, pp. 601-605, 2009.
[76] H. Parvin, et al., "A New Approachto Improve the Vote-Based Classifier Selection," Proceedings of Fourth International Conference on Networked Computing and Advanced Information Management, pp. 91-95, 2008.
[77] H. Parvin, et al., "A Scalable Method for Improving the Performance of Classifiers in Multiclass Applications by Pairwise Classifiers and GA," Proceedings of Fourth International Conference on NetworkedComputing and Advanced Information Management, pp. 137-142, 2008.
[78] H. Khosravi and E. Kabir, "Farsi font recognition based on Sobel–Roberts features," Pattern Recognition Letters, vol. 31, pp. 75-82, 2010.
[79] The ORL Database of Faces: Cambridge University Computer Laboratory, between April 1992 and April 1994.
[80] e. a. M. Rziza, "Local Curvelet Based Classification Using Linear Discriminant Analysis for Face Recognition," International Journal of Electrical and Computer Engineering, vol. 4, pp. 72-77, 2009.
[81] D. Maio, et al., "FVC2000 : Fingerprint Verification Competition," PATTERN ANALYSIS AND MACHINE INTELLIGENCE, vol. 24, pp. 402-412, MARCH 2002.
[82] D. Maio, et al., "FVC2002: Second Fingerprint Verification Competition," presented at the IEEE, 2002.
[83] D. Maio , et al., "bias.csr.unibo.it/fvc2004/databases.asp," Aprill,2003.
[84] D. Maio , et al., "bias.csr.unibo.it/fvc2004/results.asp," Aprill,2003.
[85] سبحان آذین فر، محسن شفیعی" ,پیاده سازی و بررسی روشهای متن کاوی ",دانشگاه پیام نور مشهد ,تیر ماه ۱۳۸۹.

واژه نامه انگلیسی به فارسی

Artificial Neural Network	شبکه عصبی مصنوعی
Accuracy	دقت
Artificial Life	حیات مصنوعی
Ant Colony	کلونی مورچگان
Aspiration Criterion	معیار تنفس
Binary Particle Swarm optimization	بهینه سازی توده ذرات باینری
Backtracking	بازگشت به عقب
Birds flock	دسته پرندگان
Best Value	بهترین مقدار
Classification	طبقه بندی
Complet Search	جستجوی کامل
Criterion function	تابع معیار
Correlation	همبستگی
Coefficient	ضریب
Consistency Measures	معیار مبتنی بر سازگاری
Classifier Error Rate Measures	معیار مبتنی بر خطای طبقه بندی
Crossover	ترکیب
Chromosom	کروموزوم
Characteristic loci	مکان مشخصه توسعه یافته
Crossover Mask	ماسک ترکیب
Discrete Wavelet Transform	تبدیل موجک گسسته
Discrete Fourier Transform	تبدیل فوریه گسسته
Discrete Multi wavelet Transform	تبدیل موجک گسسته چند مرحله ای
Distance Measures	معیار مبتنی بر فاصله
Dependence Measures	معیار مبتنی بر وابستگی
Deoxyribonucleic acid	دی ان ای
Data Clustering	خوشه بندی داده
Evaluation function	تابع ارزیابی
Exhaustive search	جستجوی کامل
Explorative	اکتشافی
Feature	ویژگی
Feature Reduction	کاهش ویژگی
Feature Extraction	استخراج ویژگی
Fitness function	تابع برازندگی
Feature selection	انتخاب ویژگی
Factor Analysis	فاکتور آنالیز
Fast Fourier Transform	تبدیل فوریه سریع
Fish School	گروه ماهی ها
Fittness Value	مقدار برازنگی
Fixed length	طول ثابت
Fuzzy System Control	کنترل سیستم فازی
Face Recognation	بازشناسی چهره
False Match Rate	تطابق اشتباه
False NonMatch Rate	عدم تطابق اشتباه
Genetic Algorithm	الگوریتم ژنتیک
Global flat subspace	زیر فضای تخت عمومی
General Intelligence	هوش مصنوعی
Generation procedure	تابع تولید کننده
Generality	عمومیت
Gene	ژن
Global best	بهترین سراسری
Gradient Histogram	هیستوگرام گرادیان
Heuristic Search	جستجوی اکتشافی
Handwriting digits	ارقام دستنویس

Hyperplane	ابر صفحه
Hilbert Transform	تبدیل هیلبرت
Information Measures	معیار مبتنی بر اطلاعات
Individual	فرد
Initial Population	جمعیت اولیه
Inertia weight	وزن اینرسی
Iteration	تکرار
Independent Component Analysis	آنالیز المانهای مستقل
K-nearest neighbor	نزدیکترین همسایگی
Locally flat subspace	زیر فضای تخت محلی
Local best	بهترین محلی
Learning factors	فاکتور یادگیری
Local Optima	بهینه محلی
Local Search	جستجوی محلی
Machine learning	یادگیری ماشین
Multi-Layer Perceptron	پرسپترون چند لایه
Mutation	جهش
Methodology	اصلوب شناسی
Multi dimensionl	چند بعدی
Neural Network	شبکه عصبی
Optical character recognition	بازشناسی نوری حروف
Observations	مشاهدات
Optimum Solution	پاسخ بهینه
Optimal Separating hyperplane	ابر صفحه جداساز بهینه
Pattern Recognition	بازشناسی الگو
Processing	پردازش
Particle Swarm Optimization	بهینه سازی توده ذرات
Pattern	الگو
Principal Component Analysis	تحلیل مولفه های اصلی
Projection Pursuit	تعقیب افکنش
Premature convergence	همگرایی زودرس
Position	موقعیت
Particles	ذرات
Quadratic assignment problem	تخصیص منشور قائم
Real-time	بلادرنگ
Recognition Rate	نرخ بازشناسی
Random Search	جستجوی تصادفی
Rotation	چرخش
Random Projection	افکنش تصادفی
Regression	رگرسیون
Support Vector Machines	ماشین بردار پشتیبان
Storage	ذخیره سازی
Singular Value Decomposition	تجزیه مقدارهای تکین یا منفرد
Seeding	تولید نسل
Selection	انتخاب
Swarm Intelligence	هوش دسته جمعی
Stop criterion	معیار توقف
Swarm Size	اندازه توده
Short Term Memory	حافظه کوتاه مدت
Sobel - Raberts	سوبل – رابرتز
Segmentation	قطعه قطعه سازی
Tabu Search	جستجوی ممنوعه
Training Phase	فاز آموزش
Testing Phase	فاز تست
Tabu List	لیست ممنوعه
Travelling salesman problem	فروشنده دوره گرد
Update	بروزرسانی
Variable	متغیر
Validation procedure	تابع تعیین اعتبار

Abstract:

In recent years there has been a lot of attention in Feature selection. Therefore feature Selection and reduction of data dimension is very important in Pattern recognition. In real-time systems, feature reduction increases the speed of processing and needs less memory to store information. In the context of optimization algorithms, especially Particle swarm optimization (PSO) has been considerable. So that using PSO and choosing the appropriate method to calculate the PSO fitness can be reduced a significant amount of feature vectors to achieve acceptable results efficiently.

In this thesis the features of Persian handwritten digits are extracted by combining the gradient histogram and characteristic loci methods. Also, we proposed the Binary particle swarm optimization (BPSO) algorithm by appropriate fitness function to select the important features. In order to digit recognition the features that selected in the proposed method classified by Support Vector Machine (SVM). The proposed method applied to HODA database. The results are shown 99.40% accuracy without reducing the features (400 features) and 99.11% accuracy with reducing the features (193 features). Also, ORL face database, FVC2004 Fingerprint database and UCI database are used to verify the performance of system. In comparison to other research results we show that the proposed method has a good performance in feature selection.

Keywords: BPSO, Support Vector Machines, Recognition of Persian handwritten digits, Gradient Histogram, Characteristic loci, Binary Particle Swarm Optimization

Shahrood University of Technology

Feature Reduction using Particle Swarm Optimization algorithm and SVM Classifier

Thesis

Submitted in Partial Fulfillment of the

Requirements for the Degree of Master of Science (M.Sc.)

in Electronic Engineering

Department of Electrical Engineering and Robotics

Shahrood University of Technology

By:

Mohsen Sedighi Nav

Supervisor:

Dr. Ali Soleimani

Dr. Hossein Khosravi

Advisor:

Dr. Alireza Ahmadifard

Summer 2012