دانشکده مهندسی برق و رباتیک

پایان‌نامه کارشناسی ارشد

استفاده از ترکیب کلاسه بندها و انتخاب ویژگیهای مناسب جهت بهبود دقت تشخیص هویت نویسنده از روی دستخط

وحید زارعی

استاد راهنما:

دکتر امیدرضا معروضی

استاد مشاور:

دکتر حسین خسروی

شهریور سال ۱۳۹۱


بسم الله الرحمن الرحيم

شماره : ۰۹۴۱/آ.ت.ب
تاریخ : ۹۱/۰۶/۲۸
ویرایش : ـــــ

بسمه تعالی

مدیریت تحصیلات تکمیلی
فرم شماره (۶)

فرم صورتجلسه دفاع پایان نامه تحصیلی دوره کارشناسی ارشد

با تأییدات خداوند متعال و با استعانت از حضرت ولی عصر (عج) جلسه دفاع از پایان نامه کارشناسی ارشد خانم / آقای :

وحید زارعی رشته : برق گرایش : الکترونیک

تحت عنوان : استفاده از ترکیب کلاسه بندها و انتخاب ویژگیهای مناسب جهت بهبود وقت تشخیص هویت نویسنده

که در تاریخ ۹۱/۰۶/۲۸ با حضور هیأت محترم داوران در دانشگاه صنعتی شاهرود برگزار گردید به شرح زیر است :

قبول (با درجه : عالی امتیاز ۱۹٫۱۷) ☑ دفاع مجدد ☐ مردود ☐

۱ـ عالی (۲۰ ـ ۱۹)
۲ـ بسیار خوب (۱۸/۹۹ ـ ۱۸)
۳ـ خوب (۱۷/۹۹ ـ ۱۶)
۴ـ قابل قبول (۱۵/۹۹ ـ ۱۴)
۵- نمره کمتر از ۱۴ غیر قابل قبول

عضو هیأت داوران	نام ونام خانوادگی	مرتبهٔ علمی	امضـــاء
۱ـ استاد راهنما	امید رضا سروش	استادیار	
۲ـ استاد مشاور	حسین حسنوی	استادیار	
۳ـ نماینده شورای تحصیلات تکمیلی	مرتضی رحیمیان	استادیار	
۴ـ استاد ممتحن	حسن کرمی	استادیار	
۵ـ استاد ممتحن	محمد رضا پورنوری	استادیار	

رئیس دانشکده :

تقدیم به مادر فداکارم اسوه صبر و شکیبایی ناپذیری در زندگیم و تقدیم به پدر مهربانم.

و

تقدیم به کسی که محبت و وجودش برایم امید و روشنی مسیر زندگی است.

تحست خداوند بزرگ و علیم راشاکرم که به من قدرت تفکر و نوشتن دادو تلاش رادر وجودم برای یافتن پاسخ به معماهایی هر چند کوچک از علم قرار داد. سپاس که توانستم با دانش اندک خود گامی در مسیر علم بردارم.

بر خود فرض می دانم که از استاد راهنمای فرهیخته و دانشمند خویش آقای دکتر امیدرضا معروضی و استادمشاور فرزانه ام و دانشمند توانمند جناب آقای دکتر حسین خسروی کمال تشکر و قدردانی رابجا آورم. در برابر لطف و توجه بزرگوارانه شان زانوی ادب بر زمین مینهم.

و نیز از خانواده خود و همه دوستانم که مرا در این کار یاری نمودند کمال تشکر رابجا می آورم. از دوستان خویش آقایان مهندس مسعود دهقان، مهندس علی قنبری سرخی، مهندس صالح قرائی و مهندس محسن اکرمی به طور خاص کمال تشکر رادارم.

تعهـد نامـه

اینجانب ..... دانشجوی دوره کارشناسی ارشد رشته ..... دانشکده مهندسی برق و رباتیک دانشگاه صنعتی شاهرود نویسنده پایان نامه با عنوان :

.....

تحت راهنمائی آقای دکتر ..... متعهد می شوم :

- تحقیقات در این پایان نامه توسط اینجانب انجام شده است و از صحت و اصالت برخوردار است .
- در استفاده از نتایج پژوهشهای محققان دیگر به مرجع مورد استفاده استناد شده است .
- مطالب مندرج در پایان نامه تاکنون توسط خود یا فرد دیگری برای دریافت هیچ نوع مدرک یا امتیازی در هیچ جا ارائه نشده است .
- کلیه حقوق معنوی این اثر متعلق به دانشگاه صنعتی شاهرود می باشد و مقالات مستخرج با نام « **دانشـگاه صـنعتی شـاهرود** » و یـا « Shahrood University of Technology » به چاپ خواهد رسید .
- حقوق معنوی تمام افرادی که در به دست آمدن نتایج اصلی پایان نامه تأثیرگذار بوده اند در مقالات مستخرج از پایـان نامـه رعایـت مـی گردد.
- در کلیـه مراحل انجام این پایان نامـه، در مواردی که از موجـود زنـده (یا بافت های آنها) استفاده شده است ضوابط و اصـول اخـلاقی رعایت شده است.
- در کلیه مراحل انجام این پایان نامه، در مواردی که به حوزه اطلاعات شخصی افراد دسترسی یافته یا استفاده شده است اصل رازداری ، ضوابط و اصول اخلاق انسانی رعایت شده است .

تاریخ : ۹۱/۶/۲۸

امضاء دانشجو

مالکیت نتایج و حق نشر

- کلیه حقوق معنوی این اثر و محصولات آن (مقالات مستخرج ، کتاب ، برنامه های رایانه ای ، نرم افزار ها و تجهیزات ساخته شده است) متعلق به دانشگاه صنعتی شاهرود می باشد . این مطلب باید به نحو مقتضی در تولیدات علمی مربوطه ذکر شود .
- استفاده از اطلاعات و نتایج موجود در پایان نامه بدون ذکر مرجع مجاز نمی باشد.

*** متن این صفحه نیز باید در ابتدای نسخه های تکثیر شده پایان نامه وجود داشته باشد .**

چکیده

در این تحقیق روشی نو برای تعیین هویت نویسنده بر اساس متن دستنویس فارسی بصورت برون خط و مستقل از متن بر مبنای اختلاط خبره ها ارائه شده است. در روش پیشنهادی از فیلترهای گابور، سوبل و روبرتز برای استخراج ویژگی استفاده شده است. سپس با استفاده از روش تجزیه و تحلیل مولفه‌های اصلی ،PCA ، و با کمک الگوریتم نزدیکترین همسایه کاهش ویژگی انجام گرفته است. برای ارزیابی دقت تشخیص به کمک ویژگی‌های استخراج شده ازسه شبکه عصبی MLP جهت دسته بندی استفاده شده است که خروجی آنها را با هم ترکیب کرده ایم. هر کدام از این شبکه‌ها برای متفاوت شدن در خطا بر روی یک مجموعه از ویژگی‌ها آموزش داده شده‌اند، و سپس در سه آزمایش نخست در این تحقیق به هر کدام از شبکه‌ها یک وزن داده شده است و در نهایت خروجی شبکه ها با هم ترکیب گردیده است.که در آزمایش اول به نظر هر کدام از طبقه-بندهای پایه وزن یکسان داده شده است، در آزمایش دوم وزن‌ها بر اساس میزان دقت هر طبقه‌بند تعیین گردیده است، در آزمایش سوم وزنهای هر یک از شبکه‌ها توسط الگوریتم PSO به منظور کمینه شدن خطا در مرحله آموزش تعیین گردیده است و در آزمایش چهارم در این تحقیق به هر کدام از نرونهای خروجی هر کدام از شبکه‌های عصبی وزن داده شده است که این وزن‌ها با استفاده از الگوریتم PSO همانند آزمایش سوم تعیین گردیده است. جهت آموزش شبکه عصبی از گرادیان نزولی و روش پس انتشار خطا استفاده شده است. نتایج نشان دهنده دقت هویت نویسنده از دستخط فارسی به ترتیب ۹۷,۴۲٪ ، ۹۷,۷۵٪ ، ۹۷,۵۹٪ و ۹۸,۱۷٪ می‌باشد که دقت بالایی در زمینه تعیین هویت نویسنده می‌باشد. و همچنین از نظر زمان آموزش شبکه مرکب به ترتیب برای آزمایش اول تا چهارم مقادیر ۲۴۵۰,۵۴، ۲۴۵۰,۵۴، ۲۵۶۲,۰۴ و۲۵۹۴,۵۹ ثانیه بعد از کاهش ویژگی حاصل گردیده است که نسبت به زمان به دست آمده قبل از کاهش ویژگی به ترتیب ۶۹۵,۴۶، ۶۹۵,۴۶ ، ۶۸۳,۱ و ۶۵۴,۲۲ ثانیه زمان آموزش شبکه مرکب کاهش یافته است که زمان قابل توجهی می‌باشد.

کلمات کلیدی: تعیین هویت نویسنده، ترکیب طبقه بندها، اختلاط خبره ها، کاهش ویژگی، شـبکه عصـبی، دست‌خط

مقالات استخراج شده

[۱] زارعی وحید، معروضی امیدرضا، خسروی حسین و علی قنبری، ۱۳۹۱، تعیین هویت نویسنده با استفاده از اختلاط خبره‌ها(**ME**) و به کارگیری الگوریتم **PCA** جهت کاهش ویژگی‌ها در متون دستنویس فارسی، اولین کنفرانس بین‌المللی پردازش خط و زبان فارسی، سمنان، ایران

[۲] مرتضوی احسان، زارعی وحید، خسروی حسین و معروضی امیدرضا، ۱۳۹۱، بازشناسی قلم‌های فارسی براساس ترکیب ویژگی‌ها و استفاده از **ME**، ارائه شده به مجله انجمن ماشین بینایی و پردازش تصویر ایران، تهران، ایران

[3] Zareii, V; marozi, O.R and Khosravi H. (2012). Use ME for writer identification based on Robert-Sable and Gabor filter in Persian handwriting.

فهرست مطالب

فهرست شکلها

فهرست جدول‌ها

فصل اول:

مقدمه

۱-۱ تشخیص دست‌خط فارسی

تعیین و تایید هویت بر اساس ویژگیهای بیومتریک، از زمینه‌های تحقیقاتی فعال است که در دههء اخیر مورد توجه فراوان قرار گرفته است [۱]. و در زمینه های امنیتی، حقوقی، کنترل دسترسی به سیستم ها و فعالیت های مالی کاربرد گسترده ای دارد. ویژگی‌های بیومتریک انسان به دو دسته ویژگی‌های فیزیولوژی و رفتاری تقسیم می‌گردند، که دستخط افراد جزء ویژگی‌های رفتاری محسوب می‌گردد. مطالعات نشان می‌دهد که افراد مختلف دستخط‌های متفاوتی دارند [۱]. برای این منظور از دو روش کلی برای شناسایی استفاده می‌شود: روش برون خط و روش برخط.

در روش برون‌خط[1] فقط تصویر متن دستنویس در دسترس است و ویژگی‌ها با توجه به ساختار کلمه‌ها و نویسه‌ها استخراج می‌گردند. در این روش تعداد زیادی از اطلاعات دینامیکی مربوط به طرز نوشتن افراد، که در روش برخط[2] در دسترس است ازدست می‌رود و به همین دلیل، نسبت به روش برخط دقت کمتری دارد [۱]. همچنین روش برون‌خط، خود به دو گروه کلی مستقل از متن و وابسته به متن تقسیم‌بندی می‌شود. در روش وابسته به متن باید متنی که توسط نویسندگان مختلف ارائه می‌گردد ثابت باشد در حالیکه در روشهای مستقل از متن، نیازی به یکسان بودن متن نویسندگان مختلف نیست و فقط نمونه‌های هر نویسنده متن یکسان دارند.

روش دوم، روش برخط است که در این روش علاوه بر ویژگی‌های برون‌خط از اطلاعات دینامیکی مثل فشار قلم، ترتیب نوشتن، سرعت نوشتن، فرم ضربه‌های قلم و ... نیز استفاده می‌شود. هر چند دقت این روش بیشتر است ولی بخاطر نیاز به سخت افزار ویژه کاربرد آنها نسبت به روش برون‌خط کمتر است.

[1] off line
[2] on line

۱-۲ بیان مساله

برای تعیین هویت بر اساس دستخط فارسی در دهه اخیر تحقیق‌هایی صورت گرفته که از نظر دقت در مرحله کلاس‌بندی می‌توان نتایج بهتری نسبت به آنها به دست آورد. از جمله این تحقیق‌ها می‌توان به تحقیق آقای دهقان [۲] اشاره کرد که اساس کار این تحقیق می‌باشد. و جهت بهبود دقت در تشخیص هویت افراد ترکیب طبقه بندها و انتخاب ویژگی های مناسب معرفی می گردد که ما جهت افزایش دقت و سرعت جهت انجام پروسه تشخیص هویت افراد از آنها استفاده شده است.

۱-۳ ترکیب طبقه‌بندها

طبقه‌بندی معمولا در یک فرایند یادگیری ساخته می شود. بسیاری از الگوریتم های یادگیری در حقیقت یک نوع جستجوی محلی انجام می دهند که ممکن است در کمینه های محلی گرفتار شوند. در صورت گرفتار شدن در کمینه های محلی نمی‌توان طبقه‌بند بهینه داشت. اجرای الگوریتم‌های یادگیری تحت شرایط متفاوت، یادگیری دسته جمعی[1] ، روشی است برای آنکه بتوان تقریب بهتری را از یک طبقه بند بهینه فراهم کرد. در یادگیری دسته جمعی، هر الگوریتم یادگیری با توجه به مقدار پارامترهایش، به پاسخ متفاوتی برای مساله می رسد و انتظار می رود با ترکیب این پاسخ ها، دقت طبقه بندی افزایش یابد. به همین علت، در سالهای اخیر استفاده از نتایج چند طبقه بند، به عنوان یک رویکرد موثر در بازشناسی الگو، توجه محققین زیادی را به خود جلب کرده است. و از آن در شاخه های مختلف علوم استفاده شده است. تشخیص نفوذ در شبکه های کامپیوتری، شناسایی کد پستی، بازشناسی دست‌نوشته، تشخیص هویت وشناسایی گوینده نمونه هایی از کاربردهای ترکیب طبقه بندها هستند [۳].

[1] Ensemble (Committee) Learning

به طبقه‌بندهایی که نتایج آنها با هم ترکیب می شوند، طبقه بند های پایه[1] و به مجموعه طبقه بند ها، سیستم مرکب گفته می شود. شبکه های عصبی از متداولترین طبقه بند های پایه هستند [٤].

دو طبقه بند، زمانی گوناگونی در خطا دارند که الگوهایی که به صورت نادرست طبقه بندی می‌کنند متفاوت باشد. نتایج تئوری و تجربی نشان می دهند که زمانی یادگیری دسته جمعی از یادگیری بهترین طبقه بند پایه بهتر است که طبقه بندهای پایه ضمن برخورداری از کارایی قابل قبول، گوناگون[2] در خطا بوده و قاعده مناسبی برای ترکیب نتایج آنها به کار گرفته شود. تفاوت در موارد خطای طبقه بندهای پایه، باعث می شود که طبقه بند ها خطای یکدیگر را بپوشانند. به همین علت گوناگونی در خطا از نکات اساسی در موفقیت سیستم‌های طبقه بندی مرکب است. برای ایجاد گوناگونی در خطا روشهای متعددی پیشنهاد شده است. این روشها به دو دسته ضمنی[3] و صریح[4] تقسیم می شوند [۴].

روشهای ضمنی، با تغییرات ضمنی در فرایند یادگیری طبقه‌بندهای پایه، سعی در گوناگون کردن آنها دارند. استفاده از مجموعه های یادگیری متفاوت، ساختارهای مختلف، بازنمایی های متفاوت برای بیان الگو و افزودن نویز به داده‌های ورودی از جمله روشهای ضمنی هستند. متداولترین این روشها، انتخاب تصادفی همراه با جایگزینی نمونه‌ها از بین کلیه نمونه‌های آموزشی است که روش بسته‌بندی[5] نامیده می‌شود [۴].

روشهای صریح، با تحت تاثیر قرار دادن مسیر یادگیری طبقه‌بندهای پایه، آنها را با یکدیگر گوناگون می سازند. این روشها در فرایند یادگیری طبقه‌بندها، معیاری از گوناگونی را اعمال کرده و بر اساس آن مسیر یادگیری طبقه‌بندها را در فضای یادگیری طوری تغییر می‌دهند که طبقه‌بندها گوناگون شوند. روشهای تقویتی[6]

[1] Base Classifiers
[2] Diverse
[3] Implicit
[4] Explicit
[5] Bagging Methods
[6] Boosting Methods

و روشهای جریمه‌ای[1] از مهمترین روشهای صریح برای ایجاد گوناگونی در طبقه‌بندهای پایه هستند. در روش های تقویتی، توزیع داده‌های آموزشی هر طبقه‌بند بر اساس خطاهای طبقه‌بند مرحله قبل تعیین می شود. روشهای جریمه‌ای، با به کار گیری یک مولفه جریمه در تابع خطای طبقه‌بندهای پایه، به ایجاد گوناگونی در آنها می‌پردازند [۴].

۱-۴ کاهش ویژگی و انتخاب ویژگی‌های مناسب

مساله انتخاب و کاهش ویژگی، یکی از مسائلی است که در مبحث یادگیری ماشین و همچنین شناسائی آماری الگو مطرح است. این مساله در بسیاری از کاربردها مانند طبقه‌بندی داده 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. و علاوه بر این باعث می‌شود که اطلاعات غیر مفید زیادی را به همراه داده‌های مفید ذخیره کنیم.

برای مساله انتخاب ویژگی، راه حل‌ها و الگوریتم‌های فراوانی ارائه شده است که بعضی از آنها قدمت سی یا چهل ساله دارند [۵]. مشکل بعضی از الگوریتم‌ها در زمانی که ارائه شده بودند، بار محاسباتی زیاد آنها بود، اگر چه امروزه با ظهور کامپیوترهای سریع و منابع ذخیره‌سازی 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 این تحقیق روشهای انتخاب و کاهش ویژگی مورد بررسی قرار می گیرد و برای کاهش تعداد ویژگی‌های استخراج شده از تصاویر دستنوشته‌ها جهت تعیین هویت از یکی از این روش‌ها استفاده خواهیم کرد.

[1] Penalty Methods

۱-۵ مروری بر کارهای گذشته

در این بخش مروری بر کارهایی که در این زمینه انجام شده خواهیم کرد و چند مورد از تحقیقهایی که اهمیت بیشتری داشته‌اند را به طور مختصر بررسی می‌کنیم.

Srihari تعداد زیادی ویژگی را در دو دسته کلی پیشنهاد نموده است [۱, ۶]. ویژگی‌های بزرگ و کلی که در سطح کل متن، پاراگراف و کلمه استخراج می‌شوند شامل: آنتروپی در تصویر خاکستری، تعداد کل پیکسلهای متن، تعداد کانتورهای درونی و بیرونی متن، تعداد اجزا متصل متن در چهار جهت اصلی، متوسط ارتفاع و کجی در متن، نسبت وجود پاراگراف به فضای خالی در متن، طول لغات و نسبت بالا به پایین محور اصلی خط. دومین سری ویژگی‌ها، ویژگی‌های جزئی و ریز در سطح کلمه و حرف شامل: گرادیان، ساختار و شکل تقعر لغات و حروف می‌باشد. برای ارزیابی ویژگی‌های استخراج شده، پایگاه داده بزرگی شامل ۱۰۰۰ آمریکایی که هر کدام یک متن ۱۵۶ کلمه‌ای را ۳ مرتبه بر روی کاغذ نوشتند جمع‌آوری شد. این پایگاه داده، بزرگترین پایگاه داده جمع‌آوری شده در زمینه تعیین هویت نویسنده است. ویژگی‌های جزئی با دقت بیش از ۸۰٪ ، بهتر از ویژگی-های کلی نتیجه‌بخش بودند. در ادامه همین تحقیق تعیین هویت با استفاده از برخی حروف خاص [۷, ۸] و لغات خاص [۱, ۹] با همین پایگاه داده انجام شد.

Said دیدگاهی را معرفی کرده که در آن متن دستنوشته بصورت بافت در نظر گرفته می‌شود [۱۰, ۱۱]. در این روش از فیلترهای گابور چند کاناله و ماتریس هم‌رخداد در سطح خاکستری، برای استخراج ویژگی استفاده شده است. در این روش احتیاج به وجود بلوکهای منظمی از متن دستنوشته می‌باشد. به همین دلیل ابتدا باید متن را نسبت به تاثیر عواملی مثل فاصله خطوط و کلمات نرمال کرد. برای ارزیابی این تحقیق از دو گروه ۲۰ نفری و از هر گروه ۲۵ نمونه دستنوشته استفاده شده است. با استفاده از طبقه‌بند فاصله اقلیدسی وزن‌دار دقت

۹۶٪ در این روش حاصل گردید. tan از دیدگاه‌هایی مشابه برای تشخیص نوع زبان در متون چاپی استفاده کرده است [۱۲, ۱۳]. و zhu نیز برای تشخیص فونت از روشهایی مشابه با این دیدگاه بهره برده است [۱۴].

Zois و Anastassopoulos تنها از یک لغت برای تشخیص نویسنده استفاده کرده‌اند. آنها در آزمایش خود از دستخط ۵۰ نفر که هر کدام لغت "characteristic" را ۴۵ مرتبه و به دو زبان انگلیسی و یونانی نوشته بودند استفاده کرده‌اند. بعد از آستانه‌گیری از تصویر و نازک‌سازی آن، نمای نیم‌رخ افقی تصویر محاسبه شده است. این منحنی به ۱۰ قسمت تقسیم، و سپس با استفاده از عملگرهای مورفولوژی با دو اندازه متفاوت، پردازش گردید. در نهایت بردارهای ویژگی ۲۰ بعدی، از این منحنی استخراج شدند. این بردارهای ویژگی با استفاده از طبقه‌بند بیزین طبقه‌بندی شد و نتایج برای هر دو زبان انگلیسی و یونانی نزدیک به ۹۵٪ حاصل شد [۱۵].

Bensefia از گرافم‌ها برای تعیین هویت استفاده کرد. این گرافم‌ها از بخش‌بندی متن که برای آشکارسازی کاراکترهای خاص، از متن دستنوشته و بصورت مستقل از متن تولید می‌شود، هستند. برای تعریف فضای ویژگی در کل متن‌ها از بخش‌بندی گرافم‌ها استفاده شد. این روش بر روی سه دسته پایگاه داده، شامل یک دسته ۸۸ نفری و یک دسته ۳۳ نفری که شامل متنهای تاریخی بودند و یک دسته ۱۵۰ نفری آزمایش گردید. در این تحقیقات برای شناسایی نویسنده از بازیابی اطلاعات استفاده شد. در این تحقیقات دقت ۹۰٪ در تعیین هویت نویسنده گزارش شده است [۱۶, ۱۷].

Marti، Hertel و Bunke با استفاده از خطوط متن دستنوشته و بصورت مستقل از متن ویژگی‌های متنوعی را معرفی کردند. ویژگی‌هایی شامل: ارتفاع سه قسمت اصلی خط، کجی و پهنای کاراکترها، فاصله بین اجزا متصل متن، مساحت دایره بسته کشیده شده دور متن، کانتورهای بالایی و پایینی متن و.. [۱۸, ۱۹].

Sclapbach در مطالعات خود به موضوع انتخاب این ویژگی‌ها پرداخت و با استفاده از طبقه‌بند نزدیکترین همسایگی ویژگی‌ها را طبقه‌بندی کرد. او در این تحقیق از پایگاه داده IAM شامل ۵۰ نویسنده و از هر کدام ۵ نمونه دستنوشته استفاده کرد [۲۰].

Sclapbach و Bunke از مدل مخفی مارکوف (HMM[1]) بر مبنای بازشناخت دستنوشته، برای شناسایی فرد استفاده کردند. سیستم بازشناخت برای هر شخص مخصوص به خود اوست و با نمونه‌های دستنوشته آن شخص آموزش داده می‌شود. در این روش از امتیازهای log-likelihood خروجی‌های HMM ها برای شناسایی استفاده شد. در ضمن این روش یک روش مستقل از متن می‌باشد. برای ارزیابی این روش نیز از زیرمجموعه‌ای از پایگاه داده IAM شامل ۱۰۰ نویسنده و از هرکدام ۵ نمونه استفاده و دقتی نزدیک به ۹۶٪ حاصل شد [۲۱, ۲۲].

Bulacu و Schomaker یک روش در آنالیز بافت و بصورت مستقل از متن معرفی نمودند که در آن از محل اتصال تابع چگالی احتمال زاویه ساقه‌ها، که از آشکارسازی لبه به دست آمده بود استفاده شد. ارزیابی این روش نشان از مفید بودن این ویژگی در قیاس با ویژگی‌های دیگر داشت. در ادامه، اطلاعات مربوط به موقعیت مکانی دستنوشته نیز به این ویژگی‌ها اضافه شد به این صورت که ابتدا متن دستنوشته از وسط به دو قسمت تقسیم و سپس برای هر قسمت تابع چگالی احتمال محاسبه گردید و در نهایت دوباره این توابع به هم متصل شدند. همچنین به نظر آنها رفتار آماری هر نویسنده، به وسیله شکل نوشته‌ها و گرافم‌ها تخمین زده می‌شود و این نظر مبنای ویژگی‌های آلوگراف[2] آنهاست. در نهایت تمام این ویژگی‌ها را با استفاده از یک پایگاه داده بزرگ شامل ۹۰۰ نویسنده و از هر کدام ۲ متن، ارزیابی کرده‌اند و نتایج نشان دهنده دقت ۸۶٪ در این روش است [۲۳-۲۶].

[1] HMM: Hidden markov model

[2] تغییرات متداول در گرافم

در زمینه تشخیص هویت با استفاده از دستخط فارسی تحقیق زیادی صورت نگرفته و چندین تحقیق که در زبان فارسی و عربی که شباهت زیادی با زبان فارسی دارد در ادامه آورده شده است.

الدمور و همکارش از اطلاعات لبه جهت تشخیص هویت نویسنده در زبان عربی که شباهت زیادی به دستنویس فارسی دارد استفاده نموده اند [۲۷].

شهابی‌نژاد فیلترهای گابور را برای استخراج ویژگیها از متن دستنویس فارسی به کار گرفت [۲۸]. صادقی و همکارش از ویژگیهای مبتنی بر گرادیان برای آموزش شبکه عصبی چندلایه استفاده کرده‌اند [۲۹]. دهقان و همکاران یک پایگاه داده شامل متن دستنویس ۵۰ نویسنده تهیه کرده‌اند که از هر نویسنده ۱۰ نمونه از متن دلخواه در آن وجود دارد. در تحقیق ایشان از فیلتر های سوبل و روبرتز جهت استخراج ویژگی و از طبقه‌بند نزدیکترین همسایگی K-NN[1] به منظور تعیین هویت نویسنده استفاده شده است [۲]. تعداد ویژگیها استخراج شده در این تحقیق ۶۴۰ ویژگی برای هر یک از خروجیهای فیلتر سوبل و روبرتز است.

در این گزارش ابتدا به معرفی کلیات مطالب در مورد انتخاب و ترکیب طبقه بندها در فصل ۲ می پردازیم سپس در فصل ۳ روشها ی انتخاب و کاهش ویژگی بیان میگردد و در فصل ۴ روش پیشنهادی در این تحقیق و آزمایش‌های انجام شده توضیح داده شده و در فصل پنجم نتیجه‌گیری و مقایسه صورت گرفته است.

۱-۶ ساختار پایان نامه

در این پایان نامه در فصل بعد مروری بر روش‌های انتخاب طبقه بند و ترکیب طبقه‌بند ها خواهیم داشت و انواع روشهای ترکیب طبقه‌بندها و انتخاب طبقه‌بندها را توضیح خواهیم داد همچنین برخی از نقاط قوت و ضعف های هر کدام از الگوریتم‌های ترکیب یا انتخاب طبقه‌بندها بیان خواهد شد. در فصل سوم انواع روشهای انتخاب و کاهش ویژگی توضیح داده شده است و الگوریتم‌های مختلفی در زمینه انتخاب و کاهش ویژگی بیان

[1] K Nearest Neighbor

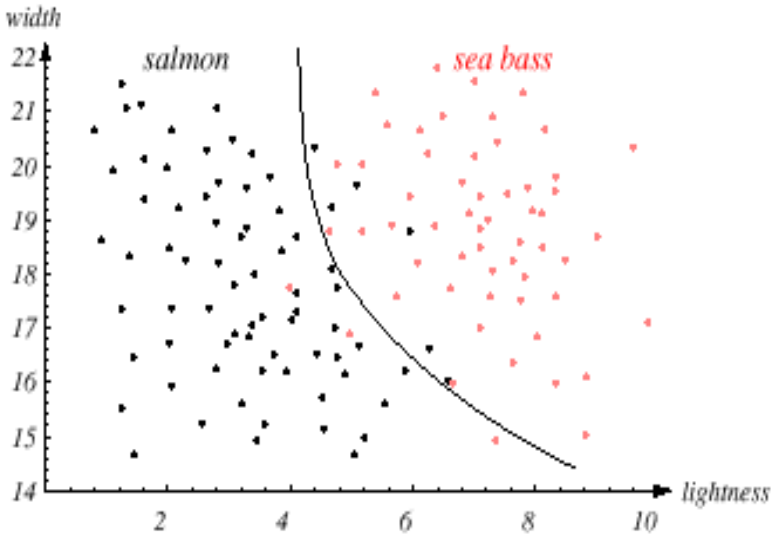
گردیده است و محدودیت و کارامدی برخی از آنها بیان گردیده است. در فصل چهارم روش پیشنهادی در این تحقیق و آزمایش‌های انجام شده توضیح داده شده و نتایج هر آزمایش که شامل دقت آزمایش‌ها و زمان اجرای الگوریتم می‌باشد به صورت جدول نمایش داده شده است. در فصل پنجم نتیجه‌گیری و مقایسه صورت گرفته است

فصل دوم:

مروری بر روشهای انتخاب و ترکیب طبقه‌بندها

۲-۱ تعاریف و مفاهیم

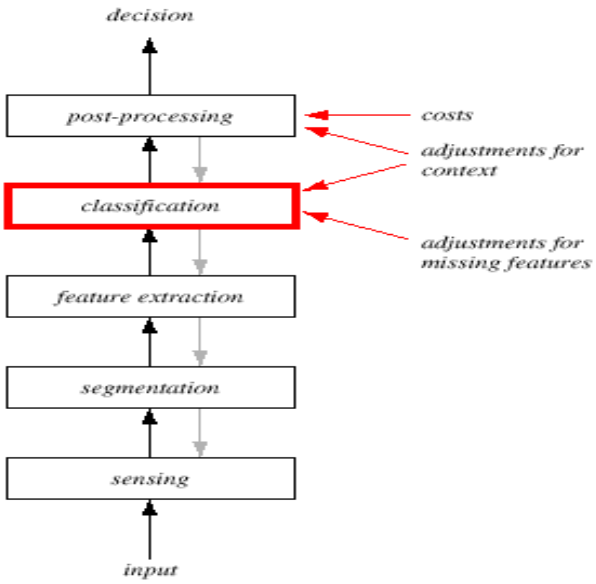
طبقه بندی: طبقه بندی فرایندی است که در آن هر الگوی ناشناخته بر اساس ویژگی های آن، به یکی از کلاسهای شناخته شده نسبت داده می شود. به عبارت دیگر، طبقه بندی یک نگاشت از فضای n بعدی ویژگی ها به فضای K بعدی برچسب‌های کلاسی است که در آن میزان تعلق بردار ویژگی $X \in R^n$ به کلاسهای مختلف، به صورت یک مقدار عددی بیان می شود. به عنوان مثال درشکل ۲-۱ دو نوع ماهی با توجه به ویژگی عرض و روشنایی در دو کلاس طبقه بندی شده اند. در شکل ۲-۲ مراحل عملیات طبقه‌بندی به صورت بلوک دیاگرام نمایش داده شده است.



شکل۲-۱:کلاس بندی ماهی ها بر اساس عرض و روشنایی [۳۰]

خبره: خبره یا طبقه‌بند پایه به طبقه‌بندهایی گفته می‌شود که ما می‌خواهیم آنها را با هم ترکیب کنیم

شایستگی طبقه‌بند: بیان کننده میزان دقت طبقه‌بند می باشد و بر اساس میزان شایستگی هر طبقه بند، نظر طبقه بند مورد اعتماد می باشد.



شکل ۲-۲: مراحل طبقه بندی الگو [۳۰]

۲-۲ مقدمه

در انتخاب طبقه‌بند فرض بر این است که یک راه حل وجود دارد که بتوان بهترین خبره را برای یک ورودی خاص x شناسایی کرد. تصمیم این خبره برای تصمیم مجموعه خبره‌ها برای x قابل قبول است.

ایده استفاده از طبقه‌بندهای متفاوت برای ورودی‌های متفاوت توسط داساراتی[۱] و شیلا[۲] در سال ۱۹۷۸ ارائه شد [۳۱]. آنها یک طبقه‌بند خطی و یک طبقه‌بند نزدیکترین همسایه‌ها (k-nn) را با هم ترکیب کردند. طبقه‌بند مرکب یک حوزه تناقض در فضای ویژگی‌ها شناسایی کرد و از knn در آن حوزه استفاده کرد، در حالی که همزمان طبقه‌بند خطی در حوزه دیگری مورد استفاده قرار می‌گرفت. در سال ۱۹۸۱ راسترینگ[۳] و ارنستین[۴] [۳۲] یک روش شناسایی برای انتخاب طبقه‌بند ارائه دادند که به روشهایی که اکنون استفاده می‌گردد نزدیک می‌باشد.

[1] Dasarathy
[2] Sheela
[3] Rastring
[4] Erenstein

میزان شایستگی طبقه‌بند قابل تشخیص می‌باشد. برای مثال اگر از ۱۰ همسایه در طبقه‌بند knn استفاده شود (10-nn) و ۹ تا از ۱۰ همسایه برچسب مشابهی را پیشنهاد کنند بنابراین میزان اعتماد به این تصمیم بالا است. اگر خروجی‌های طبقه‌بند به طور مستدل و خوب توسط احتمالات پیشین کالیبره گردد به صورت $d_{\mathrm{ij}} = \hat{P}(\omega_j|x, D_i)$ برای میزان شایستگی $D_i \in D$ برای ورودی x خواهیم داشت:

$$C(D_i|x) = max_{j=1}^{c} \hat{P}(\omega_j|x, D_i) \qquad (۱-۲)$$

یک ساختار آبشاری[1] برای طبقه‌بندها پیشنهاد می‌گردد. زمانی که میزان شایستگی طبقه‌بند بالا می‌باشد یعنی بالای آستانه از پیش تعیین شده ما کلاس پیشنهاد شده به وسیله D_i را به عنوان برچسب x برمی‌گزینیم. اگر $C(D_i|x)$ پایین باشد، x به طبقه‌بند دیگری در ساختار آبشاری واگذار می‌گردد که به روش مشابه پردازش صورت می‌گیرد. نهایتاً، اگر آخرین طبقه‌بند در سیستم نیز نامطمئن باشد، سیستم می‌بایست از ساختن یک تصمیم خودداری[2] کند یا این که ممکن است یک کلاس که بیشترین احتمال را دارد به هر راهی انتخاب کند. این گونه مدل‌های آبشاری به طور وسیع برای سیستم‌های بلادرنگ[3] مفیدند زمانی که اکثریت ورودی‌ها نیاز به تعداد کمی طبقه‌بند جهت پردازش دارند [۳۳-۳۵].

روش مناسبی در انتخاب طبقه‌بند توسط بوتر و همکاران ارائه گردیده است[۳۶] . جهت توضیح پروسه انتخاب یک عضو از مجموعه و برای ساختن تصمیم بر پایه ورودی x آنها یک ترم انتخاب طبقه‌بند دینامیک معرفی کردند.

برای انتخاب مجموعه طبقه بندها نیاز است پاسخ پرسش‌های زیر داده شود:

۱. چگونه ما طبقه‌بندهای منجصر به فرد بسازیم؟

[1] Cascade
[2] refrain
[3] Real-time

۲. چگونه میزان شایستگی طبقه‌بندها را برای یک x داده شده ارزیابی کنیم ؟ اگر چندین طبقه‌بند با داشتن بیشترین شایستگی، یک گره ایجاد کنند چگونه ما می‌توانیم از گره خارج شویم؟

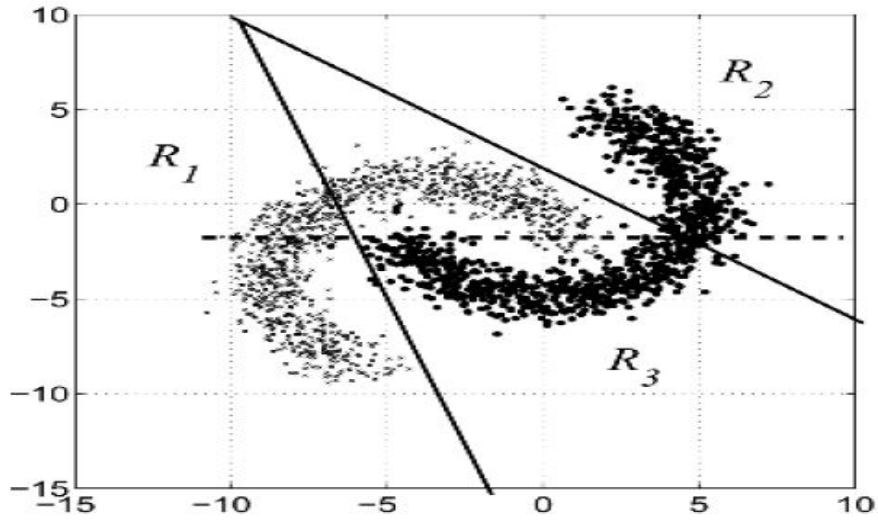
۳. یک بار که شایستگی‌ها پیدا می‌شوند از کدام استراتژی انتخاب می‌بایست استفاده کنیم؟

۴. به هر حال، اگر چندین طبقه‌بند شایستگی بالا و برابری داشته باشند، ما یک تصمیم را برداریم یا باید تصمیم‌های آنهایی که بیشترین شایستگی را دارند با هم ترکیب کنیم؟ چه موقع بهتر است که یک طبقه‌بند را برای برچسب زدن x انتخاب کنیم و چه موقع می‌بایست به دنبال تصمیم ترکیب شده باشیم؟

۲-۳تاثیر انتخاب طبقه‌بند در افزایش دقت طبقه‌بندی

ما یک مجموعه $D = \{D_1, D_2, \dots, D_l\}$ از طبقه‌بندها که قبلاً آموزش داده شده‌اند داریم. فرض کنید R^n به k ناحیه انتخاب که نواحی شایستگی نامیده می‌شوند تقسیم گردیده است$k > 1$. نواحی را با $R_k, \dots R_2, R_1$ مشخص می‌کنیم. این نواحی به کلاس‌های مناسبی وابسته نیستند و نیاز به شکل و اندازه مشخصی ندارند.

یک مثال تقسیم‌بندی نواحی در شکل ۱-۲ نشان داده شده است. یک پایگاه داده به شکل موز با ۲۰۰۰ داده نشان داده شده است. دو کلاس و بنابراین دو ناحیه طبقه‌بندی وجود دارد فرض کنید ۳ طبقه‌بند داریم D_1 اغلب ω_1 را پیشگویی می‌کند؛ D_2 اغلب ω_2 ؛ و طبقه‌بند خطی D_3 یک تابع مجزاساز است که در شکل با خط‌چین پررنگ نمایش داده شده است. دقت هر کدام از طبقه‌بندها تقریباً برابر ۰,۵ می‌باشد. نظر اکثریت آنها نیز همچنین بدون استفاده است چون اغلب با تصمیم D_3 منطبق می‌گردد و به سمت ۵۰ درصد خطا سوق می‌یابد. به هرحال اگر ما از سه ناحیه انتخاب استفاده کنیم و برای هر ناحیه یک طبقه‌بند را نامزد کنیم، خطای مجموعه ناچیز می‌شود.



شکل۲-۳: مثال تقسیم‌بندی نواحی[۴]

این مثال فقط قدرت روش انتخاب طبقه‌بند را نمایش می‌دهد. در عمل باید به شدت خوش شانس باشیم که بتوانیم نواحی پیدا کنیم که بر کارامدی مجموعه تاثیر داشته باشد. مسأله اصلی در انتخاب طبقه‌بند دقیقاً آموزش مجموعه است، پیدا کردن نواحی شایستگی به تخمین زدن شایستگی و انتخاب یک مد انتخاب بدین معنی که یک طبقه‌بند تکی را نامزد کنیم یا سه تا که بهترین هستند یا تصمیم همه را به وسیله شایستگی‌های آنها وزن‌دهی کنیم.

فرض کنید D^* طبقه‌بندی با بالاترین میانگین دقت در مجموعه D برای تمام فضای ویژگی R^n باشد. احتمال طبقه‌بندی صحیح به وسیله D_i در ناحیه R_j را با $P(D_i|R_j)$ مشخص می‌کنیم.

$D_{i(j)} \in D$ را یک طبقه‌بند مسئول نواحی R_j ,$j = 1,2,\ldots,k$ قرار می‌دهیم. احتمال طبقه‌بندی درست سیستم انتخاب طبقه‌بند ما به صورت زیر خواهد بود:

$$P(correct) = \sum_{j=1}^{k} P(R_j)P(D_{i(j)}|R_j) \qquad (۲-۲)$$

$P(R_j)$ احتمال ورودی x است که از توزیع مسأله بیرون کشیده شده و در ناحیه R_j افتاده است. برای ماکزیمم کردن $P(correct)$، $D_{i(j)}$ را به نحوی تعیین می‌کنیم که داشته باشیم

$$P\left(D_{i(j)}\middle|R_j\right) \geq P\left(D_t\middle|R_j\right) \quad \forall\, t = 1,2,\ldots,L \qquad (۳-۲)$$

از معادلات ۲-۲ و۲-۳ داریم:

$$P(correct) \geq \sum_{j=1}^{k} P\left(R_j\right)P\left(D^*\middle|R_j\right) = P(D^*) \qquad (۴-۲)$$

فرمول فوق نشان می‌دهد که طرح انتخاب طبقه‌بند در بدترین حالت به خوبی بهترین طبقه‌بند موجود در مجموعه D عمل می‌کند. علیرغم این که راهی برای بخش‌بندی فضای ویژگی‌ها به نواحی منتخب می‌باشد تنها شرط لازم و البته ماهرانه این است که مطمئن شویم $D_{i(j)}$ در بین L طبقه‌بند موجود در D بهترین طبقه‌بند برای ناحیه R_j باشد. توسعه جهت برآورده کردن این شرط یک مدل موفق برای انتخاب طبقه‌بند ایجاد می‌کند.

۲-۴ تخمین شایستگی طبقه‌بندها

در روش انتخاب طبقه‌بند ها نیاز به یافتن میزان شایستگی هر طبقه‌بند مربوط به نواحی شایستگی می‌باشد در این بخش روشهای تخمین شایسگی‌ها بررسی گردیده است.

۲-۴-۱ تخمین محلی شایستگی به صورت دینامیک

در این روش طبقه‌بندی برای تعیین برچسب ورودی x انتخاب می‌گردد، که به وسیله تخمین محلی دقت طبقه‌بندهای رقیب در زمان فاز عملیات انتخاب گردیده است. این روش خود به دو روش کلی تصمیم مستقل از تخمین‌ها و تصمیم وابسته به تخمین‌ها تقسیم میگردد، که در ادامه توضیح داده شده‌اند.

تصمیم مستقل از تخمین‌ها

در مرجع [۳۷] این روش یک روش قیاسی نامیده شده زیرا شایستگی‌ها فقط بر اساس موقعیت x تعیین می‌گردند. این روش خود شامل روشهای زیر می‌گردد:

۱)تخمین مستقیم k-nn

یک راه تخمین شایستگی شناسایی k تا از نزدیکترین همسایه‌های x از مجموعه داده‌های آموزشی یا validation برای پیدا کردن میزان دقت طبقه‌بندها بر روی این k عنصر می‌باشد [۳۲, ۳۸]. K ، یک پارامتر برای الگوریتم است که نیاز است قبلاً برای فاز عملیات تنظیم گردد.

۲)فاصله بر پایه تخمین k-nn

اکثر طبقه‌بندها خروجی‌های ملایم تولید می کنند، این خروجی‌ها می‌توانند در تخمین استفاده شوند. Giacinto و Rol پیشنهاد کردند که شایستگی هر طبقه‌بند D_i را بر اساس میانگین وزن داری از پیشگویی‌های درست طبقه‌بند برای برچسب k همسایه x محاسبه کنیم [۳۷].

تخمین زده شده به وسیله D_i برای احتمال برچسب صحیح کلاس برای z_j را با $P_i(l(z_j)|z_j)$ مشخص می‌کنیم. برای مثال فرض کنید خروجی D_i برای z_j در یک مسأله ۵ کلاسه به صورت [0.1,0.4,0.1,0.1,0.3] باشد. برچسب صحیح برای کلاس z_j به صورت $l(z_j) = \omega_5$ می‌باشد. گرچه تصمیم D_i کلاس ω_2 است، احتمال پیشنهاد شده برای برچسب صحیح کلاس $P_i(l(z_j)|z_j) = 0.3$ می‌باشد. احتمالات به وسیله فاصله از k همسایه وزن‌دهی می‌گردند. N_x را به عنوان مجموعه k نزدیکترین همسایه x می‌گذاریم. شایستگی طبقه‌بند D_i برای x به صورت زیر است.

$$C(D_i|x) = \frac{\sum_{z_j \in N_x} P_i(l(z_j)|z_j)(1/d(x,z_j))}{\sum_{z_j \in N_x}(1/d(x,z_j))} \qquad (۵-۲)$$

$d(x, z_j)$ فاصله میان x و نزدیکترین همسایه N_x بر اساس اندازه‌گیری مناسب فاصله محاسبه گردیده است، می‌باشد.

۳) تخمین توابع پتانسیل

راستریگین[1] و ارنستین[2] [۳۲] یک فاصله بر پایه شایستگی معرفی کرده‌اند که مدل توابع پتانسیل نامیده می‌شود. ما به فضای ویژگی‌ها به عنوان یک میدان نگاه می‌کنیم و فرض می‌کنیم هر نقطه در مجموعه داده ها در پتانسیل x شرکت دارد. پتانسیل طبقه‌بند D_i در x متناظر با شایستگی آن می‌باشد. بالا بودن پتانیسل معادل بالا بودن شایستگی می‌باشد. سهم مختص به $z_j \in Z$ برای $C(D_i|x)$ به صورت زیر می‌باشد

$$\varphi(x, z_j) = \frac{g_{ij}}{1+\alpha_{ij}(d(x,z_j))^2} \qquad (۶-۲)$$

که

$$g_{ij} = \begin{cases} 1 & : if\ D_i\ recognizes\ correctly\ z_j \in N_x \\ -1 & : otherwise, \end{cases}$$

و α_{ij} یک پارامتر به عنوان وزن یا میزان شرکت کردن z_j می‌باشد، در ساده ترین مورد $\alpha_{ij} = \alpha = 1$ برای تمامی $i = 1,2, \dots, l$ و $j = 1,2, \dots, N$.

$$C(D_i|x) = \sum_{z_j \in Z} \varphi(x, z_j) \qquad (۷-۲)$$

تخمین k-nn مستقیم نوعی از توابع پتانسیل با تابع فاصله زیر می‌باشد:

$$d(x, z_j) = \begin{cases} 1(or\ any\ positive\ constant), if\ z_j \in N_x \\ 0, otherwise \end{cases}$$

با این حال روشن نیست که نسخه بر پایه فاصله بهتر از تخمین مستقیم k-nn برای شایستگی باشد. گرچه گره‌های کمتر مانند تخمین بر پایه فاصله یک ویژگی مطلوب می‌باشد.

تصمیم وابسته به تخمین ها

این روش به صورت استقرائی در مرجع [۳۷] تشریح گردیده است. کلاس x توسط تمام طبقه بندها پیشگویی می شود. این روش خود شامل روشهای زیر می‌گردد:

[1] Rastrigin
[2] Erenstin

۱) تخمین k-nn مستقیم

s_i را برچسب کلاس x بگیرید که توسط طبقه بند D_i تشخیص داده شده است. k نزدیکترین همسایه x از Z را با $N_x^{(s_i)}$ مشخص میکنیم. شایستگی طبقه بند D_i برای x داده شده، $C(D_i|x)$، به صورت نسبت داده های موجود در $N_x^{(s_i)}$ که برچسب صحیح آنها s_i می باشد به کل داده های $N_x^{(s_i)}$ (k) محاسبه میگردد. این تخمین در مرجع[۵۰] دقت محلی کلاس (local class accuracy) نامیده شده است.

۲) تخمین فاصله بر پایه k-nn

احتمال این که برچسب کلاس z_j ، s_i توسط D_i صحیح تشخیص داده شده باشد با $P_i(s_i|z_j)$ مشخص می‌گردد. شایستگی D_i می‌تواند توسط $P_i(s_i|z_j)$ اندازه‌گیری گردد. که میانگین گیری شده از داده های همسایه x می‌باشد که برچسب صحیح آن‌ها s_i می‌باشد. استفاده فاصله ازx به عنوان وزن‌ها با شایستگی D_i را به صورت زیر محاسبه می‌کنیم:

$$C(D_i|x) = \frac{\sum_{z_j} P_i(s_i|z_j)(1/d(x,z_j))}{\sum_{z_j}(1/d(x,z_j))} \qquad (۸-۲)$$

جمع (سیگما) بر روی $z_j \in N_x$ می‌باشد چنان که $l(z_j) = s_i$. تعداد متفاوت همسایه‌ها (k) ممکن است برای N_x و $N_x^{(s_i)}$ پیشنهاد گردد. وود (Wood) در مرجع [۳۸] پی برد که تصمیم مستقیم- وابسته به تخمین k-nn برای شایستگی ممتاز تر از تخمین ایجاد شده با تصمیم غیر وابسته (مستقل) است. آنها k=10 را برای ایجاد مجموعه $N_x^{(s_i)}$ معرفی کرده‌اند.

مثال: تخمین شایستگی محلی طبقه‌بندها. جدول۱-۱ برچسب صحیح کلاس‌ها و برچسب‌های حدس زده شده با استفاده از طبقه‌بند D_i برای یک مجموعه فرضی از ۱۵ تا از نزدیکترین همسایه‌های x (N_x) داده شده است (برای صرفه جویی در فضا، فقط زیرنویس و کلاس‌ها داده شده‌اند).

جدول ۲-۱: برچسب صحیح کلاس‌ها و برچسب‌های حدس زده شده با استفاده از طبقه‌بند D_i برای یک مجموعه فرضی از ۱۵ تا از نزدیکترین همسایه‌های $\mathbf{x}$ (N_x)[۴]

نمونه(z_j)	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰	۱۱	۱۲	۱۳	۱۴	۱۵
برچسب صحیح($l(z_j)$)	۲	۱	۲	۲	۳	۱	۲	۱	۳	۳	۲	۱	۲	۲	۱
بچسب حدس زده شده (s_j)	۲	۳	۲	۲	۱	۱	۲	۲	۳	۳	۱	۲	۲	۲	۱

تصمیم مستقل تخمین k-nn برای شایستگی D_i دقت محاسبه شده D_i بر روی N_x می‌باشد که $C(D_i|x) = \frac{2}{3}$ می‌باشد. فرض کنید خروجی پیشنهاد شده برای x توسط D_i ، ω_2 باشد. تصمیم وابسته به k-nn مستقیم برای شایستگی D_i برای k=5 دقت D_i محاسبه گردیده بر روی $N_x^{\omega_2}$ می‌باشد وقتی که فقط ۵ تا از نزدیکترین همسایه‌هایی که برچسب ω_2 زده شده‌اند در نظر گرفته می‌شود. بنابراین $N_x^{\omega_2}$ شامل عناصر $\{1,3,4,7,8\}$ می‌باشد. چهار عنصر از $N_x^{\omega_2}$ درست برچسب گردیده‌اند (برچسب واقعی آنها نیز ω_2 بوده است). شایستگی محلی برای D_i به صورت $C(D_i|x) = \frac{4}{5}$ می‌باشد.

برای نشان دادن محاسبه تخمین بر اساس فاصله می‌بایست موارد زیر را فرض کنیم:

زیرنویس z_j در ردیف اول در جدول مقادیر واقعی $z_j \in R$ است و محل $x \in R$ در 0 می‌باشد.

فاصله اقلیدسی استفاده گردیده است. برای مثال $d(x, z_3) = 3$

D_i فقط برچسب خروجی را تولید می‌کند. ما فرض می‌کنیم $P_i(s_i|z_j) = 1$ (احتمال برای خروجی انتخاب شده) و $P_i(s_i|z_j) = 0$ برای هر $s \in \Omega$ دیگر که $s \neq s_j$ باشد بنابراین اگر کلاس پیشنهاد شده ω_2 باشد. سپس بردار احتمالات پیشنهاد شده $[0,1,0]$ خواهد بود.

تصمیم مستقل از تخمین k-nn بر پایه فاصله برای شایستگی D_i (از تمام N_x استفاده می‌کند) به صورت زیر می‌باشد:

$$C(D_i \mid x) = \frac{1+\frac{1}{3}+\frac{1}{4}+\frac{1}{6}+\frac{1}{7}+\frac{1}{9}+\frac{1}{10}+\frac{1}{13}+\frac{1}{14}+\frac{1}{15}}{1+\frac{1}{2}+\frac{1}{3}+\ldots+\frac{1}{15}} \approx 0.70$$

تابع پتانسیل تخمین به روش مشابهی محاسبه می‌گردد.

درباره خروجی پیشنهاد شده توسط D_i برای x را ω_2 بگیرید. برای تخمین تصمیم- وابسته از تمام N_x استفاده می‌کنیم و فقط عناصری که برچسب صحیح آنها ω_2 است را در نظر می‌گیریم

$$C(D_i \mid x) = \frac{1+\frac{1}{3}+\frac{1}{4}+\frac{1}{6}+\frac{1}{7}+\frac{1}{13}+\frac{1}{14}}{1+\frac{1}{3}+\frac{1}{4}+\frac{1}{7}+\frac{1}{11}+\frac{1}{13}+\frac{1}{14}} \approx 0.95$$

۲-۴-۲ پیش برآورد نواحی شایستگی

تخمین شایستگی به طور دینامیک ممکن است محاسبات بسیار طاقت فرسایی داشته باشد. ابتدا، k نزدیکترین همسایه x را مجبور هستیم بیابیم. برای تخمین بر پایه فاصله $L \times K$ همسایه ممکن است نیاز گردد. ثانیا مجبور خواهیم بود که تخمین شایستگی را محاسبه کنیم. برای کاهش پیچیدگی محاسباتی، شایستگی‌ها می‌توانند از قبل به صورت نواحی شایستگی محاسبه گردند. وقتی x برای طبقه‌بندی ارائه می‌گردد ناحیه شایستگی که x در آن می‌افتد شناسایی می‌گردد و x به طبقه‌بندی که مسئول آن ناحیه می‌باشد سپرده می‌شود. مسأله به صورت شناسایی ناحیه‌های شایستگی و طبقه‌بندهای متناظر ساده می‌گردد.

نواحی شایستگی به صورت دلخواه تا زمانی که ما تخمین معتبری از شایستگی در این نواحی داشته باشیم انتخاب می‌گردند تعداد نواحی شایستگی را با k مشخص می‌کنیم. نیازی نیست که تعداد طبقه‌بندها با تعداد نواحی برابر باشد. بعد ما تصمیم می‌گیریم کدام طبقه‌بند از D باید برای هر ناحیه R_j و $j = 1,2,..,k$ برداشته شود. بعضی از طبقه‌بندها ممکن است که هرگز کاندید نشوند بنابراین آنها در عملیات طرح ترکیب نیاز نمی‌باشند. حتی طبقه‌بندی که بالاترین دقت را برای کل فضای ویژگی داشته است ممکن است که در مجموعه

نهایی طبقه‌بندها وجود نداشته باشد. به عبارت دیگر یک طبقه‌بند ممکن است برای چندین ناحیه کاندید گردد (برای بیش از یک ناحیه کاندید گردد).

فضای ویژگی می‌تواند به نواحی منظم تقسیم‌بندی گردد ولی دراین مورد نواحی ممکن است فقط یک مقدار کوچک از داده ها را داشته باشد و به سمت شایستگی‌های نادرست سوق پیدا کند. برای اطمینان از این که نواحی حاوی مقدار کافی از داده می‌باشد می‌توانیم از خوشه‌بندی استفاده کینم و هر خوشه را منسوب به یک شایستگی نماییم. طبقه‌بندی که دقت تخمین آن در یک ناحیه R بالاترین دقت می‌باشد برای ایجاد تصمیم برای هر $x \in R$ انتخاب می‌گردد.

۳-۴-۲ خوشه بندی

یک روش انتخاب طبقه‌بند که خوشه‌بندی و انتخاب نامیده می‌شود در مراجع [۳۹, ۴۰] پیشنهاد گردیده است. مرحله آموزش عملیات خوشه‌بندی و انتخاب در زیر آورده شده است.

نواحی $R_1, R_2, \dots, R_k$ می‌توانند از قبل برای آموزش هر طبق‌بند تخمین زده شوند. یک طبقه‌بند می‌تواند بر روی داده‌های ناحیه R_j آموزش داده شود و تبدیل به خبره $D_{i(j)}$ گردد.

- **مرحله آموزش خوشه‌بندی و گزینش**[۴]

۱) طبقه بند های منحصر به فرد $D_1,\dots,D_L$ را با استفاده از مجموعه داده برچسب خورده Z را می‌سازیم. تعداد نواحی را K ناحیه انتخاب می‌کنیم.

۲) مجموعه داده Z را به K خوشه $C_1,\dots,C_K$ تقسیم میکنیم. می‌توان از الگوریتم K میانگین استفاده کرد.

۳) برای هر خوشه دقت طبقه‌بندی طبقه بندهای $D_1,\dots,D_L$ را محاسبه می‌کنیم. و دقیق‌ترین را برای طبقه بندی آن خوشه که ناحیه R_j تعریف کردیم به صورت $D_{i(j)}$ تعیین می‌کنیم.

- **مرحله عملیات خوشه‌بندی و انتخاب**[۴]

۱) نزدیک‌ترین مرکز خوشه به ورودی x را پیدا کرده و با v_j مشخص میکنیم.

۲) برای تعیین برچسب x از طبقه‌بند $D_{i(j)}$ استفاده می‌کنیم.

۲-۴-۴ خوشه‌بندی گزینشی

Liu و Yuan یک روش خوشه‌بندی گزینشی پیشنهاد کردند [۴۱]. در عوض یک بخش‌بندی تنها برای فضای ویژگی‌ها، آنها یک بخش‌بندی برای هر طبقه‌بند لحاظ کردند. طبقه‌بند D_i را بر روی تمام Z آموزش دادند.

در فاز آموزش داده‌ها به z^+ و z^- تقسیم می‌گردند، به این صورت که z^+ شامل فقط داده‌هایی هستند که توسط D_i به درستی طبقه‌بندی شده‌اند و z^- داده‌هایی هستند که به صورت نادرست طبقه‌بندی گردیده‌اند ($z^+ \cup z^- = Z$) . می‌توان فرض کرد که تعداد C خوشه بیضی‌گون در z^+ وجود دارد که هر خوشه متناظر با یک کلاس می‌باشد. میانگین‌ها و ماتریس کواریانس این خوشه‌ها بر اساس داده‌هایی که به درستی توسط D_i طبقه‌بندی گردیده‌اند تخمین زده می‌شود. در z^- تعداد خوشه‌ها k_i در طول دستورالعمل خوشه‌بندی ارزیابی می‌گردد. میانگین‌ها و ماتریس‌های کواریانس برای خوشه‌های جدید توسط داده‌های مربوط تخمین زده می‌شود. از این رو هر طبقه‌بند در مجموعه یک بخش‌بندی فضای ویژگی‌ها به $c + k_i$ بخش را پیشنهاد می‌کند و میانگین‌ها و ماتریس‌های کواریانس خوشه‌های پیشنهادی را محاسبه می‌کند. برای تکمیل آموزش ما شایستگی D_i را در تمام $c + k_i$ ناحیه پیشنهاد شده توسط آن تخمین می‌زنیم.

در فاز عملیات، ورودی x جهت طبقه‌بندی توسط هر طبقه‌بند به طور جداگانه رسیدگی می‌شود. برای مثال D_i ناحیه شایستگی که x به آن تعلق دارد از میان $c + k_i$ ناحیه برای D_i شناسایی می‌گردد از فاصله (Mahalonobis) مرکز هر خوشه تا x استفاده می‌گردد. شایستگی برای ناحیه x محاسبه می‌گردد. شایستگی

تمام طبقه‌بندها برای x داده شده با این روش محاسبه می‌گردد و با هم مقایسه می‌شوند. طبقه‌بند با بالاترین شایستگی برچسب x را می‌دهد [۴].

Liu و Yuan نتایج فوق العاده‌ای برای این روش ساده خوشه‌بندی و گزینش گزارش کردند [۴۱]. حقیقتاً اگر ما فضای ویژگی‌ها را به نواحی کوچکتر بخش‌بندی کنیم، ما یک مدل انعطاف‌پذیرتری برای برچسب زدن x داریم ولی تخمین‌های شایستگی‌ها می‌بایست به اندازه کافی معتبر باشند.

۲-۵به هم زدن وضع مساوی (شکستن گره) برای شایستگی های برابر

به هم خوردن وضع مساوی (شکستن گره) برای شایستگی‌های برابر برای مدل دینامیک انتخاب طبقه‌بند به هم خوردن وضع مساوی (شکستن گره) به صورت زیر انجام می‌پذیرد:

۱. به محض رسیدن یک ورودی x، آن را به وسیله $D_1, \mathrm{D}_2, \dots, D_l$ برچسب می‌کنیم. اگر همه طبقه‌بندها یک برچسب را برای x ایجاد کند، این برچسب را به x می‌دهیم و برمی‌گردیم.

۲. اگر اختلاف وجود داشته باشد، سپس دقت محلی را تخمین می‌زنیم، $C(D_i|x)$، برای هر $D_i, \mathrm{i} = 1,2, \dots, L$ برای انجام این کار برچسب کلاس s_i پیشنهاد شده توسط D_i را برمی‌داریم و k نقطه اطراف x برای D_i پیدا می‌کنیم که برچسب مشابه گرفته باشند. نسبت نقاطی که برچسب درست آنها s_i می‌باشند را محاسبه می‌کنیم که یک تخمین برای دقت محلی D_i مربوط به کلاس s_i می‌باشد (تخمین تصمیم بر پایه k-nn مستقیم).

۳. اگر یک برنده یکتا در رقابت دقت محلی وجود داشته باشد، برچسب x را قرار می‌دهیم و برمی‌گردیم در غیر این صورت چک می‌کنیم که برنده‌های ایجاد کننده گره برچسب یکسانی برای x دارند. در این صورت برچسب را قبول می‌کنیم و برمی‌گردیم. اگر یک کلاس واحد توسط تعداد زیادی از طبقه‌بندهای گره خورده انتخاب شده است، این برچسب را برای x برمی‌گزینیم و برمی‌گردیم.

۴. در غیر این صورت، یک برچسب کلاس میان طبقه‌بندهای با بیشترین شایستگی وجود دارد. طبقه‌بند بعدی با بیشترین شایستگی محلی را شناسایی می‌کنیم برای خارج شدن از گره. اگر همه طبقه‌بندها با هم گره خورده باشند (طبقه‌بندی برای شکسبتن گره باقی نمانده باشد). یک برچسب کلاس را به طور تصادفی از طبقه‌بنهای گره خورده برمی‌داریم و برمی‌گردیم. اگر یک برنده واحد برای شایستگی محلی (دوم) وجود داشته باشد و بتواند گره را حل کند، از برچسب برنده برای x استفاده می‌کنیم و برمی‌گردیم.

۵. اگر هیچکدام از ماده‌های فوق اتفاق نیفتد، به طور تصادفی گره را می‌شکنیم و برچسب آن را برای x انتخاب می‌کنیم و برمی‌گردیم.

مورد آخر در لیست در دستورالعمل اصلی شکستن گره (برهم زدن وضع مشابه) وجود ندارد. ما آن را برای شکستن بازگشت اضافه کرده‌ایم. جلو بردن آنالیزها برای گره‌ها پیچیدگی زیادی در برنامه ایجاد می‌کند زیرا ممکن است یک گره دیگر برای دومین بالاترین شایستگی وجود داشته باشد، با طبقه‌بندهایی در گره که اختلاف رای برای برچسب کلاس x دارند.

جدول ۲-۲ مثال‌هایی از موارد شکستن گره و خروجی‌های مربوطه نمایش داده شده است، در مورد اول یک طبقه‌بند با بالاترین سطح شایستگی وجود دارد که مشخص کننده برچسب کلاس می‌باشد. در مورد دوم یک گره از طبقه‌بندهای با بالاترین سطح شایستگی وجود دارد که رای اکثریت آنها تعیین کننده برچسب کلاس می‌باشد و اکثریت برچسب ۲ را پیشنهاد کرده‌اند، در مورد سوم هنوز ما انتخاب تصادفی بین برچسب ۱ و ۲ داریم که توسط طبقه‌بندهای با بیشترین سطح شایستگی پیشنهاد گردیده است و باید به طبقه‌بند با رتبه بعدی سطح شایستگی مراجعه کنیم و طبقه بند با دومین سطح شایستگی تعیین کننده برچسب کلاس می‌باشد که برچسب ۲ را پیشنهاد می‌کند. در مورد چهارم یک انتخاب تصادفی ساخته می‌شود زیرا یک طبقه‌بند تکی با

دومین سطح شایستگی بالا وجود ندارد. در مورد آخر طبقه‌بندهای با بالاترین سطح شایستگی ایجاد گره کرده اند و همین جور طبقه‌بندهای با دومین سطح شایستگی گره ایجاد کرده‌اند ما می‌توانیم جلوتر رویم و از بین ۴ طبقه‌بند با شایستگی‌های سطح دوم رای اکثریت که کلاس ۲ می‌باشد را برگزینیم، به هر حال اگر این باز یک گره باشد آن پیچیدگی غیر ضروری و زمان‌بر خواهد بود، به این علت مورد پنجم در بالا معرفی گردیده است. و از بین طبقه‌بندهایی که با بالاترین سطح شایستگی گره ایجاد کرده‌اند به طور رندم نظر یک طبقه‌بند را به عنوان برچسب کلاس انتخاب می‌کنیم.

۲-۶انتخاب یا ترکیب؟

گزینش توسط طراحی تضمین شده است که حداقل به اندازه بهترین طبقه بند تکی D^* برای آموزش دقت داشته باشد. به هر حال، ممکن است مدل بیش از حد آموزش داده شود و یک خطای پایین فریبنده برای آموزش ارائه دهد. برای محافظت در برابر آموزش بیش از اندازه ممکن است که ما از آزمایش آماری جهت میزان اطمینان استفاده کنیم که این کار بهترین طبقه بند در ناحیهء R_j ، $D_{i(j)}$ را به دست می دهد، اختلاف قابل توجهی باقی خواهد ماند از این رو ما فقط به اندازهء تفاوت بین بهترین طبقه بند و مابقی طبقه بندها سود

جدول ۲-۲: مثال از شکستن گره برای مدل انتخاب دینامیک [۴]

	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9
Class labels	1	2	1	2	3	2	3	1	2
Competences	0.8	0.7	0.7	0.6	0.6	0.6	0.4	0.3	0.2
Final label: **1** (single most competent classifier)									
Class labels	1	2	2	2	3	2	3	1	2
Competences	0.7	0.7	0.7	0.6	0.6	0.6	0.4	0.3	0.2
Final label: **2** (majority of the competence-tied classifiers)									
Class labels	1	2	2	2	3	2	3	1	2
Competences	0.7	0.7	0.6	0.5	0.5	0.5	0.4	0.3	0.2
Final label: **2** (competence-tie resolved by the second most competent classifier)									
Class labels	1	2	1	1	2	2	3	1	2
Competences	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7	0.7
Final label: **1** (random selection between competence-tied labels)									
Class labels	1	2	2	2	3	2	3	1	2
Competences	0.7	0.7	0.6	0.6	0.6	0.6	0.4	0.3	0.2
Final label: **1** (random selection between competence tied-labels (item 5))									

خواهیم برد، ما می توانیم یک آزمایش بین هر دو طبقه بند شکل دهیم. این کار برای حذف طبقه بند دوم کافی است.

اگر $D_{i(j)}$ به طور قابل توجهی از دومین طبقه بند بهتر باشد، $D_{i(j)}$ می تواند به عنوان طبقه بند مسئول ناحیه R_j کاندید گردد. در غیر این صورت، یک طرح شامل بیش از یک طبقه بند ممکن است نتیجه شود.

به عنوان یک مثال، فرض کنید ۵ طبقه بند بر روی یک مجموعه داده شامل ۱۰۰ داده طراحی گردیده باشد تعریف می کنیم $y_i = [y_{i,1}, y_{i,2}, \dots, y_{i,100}]^t$ یک بردار حاصل از خروجی طبقه بندی توسط D_i بر روی مجموعه داده باشد، به این شکل که $y_{ij} = 1$ ، اگر D_i به درستی داده jام را شناسایی کرده باشد و 0 اگر درست شناسایی نکرده باشد. جدول ۲-۳ توزیع $\{y_1, y_2, \dots, y_5\}$ را برای ۱۰۰ داده نمایش می دهد. تعداد کل داده هایی که توسط هر طبقه بند شناسایی گردیده اند در آخرین سطر نمایش داده شده است. ممکن است که ما

وسوسه شویم که D_1 را برای ناحیهء R_j به دلیل دقت ۷۶ درصدی آن که ۵ درصد از دومین طبقه بند بهتر می باشد انتخاب کنیم. به هر حال، آنالیز های آزمایش جفتی اظهار می کند که D_1 تفاوت قابل توجهی با D_4 و حتی با D_5 ندارد. از این رو، احتمالا بهتر است که طرح ترکیب بیش از یک تصمیم را بر روی ناحیهء R_j بدهیم.

جدول ۲-۳: توزیع کلاسبندی صحیح/غلط ۵ طبقه‌بند برای مجموعه داده با ۱۰۰ داده[۴]

y_1	y_2	y_3	y_4	y_5	Number
1	1	1	1	1	42
0	0	0	1	1	18
1	1	0	0	0	13
1	0	0	1	0	11
1	0	1	0	0	10
0	0	0	0	1	6
76	55	52	71	66	–

۲-۷ Bagging and boosting

۲-۷-۱ منشاء پیدایش Bagging

Breiman اصطلاح Bagging را به عنوان یک سر کلمه برای Bootstrap AGGregatING معرفی کرد [۴]. ایده Bagging ساده و جذاب می باشد، یک مجموعه از طبقه بند ها وجود دارند که مجموعه داده های آموزشی را به صورت خود راه انداز[1] می سازند. خروجی طبقه بند نهایی از ائتلاف نظر همه طبقه بند ها حاصل می گردد [۴۲].

تنوع لازم برای ساختن مجموعه با استفاده از مجموعه داده های آموزشی متفاوت حاصل می گردد. به صورت ایده آل می بایست مجموعه داده آموزشی به صورت رندم از کل توزیع مسئله ساخته شود. در عمل ما

[1] Bootstrap

فقط یک مجموعه داده آموزشی برچسب دار $Z = \{z_1, z_2, \dots, z_N\}$ در دسترس داریم، و مجبوریم که با پردازش تعداد L مجموعه آموزشی به صورت رندم تولید کنیم. ما نمونه گیری همراه با جایگزینی[1] از مجموعهِ آموزشی اصلی، جهت ساختن یک مجموعه آموزشی با طول N انجام می دهیم [۴۳]. برای استفاده از مجموعه داده های آموزشی متنوع می بایست طبقه بند های پایه ناپایدار باشند، و تغییر کوچکی در مجموعه داده آموزشی منجر به تغییر بزرگ در خروجی طبقه بند گردد، در غیر این صورت حاصل مجموعه اجتماع طبقه بند های تقریبا یکسان خواهد بود، بنا بر این بهبودی در کارآمدی نسبت به یک طبقه‌بند واحد وجود نخواهد داشت. به عنوان مثال‌هایی از طبقه بند های ناپایدار می توان از شبکه های عصبی[2] و درخت تصمیم گیری[3] نام برد، در حالی که نزدیکترین همسایه[4] مثالی از یک طبقه بند پایدار می باشد. مراحل آموزش و طبقه‌بندی الگوریتم BAGGING در ادامه آورده شده است. BAGGING در هر دو فاز آموزش و عملکرد یک الگوریتم موازی می باشد. بدین معنی که طبقه-بندهای گردآوری شده می توانند توسط پردازشگرهای متفاوت در صورت نیاز آموزش داده شوند [۴].

- **فاز آموزش الگوریتم BAGGING**[۴]

۱) پارامترها را به صورت زیر مقداردهی اولیه می‌کنیم

- مجموعه طبقه‌بندها $D = \phi$
- L، تعداد طبقه‌بندهایی که باید آموزش دهیم را تعیین میکنیم

۲) برای k=1, . . . L

- نمونه های را از مجموعه داده Z برمیداریم.

[1] With replacement
[2] Neural Networks
[3] Decision Tree
[4] K-nearest neighbour

- طبقه‌بند D_k را بر روی مجموعه S_k می‌سازیم.
- طبقه بند D_k را به مجموعه طبقه بندها اظافه میکنیم($D = D \cup D_k$).

۳) مجموعه D را نگهداری می‌کنیم

- فاز طبقه‌بندی الگوریتم BAGGING[۴]

۱) طبقه‌بندهای $D_1,...,D_L$ را برای ورودی x استفاده می‌کنیم.

۲) کلاسی که توسط بیشترین تعداد طبقه‌بندها حدس زده اند را برای برچسب x انتخاب می‌کنیم.

۲-۷-۲ چرا BAGGING به کار گرفته می‌شود؟

اگر خروجی طبقه بند مستقل از طبقه بند ها باشد و دارای دقت منحصر به فرد P باشد، بنا بر این نظر اکثریت کارامدی بالاتر را ضمانت می کند [۴۴]. هدف Bagging توسعه دادن طبقه بند های مستقل با گرفتن داده های آموزشی به صورت تکرار های خود راه انداز می باشد. نمونه ها به صورت شبه مستقل می باشند چون همه از مجموعه یکسان Z گرفته شده اند. به هر حال، حتی اگر آنها به صورت مستقل از توزیع مساله گرفته شده باشند، طبقه بند های بنا گذاشته روی این داده های آموزشی ممکن است که خروجی های مستقلی ایجاد نکنند.

۲-۷-۳ انواع Bagging

جنگل تصادفی[1]

در مرجع [۴۵] بریمان یک نوع از Bagging به نام جنگل تصادفی را پیشنهاد می کند. جنگل تصادفی یک رده عمومی از روشهای ساختن مجموعه طبقه بندها با به کار گیری درخت تصمیم گیری به عنوان طبقه بند

[1] Random Forest

پایه می باشد. برای لقب دادن جنگل تصادفی به یک مجموعه از طبقه بندها می بایست این مجموعه از درخت های تصمیم گیری ساخته شده باشند که برای ساختن هر کدام از این درخت های تصمیم گیری یک بردار رندم θ_k و $k = 1,2, \ldots, L$ مستقل از دیگر بردارها ولی با توزیع یکسان تولید شده باشد.

تعریف جنگل تصادفی

یک جنگل تصادفی یک طبقه بند تشکیل شده از یک مجموعه از طبقه بند های پایه با ساختار درخت تصمیم گیری می باشد، که هر درخت مربوط به یک بردار رندم θ_k می باشد، که با هم مستقل و دارای توزیع یکسان هستند. هر درخت یک نظر برای محتملترین کلاس برای ورودی x ایجاد می کند.

بنا بر این یک جنگل تصادفی می تواند با نمونه گیری از مجموعه ویژگی ها از مجموعه داده آموزشی تولید گردد. یا اینکه فقط به طور رندم بعضی از پارامترهای درخت را تغییر دهیم. هر ترکیب از این گوناگونی ها نیز می تواند منجر به یک جنگل تصادفی گردد. برای مثال ممکن است از مجموعه ویژگی ها و همچنین از مجموعهء داده های آموزشی نمونه گیری کنیم [۴]. در این حالت بردار θ_k می تواند از دو مجموعه خود راه انداز الحاق شده باشد، یکی از مجموعه داده های آموزشی و دیگری از مجموعه ویژه گی ها که با هم شامل N+n عنصر می شود.

یکی از موفقترین روشها در مورد خانواده جنگل تصادفی انتخاب رندم ورودی می باشد. همزمان با انتخاب نمونه های خود راه انداز از مجموعه داده های آموزشی، برای هر یک از گره های درخت تصمیم گیری ویژگی ها به صورت رندم انتخاب می شود. ما در اینجا یک زیر مجموعه S به صورت رندم از M ویژگی از مجموعه اصلی شامل n ویژگی انتخاب می کنیم، و در داخل S به دنبال بهترین ویژگی ها برای انشعاب گره ها می گردیم. یک زیر مجموعه جدید برای هر کدام از گره ها انتخاب می گردد.

بریمان پیشنهاد کرد درخت را به صورت CART بالغ یا تمام بدون هرس کردن رشد دهیم مقدار پیشنهاد شده برای M $log_2(n+1)$ می باشد که n تعداد کل ویژگی ها می باشد.

Pasting small votes

حجم زیادی از پایگاه داده ها که اکنون موجود هستند با پایگاه داده هایی که در روزهای آغازین شناسایی الگو به وجود آمده اند در دهه های ۱۹۶۰ و ۱۹۷۰ مشترکند. وجود بیش از اندازه داده و رشد پیوسته محاسبات مقدار زیادی داده ایجاد می کند که می بایست ذخیره گردد، برای مثال به طور جزئی تجارت، بازارهای مالی و بورس، بیو انفورماتیک، صنعت، مخابرات، . . . .. در آغاز توجه به این بود که چگونه می توان از پایگاه های داده استفاده کرد که نتایج معتبر به دست آید، با گذشت زمان مرکز توجه به سمت اینکه چگونه با پایگاه های داده برخورد کنیم اگر که با حافظه کامپیوتر متناسب نباشد.

بریمان [۴۶] استفاده از small votes را پیشنهاد نمود که به موجب آن طبقه بند های خاص بر روی بخشهای کوچکی از داده های آموزشی که bite نامیده می شوند آموزش داده می شوند [۴۷] مجموعه داده های آموزشی نیز به صورت رندم از پایگاه داده بزرگ نمونه گیری می شود که Rvotes نامیده می شوند(شبیه bagging) یا اینکه بر پایهء میزان اهمیت می باشد که Ivotes نامیده می شوند(شبیه boosting).
Pasting small votes برای آموزش on-line مناسب هستند بدین گونه که مجموعه می تواند با اظافه کردن یک طبقه بند جدید در هر مرتبه به روز شود، در هر مرتبه به مقدار کافی داده آموزشی جدید جمع می گردد.

مجموعه $Z = \{z_1, z_2, \dots, z_N\}$ به عنوان یک مجموعه آموزشی برچسب دار (لیبل شده) موجود است (فرض کنید N حدود یکصد هزار می باشد) و M اندازه bite می باشد (برای مثال ۲۰۰). یک طرح ممکن جهت ساختن مجموعه D ثابت گرفتن تعداد طبقه بند ها L در مراحل کار می باشد. تعداد L مجموعه نمونه به صورت رندم با جایگزینی با اندازه M از Z گرفته می شود(Rvotes) و بر روی هر مجموعه نمونه یک طبقه بند

می سازیم. برای حدس زدن برچسب کلاس از نظر اکثریت استفاده می کنیم. از مجموعه validation برای تخمین دقت D و بهینه نمودن مجموعه با مراجعه به L و M . Pasting Rvotes ساده است ولی مشاهده گردیده که در مقایسه با Pasting Ivote دقت کمتری دارد [۴۶, ۴۷].

Pasting Ivote نیاز به تخمین دقت در هر مرحله از ساخت دارد. مجموعه داده آموزشی جدید از Z نمونه گیری می شود (همراه با جایگذاری) مطابق دقت کنونی نمونه گیری انجام می شود. ایده این می باشد که نصف عناصر یک bite جدید آسان و نصف دیگر سخت باشد. فرض کنید l طبقه بند ساخته شده است بنابراین مجموعه کنونی $D_L = \{D_1, D_2, \dots, D_l\}$ و تخمین خطای تعمیم یافته برای D_l ، e_l باشد (فرض کنید $e_l < 1/2$). داده های آموزشی برای طبقه بند جدید D_{l+1} از Z بر اساس شیوه زیر نمونه گیری می شود:

۱) یک عنصر z_j از Z به صورت رندم می گیریم (از توزیع یکنواخت استفاده می کنیم).

۲) طبقه بند هایی که z_j در مجموعه آموزشی آنها قرار دارد را از D_L شناسایی می کنیم، به این طبقه بندها out-of-bag گفته می شود. اگر z_j در تمام مجموعه داده های آموزشی وجود داشته باشد از آن صرفنظر می کنیم و یک عنصر دیگر از Z برمی داریم.

۳) z_j را به طبقه بند های D_L جهت تشخیص کلاس اعمال می کنیم و نظر اکثریت را به عنوان برچسب کلاس z_j پیدا می کنیم.

۴) اگر بر چسب حدس زده نادرست باشد، z_j به مجموعه داده آموزشی برای طبقه بند D_{l+1} اظافه می گردد. در غیر این صورت، به مجموعه داده های آموزشی با احتمال $\frac{e_l}{e_l - 1}$ اظافه میگردد.

۵) مراحل ۱ تا ۴ را ادامه می دهیم تا وقتی که M عنصر از Z برای آموزش طبقه بند D_{l+1} انتخاب گردد.

یک رویهء آبشاری مشابه جهت فیلتر کردن(فیلترینگ) مجموعه های آموزشی برای یک مجموعه از طبقه بند ها شامل سه شبکه عصبی توسط دراکر[۱] در مرجع [۴] پیشنهاد گردیده است.

مرحله بعد آموزش طبقه بند D_{l+1} بر روی مجموعه داده انتخاب شده می باشد. تخمین e_{l+1} جهت انتخاب کردن مجموعه داده آموزشی بعدی نیاز می باشد که می تواند از راه های گوناگون محاسبه گردد.

بریمان استفاده از تخمین out-of-bag را پیشنهاد کرد که به وسیلهء آن e_l تخمین زده می شود. زمانی که انتخاب مجموعه داده آموزشی برای طبقه بند D_{l+1} صورت می پذیرد، نرخ طبقه بندی نادرست[۲] از کل تعداد عناصر امتحان شده با $\tilde{e}$ مشخص می گردد(به معنی تعدادی که صرفنظر نشده اند). خطای e_l توسط تخمین زیر محاسبه می گردد.

$$e_l = \alpha e_{l-1} + (1-\alpha)\tilde{e} \qquad (۹-۲)$$

α یک پارامتر در رنج (۰و۱) می باشد(مقدار پیشنهاد شده ۰/۷۵ می باشد [۴۶]).

برای ساختن یک bite با اندازهء M که شامل تقریبا نصف آسان و نصف سخت می باشد که به واسطه دستور العمل فوق انتخاب شده اند. ما باید بر روی میانگین حدود $M/2e_{l-1}$ داده که به صورت رندم انتخاب گردیده اند آزمایش کنیم اگر e_{l-1} بزرگ باشد، بنابر این تعداد عناصر آزمایش شده خیلی بزرگ نیستند. بنا بر این تخمین $\tilde{e}$ ممکن است واریانس بالایی داشته باشد.

برای شروع الگوریتم ابتدا یک طبقه بند را بر روی مجموعه داده های آموزشی که به اندازهء M به صورت رندم نمونه برداری شده اند بنا می گذاریم و مقدار e_0 را بر روی یک مجموعه آزمایشی out-of-bag ارزیابی می کنیم. از e_0 جهت انتخاب مجموعه داده آموزشی برای D_1 استفاده می کنیم و به طور همزمان e_1 را محاسبه می کنیم. و با e_1 یک مجموعه داده آموزشی برای D_2 انتخاب می کنیم و به همین صورت تا آخر ادامه می دهیم .

در مرجع [۴۶] می توان مشاهده کرد که e_1 یک تخمین نویزی است و اغلب یک بایاس سیستماتیک (اصولی) بالا یا پایین مقدار صحیح خطا برای D_l دارد. به هر حال توجه داشته باشید که اغلب e_l مقدار

خطایD_1 را به خوبی دنبال می کند. بنابر این اگر زمانی که سطح e_1 صفر گردد یا اینکه زمانی که شروع به افزایش می کند آموزش را متوقف کنیم، می بایست مقدار واقعی خطا نیز بهترین مقدار ممکن را داشته باشد. در آزمایشاتی که توسط بریمان صورت گرفت N مقادیر ۲۰۰۰ تا ۴۳۰۰۰ را داشته و M از ۱۰۰ تا ۸۰۰ تغییر کرده، و درخت تصمیم گیری بدون حرس کردن جهت طبقه بند پایه انتخاب گردیده است.

یک راه دیگر این است که از out-of-bag استفاده نکنیم و از یک مجموعهء جداگانه validation برای ارزیابی e_1 استفاده کنیم. روش پیشنهادی را می توان برای پایگاه های داده بزرگ به کار برد، کنار گذاشتن یک مجموعه داده validation با اندازه انتخاب شده بدون کم شدن ، اندازه مجموعه داده آموزشی به طور اساسی نباید یک مشکل باشد. تمام عناصر validation در تمام مراحل ساخت مجموعهء طبقه بندها out-of-bag هستند. این موضوع می تواند از تغییرپذیری و بایاس سیستماتیک تخمین out-of-bag اجتناب کند و معادلهء ۲-۹ نیز دیگر مورد نیاز نیست.

چاولا[1] [۴۷] استفاده از یک نسخهء توضیع شده از pasting small votes را پیشنهاد کرد. پایگاه داده Z به زیر مجموعه های گسسته تقسیم می گردد و یک مجموعه از Ivotes بر روی هر زیر مجموعه رشد داده می شود. سپس D با متحد کردن تمام طبقه بند های طراحی شده بر تمام پایگاه داده شکل می گیرد. اساس این نسخهء توزیع شده در اصل یک گین قابل توجه در زمان محاسبه در مقایسه با Ivotes بر تمامZ می باشدکه خود این در مقایسه با Adaboost بر روی Z با صرفه می باشد. هر دستور العمل Ivotes به صورت تکی می توانند بر روی پردازنده جدا گانه run شود. و زمان آموزش نیازی به هیچ مراوده ای ندارد. دقت مجموعهء نهایی را می توان در مرجع [۴۷] مشاهده کرد در مقایسه با Ivotes یا Adaboost بر روی تمام Z . این نتایج برای استفاده از بخش های جداگانه که به گوناگونی اضافی در مجموعه نهایی منجر می شود قابل قبول است.

نهایتا برای استخراج منفعت عملکرد الگوریتم ها با Ivotes توزیع شده برای پر شدن مجموعه ما نیاز به وسیله مناسب محاسبهء توزیع شده داریم، برای مثال یک گروه کامپیوتر.

۴-۷-۲ Boosting

اساس: الگوریتم Hedge(β)

Boosting از الگوریتمی به نام Hedge(β) الهام گرفته شده است [۴۸]. این الگوریتم وزنهایی را برای یک سری از استراتژی هایی که برای پیشگویی خروجی یک واقعه مشخص استفاده می شوند تعیین می کند. وزن استراتژیS_i ، اگر به طور شایسته نسبت داده شده باشد، می تواند به عنوان احتمال اینکه S_i بهترین پیشگویی را در گروه داشته باشد تفسیر گردد(بیشترین دقت را داشته باشد). توزیع به صورت آنلاین بعد از آمدن هر خروجی به روز می گردد. استراتژی هایی که درست پیشگویی کرده بودند به وزن های آنها اظافه می گردد و از وزن استراتژی هایی که پیشگویی نادرست داشته اند کاسته می شود.

در اینجا Hedge(β) را برای بر پا کردن ترکیب طبقه بندها شرح می دهیم، گر چه قدری با روش پیشنهاد شده در مرجع [۴۸] متفاوت می باشد. در اینجا استراتژی های مربوط به طبقه بند ها در ساختن مجموعه و رویداد متناظر با برچسب z_j که به صورت رندم از Z بیرون کشیده شده است. فرض کنید مجموعه طبقه بندهای $D = \{D_1, D_2, \dots, D_l\}$ موجود می باشد ولی ما نمی دانیم که هر طبقه بند در مقابل سوال مسئله چگونه عمل می کند. یک نمونه از Z انتخاب می گردد و هر طبقه بند یک پیشگویی برای برچسب آن انجام می دهد. کدام پیشگویی باید انتخاب گردد؟ در ابتدا دلیلی برای ترجیح دادن یک طبقه بند بر بقیه نداریم، بنا براین یک طبقه بند را به صورت رندم از D انتخاب می کنیم و تصمیم آن را بر می داریم. Loss را برابر ۱ اگر که برچسب کلاس اشتباه باشد و ۰ اگر برچسب کلاس درست حدس زده شده باشد تعریف می کنیم. مقدار Loss قابل قبول برای انتخاب رندم یک طبقه بند میانگین تعداد طبقه بندهای موجود در D می باشد که نمونه انتخاب شده را اشتباه

برچسب داده اند. از این رو که می خواهیم تصمیم را برای نمونهء بعدی بهبود دهیم معقول است بالا بردن احتمال طبقه بند هایی که بیشترین تصمیم درست را برای داده های قبلی داشته اند. بنابر این ما توزیع را بر روی D تغییر می دهیم بر روی مثالهایی که بیشتر طبقه بندی درست گردیده اند. اگر ما برای توسعه بدانیم کدام طبقه بند بهترین می باشد و اغلب همراه با تصمیم آن جلو می رویم، سپس Loss تجمعی (خطا بر اساس همهء مثال های آزمایش شده) به سمت کوچکترین مقدار میل میکند. به هر حال ما فرض می کنیم که چنین اطلاع یا دانشی وجود ندارد و به این منظور الگوریتم Hedge(β) را به کار میگیریم. فرض کنید ما یک پیش بینی به وسیلهء انتخاب رندم یک طبقه بند که به صورت رندم از توزیع D انتخاب گردیده انجام می دهیم. الگوریتم Hedge(β) توزیع D را به گونه ای تغییر می دهد که Loss تجمعی (جمع شونده) برای تصمیم می نیمم گردد. این توزیع با نرمالیزه کردن یک سری از وزن ها که با آمدن هر داده جدید از Z به روز می گردد محاسبه می شود. وزنها برای طبقه بندهایی که اشتباه تشخیص داده اند (Loss =1 دارند) کاهش می یابد با ظریب β که از قبل تعیین گردیده است در حالی که وزنهای دیگر تغییر نمی کنند وقتی که توزیع دوباره محاسبه می گردد، احتمالات برای طبقه بند هایی که موفق بوده اند افزایش می یابد.

Adaboost Algoritm

Boosting که در مرجع [۴۸] تعریف گردیده است مربوط می باشد به مسالهء عمومی تولید یک قاعدهء پیشگویی خیلی دقیق به وسیلهء ترکیب کردن به صورت ناهموار قاعده های غیر دقیق و هموار کردن آن. ایدهء عمومی Boosting توسعه دستهء طبقه بندهای D به صورت نموی می باشد و هر بار یک طبقه بند اظافه می گردد. طبقه بندی که در مرحلهء k ام به گروه ملحق می شود بر روی یک مجموعه از داده هایی که به صورت گزینشی از پایگاه داده آموزشی نمونه گیری شده است آموزش داده می شود. نمونه گیری از توزیع یکنواخت شروع می گردد و به سمت افزایش احتمال برای داده های سخت پیش می رود. بنا بر این توزیع داده ها در هر

مرحله به روز می شود، و احتمال داده هایی که در مرحلهء k-1 به صورت اشتباه تشخیص داده شده اند برای مرحلهء بعد افزایش می یابد. اینجا یک تناظر با Hedge(β) وجود دارد که ترنسپوز شده اند. طبقه بند های موجود در D، آزمایش یا رویدادهای ما هستند و داده ها در Z استراتژی یا راهبرد می باشند که احتمال داده ها در هر مرحله به روز می گردد. الگوریتم Adaboost در مرجع [۴۸] معرفی گردیده است که مخفف شده ADAptive BOOSTing resampling می باشد. دو مرحلهء اجرا برای Adaboost عبارتند از بازوزندهی[۱] و باز نمونه گیری[۲] در هر مرحله. توصیف بالا به باز نمونه گیری مربوط می باشد. برای اجرای دوباره وزن دهی ما فرض می کنیم که طبقه بند های پایه می توانند مستقیما از احتمالات بر روی Z به عنوان وزن ها استفاده کنند. نمونه گیری در این مورد نیاز نمی باشد، بنا بر این الگوریتم به طور کامل قطعی می گردد. Adaboost در آغاز برای مسائل دو کلاسی پیشنهاد گردید. و سپس برای چند کلاس توسعه یافت.

الگوریتم arc-x4

Breiman، bagging و boosting را از زوایای عجیبی مطالعه کرد. و یک کلاس از اللگوریتم boosting را arcing نامید که سر کلمه ای از Adaptive Resample and combinING می باشد. الگوریتم arc-fs او Adaboost می باشد.

Breiman یک الگوریتم boosting پیشنهاد داد که آن را arc-x4 نامید تا موفقیت زمینه های Adaboost را در جزئیات فنی آن یا استفاده آن را در طرح باز نمونه گیری بررسی کند. دو تفاوت بین Adaboost وarc-x4 وجود دارد، نخست وزن دادهء z_j در مرحلهء k ام محاسبه می گردد و متناسب با تعداد دفعاتی که z_j به صورت نادرست توسط k-1 طبقه بندی که تا کنون ساخته شده است طبقه بندی

شده است. دوم اینکه تصمیم نهائی از ائتلاف آراء ساخته می شود در عوض نظر اکثریت وزن دار گردیده است. الگوریتم arc-x4 در شکل زیر شرح داده شده است.

Breiman زمانی که Adaboost از یک تئوری سرچشمه گرفت، پذیرفت arc-x4 یک الگوریتم تک کاره و فاقد عمومیت می باشد [۴۹]. پارامتر الگوریتم، توان m_j توسط یک آزمایش یا تجربه عدد ثابت ۴ انتخاب گردید(به همین جهت arc-x4 نام گرفت).هنوز arc-x4 نسبت به Adaboost برتری دارد. بریمان رفتارهای Adaboost و arc-x4 را با هم مقایسه کرد و دریافت که Adaboost حرکت های تند و ناگهانی دارد در صورتی کهarc-x4 بیشتر رفتار آهسته و نرم دارد. این مطلب به عنوان مثال توسط انحراف استاندارد وزنهای اختصاص یافته به یک داده تکی منعکس می گردد. این انحراف استاندارد برای Adaboost خیلی بزرگتر ازarc-x4 است.

دلیل به کار گرفتن Adaboost چیست؟

یکی از توضیحات برای موفقیت Adaboost این خصوصیت الگوریتم می باشد که خطای مرحله آموزش را خیلی سریع به صفر می رساند، عملا در تعداد تکرار ها کمی این اتفاق می افتد.

مرز بالای خطای آموزش

فریوند و اسچاپیر یک مرز بالا برای خطای آموزش Adaboost پیدا کردند [۴۳]. برای مورد دو کلاسه تئوری زیر نتایج را بیان می کند:

تئوری ۲-۱: Ω را به صورت $\Omega = \{\omega_1, \omega_2\}$ و ε را خطای آموزش مجموعه تعریف می کنیم. $\varepsilon_i, i = 1,2, \dots, l$ خطا های وزن دار شده طبقه بند ها در D هستند که توسط رابطه ۴-۶ به دست میآیند داریم:

$$\varepsilon < 2^L \prod_{i=1}^{L} \sqrt{\varepsilon_i(1-\varepsilon_i)}$$

نتیجه تئوری فوق نشان می دهد با اظافه کردن تعداد بیشتر طبقه بند خطای آموزش برای مجموعه به سمت صفر میل میکند.

مرز خطای برای مورد چند کلاسه شبیه به مورد دو کلاسه می باشد تا وقتی که هر یک از طبقه بند ها خطای کوچکتر از ۰/۵ داشته باشند. Adaboost.M1 این شرط را در مرحلهء۲ لحاظ کرده است. با شرط اولیهء $\varepsilon_i \geq 0.5$ تئوری ۲-۲ در ادامه برای مورد عمومی c کلاسه ارائه گردیده است.

تئوری ۲-۲: با فرض $\Omega = \{\omega_1, \omega_2, \dots, \omega_c\}$ و ε به عنوان خطای آموزش مجموعه و $\varepsilon_i, i = 1,2, \dots, L$ خطای وزن دار شده برای طبقه بندهای موجود در D که با رابطهء ۴-۶ بیان گردیده و شرط $\varepsilon_i < 1/2$ داریم:

$$\varepsilon < 2^L \prod_{i=1}^{L} \sqrt{\varepsilon_i(1-\varepsilon_i)}$$

فصل سوم:

مروری بر روشهای انتخاب ویژگی

۳-۱ مقدمه

مساله انتخاب ویژگی، یکی از مسائلی است که در مبحث یادگیری ماشین و همچنین شناسائی آماری الگو مطرح است. این مساله 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. و علاوه بر این باعث می‌شود که اطلاعات غیر مفید زیادی را به همراه داده‌های مفید ذخیره کنیم.

برای مساله انتخاب ویژگی، راه حل‌ها و الگوریتم‌های فراوانی ارائه شده است که بعضی از آنها قدمت سی یا چهل ساله دارند [۵]. 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 این بخش ما در ابتدا تعاریفی که برای انتخاب ویژگی ارائه شده‌اند و همچنین، تعاریف مورد نیاز برای درک این مساله را ارائه می‌دهیم. سپس روش‌های مختلف برای این مساله را بر اساس نوع و ترتیب تولید زیرمجموعه ویژگی‌های کاندید و همچنین نحوه ارزیابی این زیرمجموعه‌ها دسته بندی می‌کنیم. سپس تعدادی از روش‌های معرفی شده در هر دسته را معرفی و بر اساس اهمیت، تا جائی که مقدور باشد، آنها را تشریح و الگوریتم برخی از آنها را ذکر می‌کنیم. لازم به ذکر است که بدلیل اینکه مبحث انتخاب ویژگی به مبحث طبقه بندی بسیار نزدیک است، بعضی از مسائلی که در اینجا مطرح می‌شود مربوط به مبحث طبقه بندی می‌باشد. توضیحات ارائه شده برای الگوریتم‌های مختلف در حد آشنائی است. شما می‌توانید برای کسب اطلاعات بیشتر به منابع معرفی شده مراجعه کنید.

۳-۲تعاریف

مساله 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 .

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، به طوریکه $M < N$ باشد و همچنین مقدار یک تابع معیار[1] برای زیرمجموعه مورد نظر، نسبت به سایر زیرمجموعه های هم-اندازه دیگر بهینه باشد. این تعریفی است که Fukunaga و Narenda در سال ۱۹۷۷ ارائه داده-اند.

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 می آید.

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.

[1] - Criterion function

روش‌های مختلف انتخاب ویژگی، تلاش می‌کنند تا از میان 2^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.

به طور کلی روش‌های مختلف انتخاب ویژگی را بر اساس نوع جستجو به دسته های مختلفی تقسیم بندی می‌کنند. در بعضی 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:

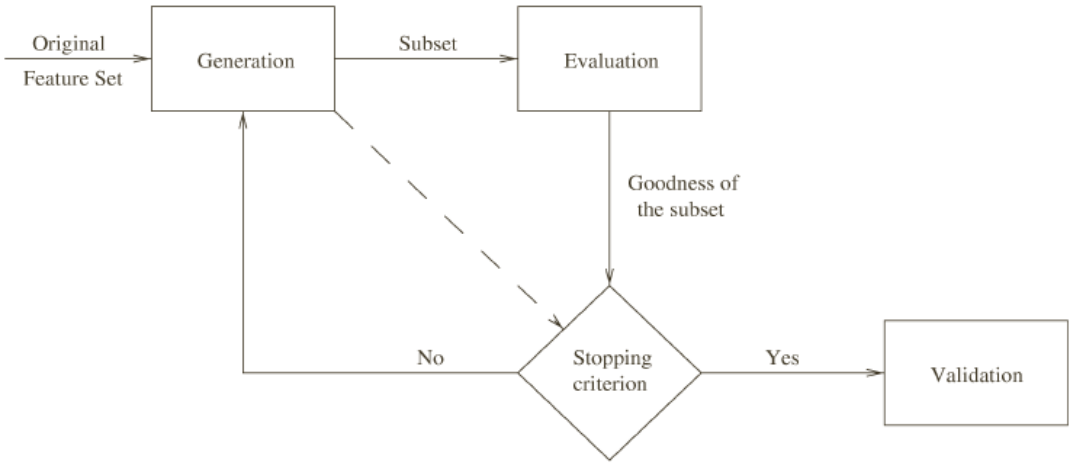
۱. **تابع تولید کننده**[1]: این تابع زیر مجموعه‌های کاندید را برای روش مورد نظر پیدا می‌کند.

۲. **تابع ارزیابی**[2]: 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**[3]: تصمیم می‌گیرد که آیا زیر مجموعه انتخاب شده معتبر است یا خیر؟

[1] - Generation procedure
[2] - Evaluation function
[3] - Validation procedure



شکل ۳-۱: 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 می گویند.

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 بازمی‌گرداند. این مقدار با بهترین زیرمجموعه قبلی مقایسه می‌شود. اگر زیر مجموعه 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) وقتیکه اضافه یا حذف کردن ویژگی، زیرمجموعه بهتری را تولید نکند

۲) وقتیکه 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 مساله مورد نظر ما را دارد یا نه؟ برای اینکار می‌توانیم از داده‌های نمونه‌برداری شده و یا مجموعه داده های شبیه سازی شده استفاده کنیم.

۳-۳ روش‌های مختلف انتخاب ویژگی

در این بخش ابتدا روش‌های مختلف انتخاب ویژگی را بر اساس دو معیار تابع تولید کننده و تابع ارزیابی طبقه بندی می‌کنیم. سپس آنها را بر اساس عملکرد دسته‌بندی و نحوه اجرای هر دسته را به اختصار شرح می‌دهیم.

۳-۳-۱ 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 زیر مجموعه، 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.

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 [۵۰]: "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 شده اند، بعضی از آنها از تکنیک بازگشت به

[1] - Complete Search

عقب (Backtracking) نیز در جریان کار استفاده کرده‌اند، مانند: branch and bound، best first search و beam search.

جستجوی مکاشفه ای[1]

در روش‌های با این نوع جستجو، در هر بار اجرای الگوریتم، یک ویژگی به مجموعه ویژگی 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.

جستجوی تصادفی[2]

روش‌هایی که از این نوع جستجو استفاده می‌کنند، محدوده کمتری از فضای کل حالات را جستجو می کنند، 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 می-دهد. 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.

۲-۳-۳ تابع ارزیابی

پیدا شدن یک زیرمجموعه بهینه از مجموعه ویژگی‌ها، به صورت مستقیم با انتخاب تابع ارزیابی بستگی دارد. چرا که اگر تابع ارزیابی به زیرمجموعه ویژگی بهینه یک مقدار نامناسب نسبت دهد، این زیرمجموعه هیچگاه

[1] - Heuristic Search
[2] Random Search

بعنوان زیرمجموعه بهینه انتخاب نمی‌شود. مقادیری که توابع ارزیابی مختلف به یک زیرمجموعه می دهند، با هم متفاوت است.

توابع ارزیابی را می‌توان به طرق مختلفی دسته بندی کرد. در اینجا ما دسته بندی‌ای که توسط Dash و Liu ارائه شده است [۵۱] را بیان می‌کنیم. آنها این معیارها را به پنج دسته تقسیم کرده‌اند:

۱. **معیارهای مبتنی بر فاصله**[1]: در این معیارها، مثلاً برای یک مساله 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 می آید 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 (Coefficient) 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

[1] - Distance Measures
[2] - Information Measures
[3] - Dependence Measures
[4] - Correlation

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 مساله نگاه می‌کنند، این کار را انجام نمی‌دهیم.

۴. **معیارهای مبتنی بر سازگاری**[1]: 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 ویژگی‌های 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**[2]: 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 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**[3]: اینکه بتوان زیرمجموعه انتخاب شده را برای طبقه‌بندی کننده‌های متفاوت به کار ببریم.

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

---

[1] - Consistency Measures
[2] - Classifier Error Rate Measures
[3] - Generality

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.

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

شکل ۳-۲: مقایسه توابع ارزیابی مختلف [۵]

۳-۳-۳ دسته بندی و تشریح الگوریتم های مختلف انتخاب ویژگی

در این قسمت بر اساس تابع ارزیابی و تابع تولید کننده، روش‌های مختلف انتخاب ویژگی را به چند دسته تقسیم بندی می‌کنیم و سپس تعدادی از روش‌ها را شرح داده و الگوریتم کار را به صورت شبه کد، ذکر می-کنیم.

قبل از اینکه بحث را ادامه دهیم، لازم است که متغیرهای به کار رفته در شبه کدها را معرفی کنیم. این متغیرها و شرح آنها به صورت زیر می‌باشد:

- **متغیر D**: مجموعه آموزشی
- **متغیر S**: مجموعه ویژگی اصلی (شامل تمام ویژگی‌ها)
- **متغیر N**: تعداد ویژگی‌ها
- **متغیر T**: زیرمجموعه ویژگی انتخاب شده
- **متغیر M**: تعداد ویژگی‌های انتخاب شده یا تعداد ویژگی‌هایی که لازم است انتخاب شوند.

تابع ارزیابی مبتنی بر فاصله - تابع تولید کننده مکاشفه ای

مهمترین روش در این گروه Relief است. در اینجا ما ابتدا این روش را به عنوان نماینده این گروه شرح می-دهیم، سپس یک مرور مختصری بر سایر روش‌ها خواهیم داشت.

روش Relief از یک راه حل آماری برای انتخاب ویژگی استفاده می‌کند، همچنین یک روش مبتنی بر وزن است که از الگوریتم‌های مبتنی بر نمونه الهام گرفته است. روش کار به این صورت است که از میان مجموعه نمونه‌های آموزشی، یک زیرمجموعه نمونه انتخاب می‌کنیم. کاربر بایستی تعداد نمونه‌ها(NoSample) در این زیرمجموعه را مشخص کرده باشد. و آنرا به عنوان ورودی به الگوریتم ارائه دهد. الگوریتم به صورت تصادفی یک نمونه از این زیرمجموعه را انتخاب می‌کند، سپس برای هر یک از ویژگیهای این نمونه، نزدیکترین برخورد[1] و نزدیکترین شکست[2] را بر اساس معیار اقلیدسی پیدا می‌کند.

نزدیکترین برخورد نمونه‌ای است که کمترین فاصله اقلیدسی را در میان سایر نمونه‌های هم‌کلاس با نمونه انتخاب شده دارد. نزدیکترین شکست نیز نمونه‌ای است که کمترین فاصله اقلیدسی را در میان نمونه‌هایی که هم‌کلاس با نمونه انتخاب شده نیستند، دارد.

[1] - Nearest Hit
[2] - Nearest Miss

ایده اصلی در این الگوریتم این است که هر چه اختلاف بین اندازه یک ویژگی در نمونه انتخاب شده و نزدیکترین برخورد کمتر باشد، این ویژگی بهتر است و بعلاوه یک ویژگی خوب آن است که اختلاف بین اندازه آن ویژگی و نزدیکترین شکست وی بیشتر باشد. دلیل کار هم خیلی ساده است، ویژگی‌هایی که به خوبی دو کلاس (یا یک کلاس از سایر کلاس‌ها) را از هم تمییز می‌دهند، برای نمونه‌های متعلق به دو کلاس متفاوت مقادیری نزدیک به‌هم نمی‌دهند و یک فاصله معنی‌داری بین مقادیری که به نمونه‌های یک کلاس می‌دهند و مقادیری که به سایر کلاس(ها) می‌دهند وجود دارد.

الگوریتم پس از تعیین نزدیکترین برخورد و نزدیکترین شکست، وزن‌های ویژگی‌ها را به روزرسانی می‌کند، این به‌روزرسانی به این صورت است که مربع اختلاف بین مقدار ویژگی مورد نظر در نمونه انتخاب شده و نمونه نزدیکترین برخورد از وزن ویژگی کم می‌شود و در عوض مربع اختلاف بین مقدار ویژگی در نمونه انتخاب شده و نزدیکترین شکست به وزن ویژگی اضافه می‌شود. هر چه مقدار این وزن بزرگتر باشد، ویژگی مورد نظر، بهتر می‌تواند نمونه‌های یک کلاس را از دیگران جدا کند.

بعد از تعیین فاصله برای تمام نمونه‌های موجود در مجموعه نمونه‌ها، الگوریتم ویژگی‌هایی را که وزن آنها کمتر یا مساوی با یک حد آستانه‌ای است، را حذف می‌کند، و سایر ویژگی‌ها بعنوان زیرمجموعه ویژگی جواب باز می‌گردند. مقدار حد آستانه‌ای توسط کاربر تعیین می‌گردد، البته ممکن است که بصورت اتوماتیک بوسیله یک تابعی از تعداد کل ویژگی‌ها تعیین شود و یا اینکه با سعی و خطا تعیین گردد. همچنین می‌توان ویژگی‌هایی که وزن آنها منفی است را حذف کرد.

الگوریتم Relief برای ویژگی‌های دارای نویز یا ویژگی‌های دارای همبستگی خوب کار می‌کند و پیچیدگی زمانی آن بصورت خطی و تابعی از تعداد ویژگی‌ها و تعداد نمونه‌های مجموعه نمونه می‌باشد. و هم برای داده‌های پیوسته و هم برای داده‌های صوری خوب کار می‌کند [۵۲].

یکی از محدودیت‌های اساسی این الگوریتم این است که ویژگی‌هایی که دارای افزونگی باشند را پیدا نمی‌کند و بنابراین مجموعه‌های غیر بهینه را پیدا می‌کند که دارای افزونگی هستند. این مشکل را می‌توان با یک جستجوی تعیین جامعیت برای زیرمجموعه‌های تولید شده توسط الگوریتم حل کرد. علاوه بر این مشکل دیگر این الگوریتم این است که با مسائل دو کلاسه خوب کار می‌کند. این محدودیت نیز با الگوریتم Relief-F [۵] مرتفع شده است، با الگوریتم جدید مشکل داده‌های غیر کامل (نمونه‌های آموزشی غیرکامل) نیز حل شده است.

روشی که Jakub Segen [۵۳] برای انتخاب ویژگی مطرح کرده است، از یک تابع ارزیابی استفاده می‌کند که مجموع یک معیار اختلاف آماری و یک معیار پیچیدگی ویژگی را محاسبه کرده و آنرا مینیمم می‌کند. این الگوریتم، اولین ویژگی را که بهتر بتواند کلاس‌ها را از هم تمییز دهد را پیدا می‌کند. سپس ویژگی‌هایی را پیدا می‌کند، که در ترکیب با ویژگی‌های انتخاب شده، جدائی‌پذیری کلاس‌ها را افزایش دهند. این فرآیند زمانی متوقف می‌شود که به حداقل معیار بازنمائی مورد انتظار برسیم.

تابع ارزیابی مبتنی بر فاصله – تابع تولید کننده کامل

استفاده از این ترکیب در روش‌های قدیمی نظیر B&B (Branch and Bound) یافت می‌شود. سایر روش‌های این گروه، نسخه‌های متفاوتی از B&B هستند. به این ترتیب که یا یک تابع تولید کننده دیگری را استفاده کرده‌اند (BFF [54]) و یا اینکه از یک تابع ارزیابی متفاوتی استفاده کرده‌اند (Bobrowski’s

[55] method). در اینجا ابتدا به شرح B&B می‌پردازیم و سپس یک شرح مختصری در مورد دو روش دیگر ارائه می‌دهیم.

تعریف کلاسیک ارائه شده بوسیله Fukunaga و Narenda از انتخاب ویژگی، احتیاج دارد که تابع ارزیابی یکنوا باشد. یعنی اگر دو زیرمجموعه ویژگی A و B با اندازه‌های M و N موجود باشند، و $B \subset A$ در اینصورت مقدار تابع ارزیابی برای A نباید بیشتر از مقدار تابع برای B باشد. این تعریف باعث ایجاد مشکل در مسائل دنیای واقعی می‌شود، زیرا اندازه تخمینی زیرمجموعه ویژگی بهینه در حالت عمومی ناشناخته است.

البته به سادگی می‌توان این تعریف را تغییر داد تا با مسائل عمومی سازگار شود، به اینصورت که می‌گوئیم: الگوریتم‌های مشابه B&B تلاش می‌کنند که دو شرط زیر را همزمان ارضاء کنند:

۱. زیرمجموعه ویژگی جواب تا حد امکان کوچک باشد.

۲. یک کران برای مقدار تابع ارزیابی را در نظر بگیرد. (یا یک اندازه مینیمم برای تعداد ویژگی‌های انتخاب شده مثلاً بهترین زیرمجموعه ویژگی سه عنصری)

بوسیله کران تعیین شده، فضای جستجو تا حد امکان کوچک می‌شود. به این ترتیب الگوریتم B&B از یک زیرمجموعه شامل تمام ویژگی‌های موجود شروع می‌کند و درخت جستجو را تشکیل می‌دهد. در این درخت در ریشه تمام ویژگی‌ها قرار دارند و فرزندان وی، زیرمجموعه‌هایی هستند که زیرمجموعه، گره پدر هستند و از حذف تنها یکی از عناصر پدرشان تشکیل شده‌اند. این روند برای سایر گره‌های درخت تکرار می‌شود تا به مجموعه‌ها تک عنصری (یا تعداد ویژگی‌های تعیین شده بعنوان کران) برسیم. یعنی برگ‌های درخت مجموعه-های تک عنصری هستند و ریشه درخت یک مجموعه شامل همه ویژگی‌های موجود.

با توجه به این خاصیت که تمام زیرمجموعه‌های یک مجموعه مقدار کمتری برای تابع ارزیابی دارند، در حین جستجو اگر یک گره به واسطه کم بودن مقدار تابع ارزیابی انتخاب نشد، زیرشاخه‌های آنرا برای یافتن جواب جستجو نمی‌کنیم، چون قطعاً تابع ارزیابی مقدار کمتری را برای آنها باز می‌گرداند.

عموماً توابع ارزیابی زیر برای اینکار استفاده می‌شوند:

- فاصله ماهالانوبیس (Mahalanobis Distance)
- تابع جداساز (Discriminant Function)
- معیار فیشر (Fisher Criterion)
- فاصله باتاچاریا (Bhattacharya)
- Divergence

یک الگوریتم مشابه برای انتخاب ویژگی BFF است، در این الگوریتم، تابع جستجو به این صورت تغییر کرده است که مشابه حل مساله جستجوی یک مسیر بهینه در یک درخت وزن‌دار با یک استراتژی تغییر یافته از Best first search است. این الگوریتم تضمین می‌کند که بهترین هدف(زیرمجموعه بهینه) بدون از دست دادن جامعیت مساله پیدا شود، البته با ارضای معیار یکنوا بودن تابع ارزیابی.

تابع ارزیابی مبتنی بر اطلاعات - تابع تولید کننده مکاشفه ای

در این دسته دو روش وجود دارد:

۱) روش درخت تصمیم(DTM)

در روش درخت تصمیم، نمونه‌ها به یک الگوریتم C4.5 [۵۶]، که یکی از درختهای تصمیم‌گیری است اعمال می‌شوند، سپس درخت هرس شده حاصل از الگوریتم C4.5 را گرفته و کلیه ویژگی‌هایی که در آن وجود دارد را بعنوان جواب مساله باز می‌گردانیم.

الگوریتم C4.5، از یک تابع مکاشفه بر پایه اطلاعات استفاده می‌کند.

۲) روش استفاده شده توسط Koller و Sahami

روش استفاده شده توسط Koller و Sahami که اخیراً ارائه شده است، بر این پایه استوار است که ویژگی-هایی که داده مفید چندانی را در بر ندارند و یا اصلاً داده مفیدی را در اختیار قرار نمی‌دهند و می‌توان آنها را با سایر ویژگی‌ها نمایش داد، یا ویژگی‌هایی که بی‌ربط هستند و یا داده اضافی هستند، را شناسائی و حذف می-کنیم. برای پیاده سازی این مطلب، تلاش می‌کنیم تا با پوشش مارکوف آنها را پیدا کنیم، به این صورت که یک زیرمجموعه مانند T، یک پوشش مارکوف برای ویژگی f_i است، اگر f_i برای زیرمجموعه T بصورت مشروط هم از کلاس و هم از سایر ویژگی‌هایی که در T نیستند، مستقل باشد [۵۷].

تابع ارزیابی مبتنی بر اطلاعات – تابع تولید کننده کامل

مهمترین روشی که در این گروه می‌توانیم پیدا کنیم، روش Minimum Description Length Method (MDLM) است [۵۶]. نویسندگان این روش تلاش می‌کنند تا همه ویژگی‌های بدون استفاده[1] (بی‌ربط یا اضافی) را حذف نمایند، با این دید که اگر ویژگی‌های زیرمجموعه V را بتوانیم بصورت یک تابع ثابتی مانند F که وابسته به کلاس نیست، بر اساس یک زیرمجموعه ویژگی دیگر مانند U بیان کنیم. در این صورت وقتی که مقادیر ویژگی‌های زیرمجموعه U شناخته شده باشند، ویژگی‌های موجود در زیرمجموعه V بدون استفاده هستند.

از دیدگاه انتخاب ویژگی، اجتماع دو زیرمجموعه U و V، مجموعه کامل، شامل تمام ویژگی‌ها را تشکیل می-دهد. و کاری که ما باید در انتخاب ویژگی انجام دهیم این است که این دو زیرمجموعه را جدا کنیم. برای انجام این کار، نویسندگان MDLM، از معیار Minimum Description Length Criterion (MDLC) که بوسیله Rissanen ارائه شده است [۵۸]، استفاده کرده‌اند. آنها فرمولی را بدست آورده‌اند، که شامل تعداد

[1] - Useless

بیتهای لازم برای انتقال کلاسها، پارامترهای بهینه سازی، ویژگی‌های مفید و ویژگی‌های غیرمفید است. الگوریتم تمام زیرمجموعه‌های ممکن (2N) جستجو می‌کند و بعنوان خروجی زیرمجموعه‌ای را بازمی‌گرداند که معیار MDLC را ارضا کند. این روش می‌تواند تمام ویژگی‌های مفیدی را پیدا کند که دارای توزیع نرمال باشند. برای حالتهای غیر نرمال این روش قادر نیست، ویژگی‌های مفید را پیدا کند. الگوریتم زیر روش کار و فرمول‌های استفاده شده را نشان می‌دهد.

تابع ارزیابی مبتنی بر وابستگی - تابع تولید کننده مکاشفه ای

دو روش عمده در این گروه می‌بینیم:

۱) Probability of Error & Average Correlation Coefficient (POE1ACC)

که خود شامل هفت روش است [۵۹]، ما در اینجا روش هفتم را که به گفته نویسنده کاملتر است را بررسی می‌کنیم.

در این روش اولین ویژگی به این صورت تعیین می‌شود که احتمال خطا را برای تمام ویژگی‌ها محاسبه می‌کنیم، ویژگی با کمترین احتمال خطا (P_e)، به عنوان اولین ویژگی انتخاب می‌شود. ویژگی بعدی، آن ویژگی است که مجموع وزن‌دار P_e و میانگین ضریب همبستگی(ACC) با ویژگی(های) انتخاب شده را مینیمم کند. سایر ویژگی‌ها به همین ترتیب انتخاب می‌شوند. میانگین ضریب همبستگی به اینصورت است که میانگین ضریب همبستگی ویژگی کاندید با ویژگی‌های انتخاب شده در آن نقطه محاسبه می‌شوند. این روش می‌تواند تمام ویژگی‌ها را بر اساس مجموع وزن‌دار درجه‌بندی کند. شرط خاتمه نیز در این روش تعداد ویژگی‌های مورد نیاز خواهد بود.

۲) روش PreSet

این روش از تئوری مجموعه‌های ناهموار[1] استفاده می‌کند. در اینجا یک کاهش[2] پیدا مـی‌کنـیم. یـک کـاهش مانند R از یک مجموعه P به این معنی است که نمونه‌ها بوسیله آن به خوبی مجموعه P طبقه بندی شوند. پس از پیدا کردن یک کاهش، تمام ویژگی‌هایی که در مجموعه کاهش داده شده وجود ندارند، را از مجموعه ویژگـی حذف می‌کنیم. سپس ویژگی‌ها را بر اساس اهمیت آنها درجه‌بندی می‌کنیم. اهمیت یک ویژگی بـر ایـن اسـاس بیان می‌شود که یک ویژگی در جریان کلاس‌بندی چقدر اهمیت دارد. این معیار بر پایه صفات وابستگی ویژگـی تعیین می‌گردد .

تابع ارزیابی مبتنی بر سازگاری - تابع تولید کننده کامل

روش‌هایی که در این گروه قرار دارند، در سالهای اخیر ارائه شده‌اند. ما به صورت مختصر سه روش این گروه را بررسی می‌کنیم ولی بحث اصلی ما بر روی روش اول است.

۱) روش Focus

این روش یک حداقل گرا است، به این معنی که سعی می‌کند که حداقل تعداد ویژگی ممکن را برای ارائه پیدا کند. این روش فرضیه‌های قابل تعریف را بررسی کرده و فرضیه‌ای که بتواند سازگاری را با حداقل تعداد ویژگـی ممکن برقرار کند را بعنوان جواب بازمی‌گرداند.

در ساده‌ترین پیاده سازی برای این روش، برای یافتن یک ناسازگاری با یک زیرمجموعه از ویژگی‌های انتخـاب شده، درخت جستجو را به صورت سطح به سطح (جستجو در پهنا)، پیمایش مـی‌کنـیم. در ایـن جریـان کـه از مجموعه‌های کوچکتر شروع می‌شود، در صورتی که به یک ناسازگاری برسیم، مجموعه انتخاب شده رد می‌شـود

[1] - Rough
[2] - Reduct

و جستجو با مجموعه بعدی ادامه می‌یابد. به محض اینکه به یک مجموعه برسیم که ناسازگاری نداشته باشد، جستجو متوقف شده و مجموعه یاد شده به عنوان جواب انتخاب می‌شود.

می‌توان گفت که این روش به نویز حساس است و نمی‌تواند نویز را مدیریت کند، زیرا در صورتی که نویز وجود داشته باشد، هیچ زیرمجموعه‌ای را نمی‌توان پیدا کرد که ناسازگاری نداشته باشد و الگوریتم تمام ویژگی‌ها را به عنوان جواب بازمی‌گرداند. با یک تغییر کوچک می‌توانیم این مساله را حل کنیم، به اینصورت که اجازه می‌دهیم یک میزان معینی از ناسازگاری در مجموعه انتخاب شده وجود داشته باشد.

۲) روش Schlimmer

این روش از یک شمارش سیستماتیک برای تابع تولید کننده و یک معیار ناسازگاری نیز به عنوان تابع ارزیابی استفاده می‌کند. همچنین با استفاده از یک تابع مکاشفه‌ای سرعت جستجو را برای یافتن زیرمجموعه بهینه افزایش می‌دهد. این تابع مکاشفه‌ای یک معیار قابلیت اعتماد است، بر پایه این ایده که احتمال مشاهده یک ناسازگاری مشاهده شود، نسبتی از درصد مقادیری است که زیاد مشاهده شده‌اند [۵۰].

۳) روش MIFES1

این روش در انتخاب ویژگی شباهت زیادی به روش Focus دارد. در اینجا مجموعه نمونه‌ها را به شکل یک ماتریس ارائه می‌دهیم، هر عنصر نماینده یک ترکیب یکتا از یک نمونه منفی و یک نمونه مثبت است. یک ویژگی مانند f یک پوشش برای یک نمونه از ماتریس نامیده می‌شود، اگر برای نمونه‌های منفی و نمونه‌های مثبت، مقادیر عکسی داشته باشد. این روش از یک پوشش با همه ویژگی‌ها شروع می‌کند، و تکرار می‌شود تا وقتی که هیچ کاهشی نتوانیم برای پوشش داشته باشیم. مشکل اساسی این روش اینست که فقط برای مسائل

دو کلاسه و ویژگی‌های منطقی قابل استفاده است. توضیحات کاملتر راجع به پوشش و نحوه اجرای الگـوریتم را می‌توانید در مرجع شماره [۶۰] پیدا کنید.

تابع ارزیابی مبتنی بر سازگاری - تابع تولید کننده تصادفی

نماینده این گروه که جدیدتر هستند، روش LVF است [۶۱]. این روش فضای جستجو را بصورت تصادفی بـا استفاده از یک الگوریتم Las Vegas جستجو می‌کند، که یکسری انتخاب‌های احتمالی انجـام مـی‌دهـد تـا بـا استفاده از آنها سریعتر به جواب بهینه برسیم، همچنین از یک معیار سـازگاری کـه بـا معیـار اسـتفاده شـده در الگوریتم Focus متفاوت است.

این روش برای هر زیرمجموعه کاندید، تعداد ناسازگاری را محاسبه می‌کند، که بر این ایـده اسـتوار اسـت کـه کلاس محتمل‌تر آن است که در میان نمونه‌های این زیرمجموعه ویژگی، تعـداد بیشـری متعلـق بـه آن کـلاس باشند. یک حد آستانه‌ای برای ناسازگاری در نظر گرفته می‌شود، که در ابتدا ثابت و بصـورت پـیش فـرض صـفر است و هر زیرمجموعه‌ای که مقدار ناسازگاری آن بیشتر باشد، رد می‌شود.

این روش می‌تواند هر زیرمجموعه بهینه را پیدا کند، حتی برای داده‌های دارای نویز، به شرط آنکه سطح نـویز درست را در ابتدا مشخص کنیم. یک مزیب این روش اینست که احتیاجی نیست که کاربر مدت زیادی را بـرای بدست آوردن یک زیرمجموعه بهینه منتظر بماند، زیرا الگوریتم هر زیرمجموعه‌ای که بهتر از بهترین جواب قبلی باشد (هم از لحاظ اندازه زیرمجموعه انتخاب شده و هم از لحاظ نرخ سازگاری)، را به عنوان جواب باز می‌گرداند.

این الگوریتم کارا است، زیرا تنها مجموعه‌هایی برای ناسازگاری تست می‌شوند، که تعداد ویژگی‌های درون آن کمتر یا مساوی بهترین زیرمجموعه‌ای است که تا کنون پیدا شده است. پیاده سازی آن آسان است و پیدا شدن زیرمجموعه بهینه را اگر منابع موجود اجازه دهند، تضمین می‌کند. یکی از مشکلات این الگوریتم این اسـت کـه

برای پیدا کردن جواب بهینه زمان بیشتری نسبت به الگوریتم‌هایی که از توابع تولید کننده مکاشفه‌ای استفاده می‌کنند، نیاز دارد، چون این الگوریتم از دانش مربوط به زیرمجموعه‌های قبلی استفاده نمی‌کند.

تابع ارزیابی مبتنی بر خطای طبقه بندی کننده- تابع تولید کننده مکاشفه ای

همانطور که قبلاً نیز اشاره کردیم، به مجموعه روش‌هایی که از تابع ارزیابی مبتنی بر نرخ خطای طبقه‌بندی کننده استفاده می‌کنند، (بدون توجه به نوع تابع تولید کننده استفاده شده) روش‌های wrapper می‌گویند. در این گروه روش‌های مشهور زیر را می‌توانیم ببینیم:

۱) روش SFS (Sequential Forward Selection)

این روش، کارش را با یک مجموعه خالی شروع می‌کند، سپس در هر تکرار یک ویژگی با استفاده از تابع ارزیابی مورد استفاده، به مجموعه جواب اضافه می‌کند، این کار را تکرار می‌کند تا زمانیکه تعداد ویژگی لازم انتخاب شود. مشکلی که این روش با آن روبروست، اینست که ویژگی اضافه شده در صورتیکه مناسب نباشد، از مجموعه جواب حذف نمی‌شود [۶۲].

۲) روش SBS (Sequential Backward Selection)

این روش برعکس SFS کارش را با مجموعه‌ای شامل تمام ویژگی‌ها شروع می‌کند و در هر بار تکرار الگوریتم، ویژگی که بوسیله تابع ارزیابی انتخاب می‌شود، را از مجموعه مورد نظر حذف می‌کند. این کار را تا زمانی ادامه می‌دهد که تعداد ویژگی‌ها برابر یک تعداد معینی شود. مانند روش قبل مشکل این روش اینست که ویژگی حذف شده را دیگر به مجموعه اضافه نمی‌کند، حتی اگر مناسب باشد [۶۲].

روش‌های دیگری که در این گروه وجود دارند، نسخه‌های متفاوتی از دو روش قبلی یا ترکیب آنها هستند.

۳) روش SBS-Slash

این روش بر پایه این مشاهده است که هنگامی که تعداد زیادی ویژگی وجود دارد، بعضی از طبقه بندی کننده‌ها (مانند ID3 یا C4.5) مکرراً تعداد زیادی از آنها را استفاده نمی‌کنند. الگوریتم با یک مجموعه ویژگی کار خودش را شروع می‌کند (مانند SBS)، اما بعد از یک مرحله تمام ویژگی‌هایی را که در این مرحله یاد گرفته است و استفاده نشده‌اند، را حذف (Slashes) می‌کند [۶۳].

۴) روش PQSS ((p,q) Sequential Search)

در اینجا از بعضی از خواص بازگشت به عقب[1] استفاده می‌کنیم. عملکرد الگوریتم به این صورت است که در هر مرحله p ویژگی به مجموعه اضافه و q ویژگی از آن حذف می‌کند. حال اگر الگوریتم از مجموعه خالی شروع - کرده باشد، بایستی اندازه p بزرگتر از اندازه q باشد. ولی اگر از مجموعه تمام ویژگی‌ها شروع شده باشد، بایستی اندازه p کوچکتر از q باشد [۶۴].

۵) روش BDS (Bi-Directional Search)

مانند روش‌های قبل است با این تفاوت که جستجو را از دو طرف انجام می‌دهد [۶۴].

۶) روش Schemata Search

الگوریتم کارش را با مجموعه خالی و یا مجموعه تمام ویژگی‌ها شروع می‌کند و در هر تکرار، بهترین زیرمجموعه را با حذف یا اضافه تنها یک ویژگی به مجموعه ویژگی، پیدا می‌کند. برای اینکه هر زیرمجموعه را ارزیابی کند، از تعیین اعتبار Leave-One-Out Cross Validation (LOOCV) استفاده می‌کند. در هر

[1] - Backtracking

تکرار زیرمجموعه‌ای انتخاب می‌شود که کمترین خطای LOOCV را داشته باشد. کار به اینصورت ادامه می‌یابد تا هیچ تغییر با تک ویژگی نتواند باعث بهتر شدن زیرمجموعه شود [۶۵].

۷) روش RC (Relevance in Context)

در اینجا این واقعیت تشریح شده است که بعضی از ویژگی‌ها فقط در قسمتی از فضای کار مربوط هستند. روش کار مشابه SBS است، اما با تغییرات عمده‌ای که باعث محلی شدن آن شده است(انتخاب ویژگی‌های مرتبط بر اساس تصمیم گیری بوسیله نمونه‌ها) [۶۶].

۸) روش Queiros and Gelsema

شبیه SFS است اما پیشنهاد می‌کند که در هر تکرار، هر ویژگی در با تنظیمات متفاوتی بوسیله اثرات متفاوت ناشی از ویژگی‌های قبلی ارزیابی شود [۶۷]. دو نمونه از این تنظیمات به اینصورت هستند:

i) همیشه فرض کنیم که ویژگی‌ها مستقل هستند (ویژگی‌های قبلی را در نظر نمی‌گیریم).

ii) هیچگاه فرض نمی‌کنیم که ویژگی‌ها مستقل هستند (ویژگی‌های قبلی را در نظر می‌گیریم).

در این روش و تعدادی از روش‌های قبلی در این گروه از نرخ خطای بیز به عنوان خطای طبقه بندی کننده استفاده می‌کنیم.

تابع ارزیابی مبتنی بر خطای طبقه بندی کننده - تابع تولید کننده کامل

در این گروه چهار روش وجود دارد، که دو روش اول آن بوسیله Ichino و Sklansky ارائه شده است.

۱) روش Linear Classifier

۲) روش Box Classifier

در دو روش فوق مساله انتخاب ویژگی بوسیله برنامه نویسی صفر و یک حل شده است [۶۸, ۶۹].

۳) روش AMB&B (Approximate monotonic branch and bound)

این روش برای حل مشکلات B&B ارائه شده است، به این صورت که به تابع ارزیابی اجازه داده می‌شود که غیر یکنوا باشد [۷۰]. در اینجا به تابع تولید کننده اجازه داده می‌شود که زیرمجموعه‌هایی تولید کند که محدودیت تعیین شده را نقض می‌کنند، اما زیرمجموعه‌ای که بعنوان جواب انتخاب می‌شود، نباید محدودیت ذکر شده را نقض کرده باشد.

۴) روش BS (Beam Search)

این روش یک نمونه از جستجوی Best-First، است که از صف محدود شده برای محدود کردن فضای جستجو استفاده می‌کند. صف از بهترین به بدترین مرتب می‌شود، در اینصورت، بهترین زیرمجموعه در ابتدای صف قرار داده می‌شود. تابع تولید کننده به این صورت عمل می‌کند که زیرمجموعه موجود در ابتدای صف را انتخاب و کلیه زیرمجموعه‌های ممکن با اضافه کردن یک ویژگی به آن را تولید می‌کند و آنها را در محل مناسبشان در صف قرار می‌دهد. در صورتی که هیچ محدودیتی در اندازه صف نداشته باشیم، این روش یک جستجوی جامع است. در حالتی که محدودیت طول برابر یک را برای صف داشته باشیم، این روش با SFS برابر است.

تابع ارزیابی مبتنی بر خطای طبقه بندی کننده - تابع تولید کننده تصادفی

در این گروه پنج روش وجود دارد، که به شرح ذیل می‌باشند.

۱) روش LVW

این روش زیرمجموعه‌هایی به صورت کاملاً تصادفی با استفاده از الگوریتم Las Vegas تولید می‌کند [۷۰].

۲) روش الگوریتم ژنتیک GA (Genetic Algorithm)

در این روش یک جمعیت از زیرمجموعه‌های کاندید تولید می‌کنیم. در هر بار تکرار الگوریتم، با استفاده از عملگرهای جهش و بازترکیبی بر روی عناصر جمعیت قبلی، عناصر جدیدی تولید می‌کنیم. با استفاده از یک تابع ارزیابی، میزان شایستگی عناصر جمعیت فعلی را مشخص کرده و عناصر بهتر را به عنوان جمعیت نسل بعد انتخاب می‌کنیم. پیدا شدن بهترین جواب در این روش تضمین نمی‌شود ولی همیشه یک جواب خوب به نسبت مدت زمانی که به الگوریتم اجازه اجرا داده باشیم، پیدا می‌کند.

۳) روش SA (Simulated Annealing)

در اینجا نیز مانند الگوریتم ژنتیک، تابع تولید کننده آن از تولید تصادفی استفاده می‌کند ولی در تولید تصادفی از یک جریان خاصی پیروی می‌کند.

۴) روش RGSS (Random Generation plus Sequential Selection)

این روش مشابه SBS و SFS است با این تفاوت که یک زیرمجموعه تصادفی تولید می‌کند و سپس SBS و SFS را با استفاده از این زیرمجموعه تولید شده اجرا می‌کند. در واقع فاکتور تصادف را به دو روش ذکر شده تزریق می‌کند، تا کارآئی آنها را افزایش دهد.

۵) روش RMHC-PF1 (Random Mutation Hill Climbing-Prototype and Feature selection)

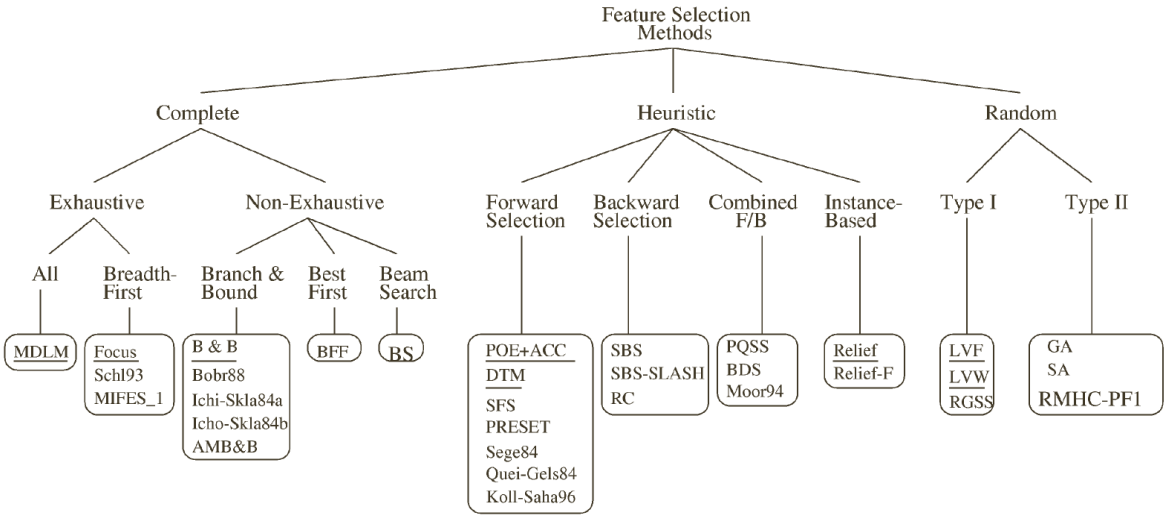
نمونه‌های اولیه[1] و ویژگی‌ها در اینجا همزمان برای استفاده در طبقه بندی کننده نزدیکترین همسایه انتخاب می‌شوند، همچنین برای ثبت نمونه‌های اولیه و ویژگی‌ها از یک بردار شرطی استفاده می‌شود. تابع ارزیابی نیز، نرخ خطای طبقه‌بندی کننده نزدیکترین همسایه می‌باشد. در هر تکرار، بصورت تصادفی یکی از بیتهای بردار جهش داده می‌شوند، تا یک بردار جدید برای تکرار بعدی تولید شود.

تمام روش‌های این گروه پارامترهای زیادی دارند که بایستی تنظیم شود، مثلاً LVW حد آستانه‌ای برای نرخ ناسازگاری، در الگوریتم‌های ژنتیک اندازه جمعیت اولیه، نرخ بازترکیبی و نرخ جهش و یا در SA، تعداد تکرار حلقه، دمای اولیه و احتمال جهش. تنظیم دقیق این پارامترها عملکرد این الگوریتم‌ها را بهبود می‌بخشد.

۳-۳-۴ جمع بندی روشهای انتخاب ویژگی

برای اینکه یک جمع‌بندی از کلیه روش‌های انتخاب ویژگی داشته باشیم، نمودار آنها را برحسب سه نوع تابع تولید کننده در شکل زیر نشان داده‌ایم.

[1] - Prototype

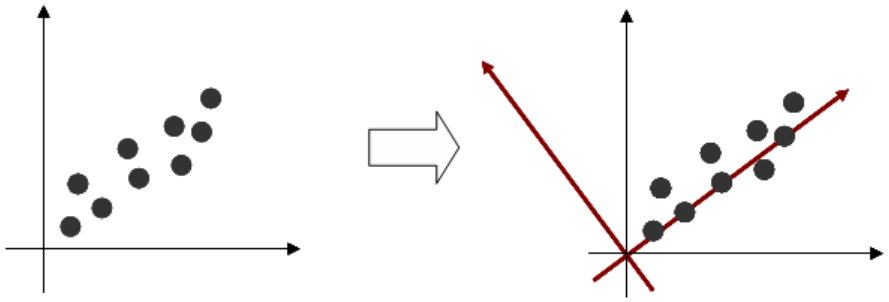


شکل ۵-۱۰: طبقه بندی روش‌های مختلف انتخاب ویژگی[۵۱]

Principal Component Analysis (PCA)۴-۳

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 [۳۰]. 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. در شکل زیر ایـن مطلـب بـرای داده‌هـای دو بعدی نشان داده شده است.



شکل ۳-۳: انتخاب محورهای جدید برای داده‌های دو بعدی

روش PCA به نامهای دیگری نیز معروف است. مانند:

- Karhunen Loeve Transform (KLT)
- Hotelling Transform
- Empirical Orthogonal Function (EOF)

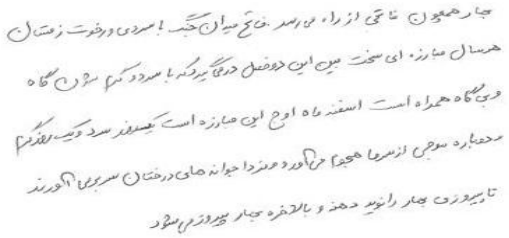
قبل از اینکه به جزئیات این روش بپردازیم ابتدا مفاهیم ریاضی و آماری مرتبط با ایـن روش را بطـور مختصـر بیان می‌کنیم. این مفاهیم شامل انحراف از معیار استاندارد، کواریانس، بردارهای ویژه و مقادیر ویژه می‌باشد.

فصل چهارم:

روش پیشنهادی و آزمایش‌های انجام شده

۴-۱ معرفی پایگاه داده

برای تحقیق انجام شده در این گزارش با توجه به اینکه میخـواهیم از آنـالیز بافـت بـرای تشـخیص هویـت استفاده کنیم، از پایگاه داده‌ای که توسط دهقان [۲] جمع‌آوری شده است، استفاده نموده‌ایم. پایگاه داده‌ای که در این مرجع استفاده گردیده است شامل دستخط ۵۰ نفر در سنین مختلف و با جنسیت های مختلف می‌باشد که ما برای تحقیق خود از دستخط ۴۰ نفر اول آنها استفاده نموده‌ایم که از هر نفر ۱۰ نمونه در دسـترس مـی-باشد. در پایگاه داده معرفی شده تمام دستنوشته‌ها با یک خودکار آبی مشترک نوشته شده‌اند. تمام ۴۰۰ نمونـه دستنوشته موجود، توسـط یـک پویشـگر تصـویر مناسـب و بـا دقـت 300dpi و بصـورت ۸ بیتـی و در سـطح خاکستری رقمی شده‌اند [۲]. در تصویر شکل ۲ یک تصویر از دست‌نوشته های موجود بـرای نمونـه آورده شـده است.



شکل ۴-۱: یک نمونه از تصویر دست‌خط استفاده شده در این تحقیق

۴-۲نرمال‌سازی تصویر و استخراج ویژگی‌ها

تصویر متن دستنویس را باید نسبت به تاثیر عواملی مثل فاصله خطوط، فاصله کلمات و غیره نرمال‌سازی کرد تا یک بلوک یکنواخت از متن دستنویس مورد نظر حاصل شود [۷۱]. مراحل نرمال‌سازی تصویر متن دستنویس بصورت مفصل در گزارش پایان‌نامه آقای دهقان آورده شده و به دلیل جلوگیری از زیادشدن حجم این گزارش مراحل انجام عملیات نرمال سازی دستخط را در این گزارش نیاورده و خواننده در صورت علاقه می-تواند به این مرجع مراجعه نماید. بعد از نرمال‌سازی تصویر می‌بایست استخراج ویژگی از تصویر نرمال شده انجام گردد با توجه به اینکه ما در این تحقیق از ویژگی‌های برون خط از دست‌خط می‌خواهیم استفاده کنیم و به نگاه بافت به دستخط داریم از فیلترهای گابور که در تحقیق شهابی‌نژاد [۲۸] به کار گرفته شده و همچنین از فیلتر-های سوبل و روبرتز که در تحقیق دهقان [۲] استفاده شده‌اند استفاده نموده‌ایم. مراحل استخراج ویژگی در این تحقیق مشابه با مراحل استفاده شده در تحقیقهای ذکر شده می‌باشد و برای جلوگیری از زیاد شدن حجم این گزارش در این گزارش مراحل استخراج ویژگی بیان نشده است و خواننده علاقه‌مند میتواند به این مراجع مراجعه نماید.

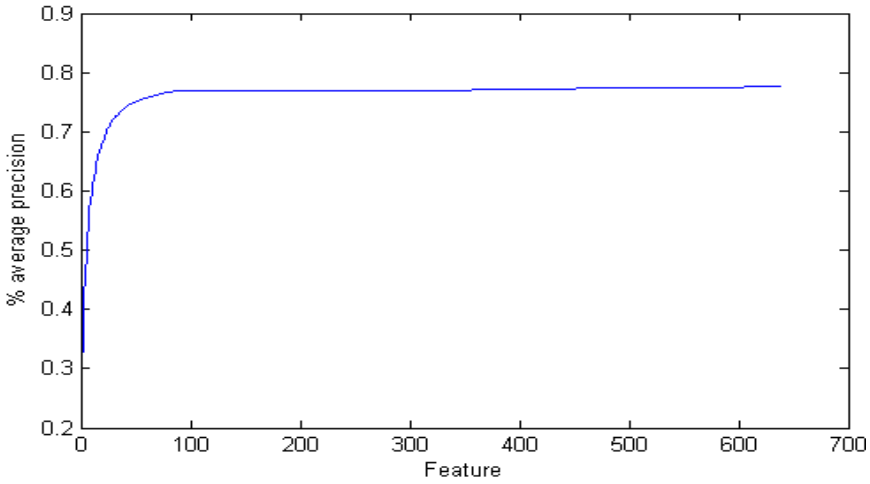
۴-۳ کاهش ویژگی

در این گزارش با استفاده از تکنیک PCA ویژگی‌ها را کاهش داده‌ایم و سپس از شبکه عصبی برای تشخیص هویت نویسنده استفاده کرده‌ایم مراحل کار به صورت زیر می‌باشد:

ابتدا الگوریتم PCA را روی مجموعه ویژگی‌های استخراج شده از هر کدام از روش های فوق اجرا کرده و آنها را از موثرترین ویژگی تا کم‌اهمیت ترین ویژگی مرتب میکنیم. سپس با استفاده از یک کلاس‌بند ساده یا

ارزیاب با اضافه نمودن هر ویژگی از بهترین ویژگی تا آخرین ویژگی در هر مجموعه، میزان دقت را ارزیابی می‌کنیم. در اینجا از ارزیاب K نزدیکترین همسایه (KNN) با اضافه نمودن هر ویژگی ارزیاب را با مجموعه داده آموزش، آموزش داده و سپس با مجموعه داده آزمایش دقت را ارزیابی می‌کنیم. مقادیر دقت به دست آمده متناظر با هر تعداد ویژگی که ارزیابی برای آن صورت گرفته را ذخیره می‌کنیم تا اینکه تمام ویژگی‌ها اضافه شوند. در پایان نموداری برای تعداد ویژگی‌ها و دقت متناظر با آن رسم می‌کنیم، نمونه ای از نمودار ایجاد شده برای ویژگی‌های استخراج شده از روش سوبل در شکل ۳ آورده شده است. از نمودار به دست آمده مناسب‌ترین تعداد از ویژگی‌ها را برای استفاده در مرحله کلاس‌بندی انتخاب کرده و بقیه ویژگی‌ها را حذف می‌کنیم. اینکار با یافتن نقطه اشباع در نمودار دقت بر حسب تعداد ویژگیهای اضافه شده انجام می شود.

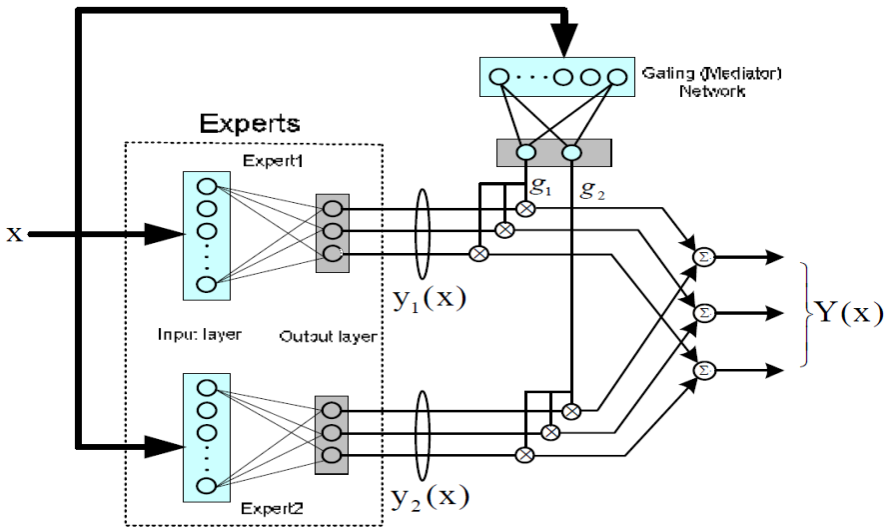
کاهش تعداد ویژگی ها علاوه بر کاهش چشم‌گیر در زمان مرحله آموزش شبکه های عصبی MLP، بهبود دقت را نیز به همراه داشته است که با توجه به جدول ۱ این موضوع قابل درک می‌باشد. تعداد ویژگی‌ها برای استخراج ویژگی برای هر کدام از روشهای فیلتر گابور، فیلتر سوبل و فیلتر روبرت به ترتیب برابر ۱۲۰، ۶۴۰ و۶۴۰ بوده که بعد از کاهش ویژگی برای هر کدام به ترتیب به ۶۰، ۱۲۰و ۱۲۰ کاهش یافته است.



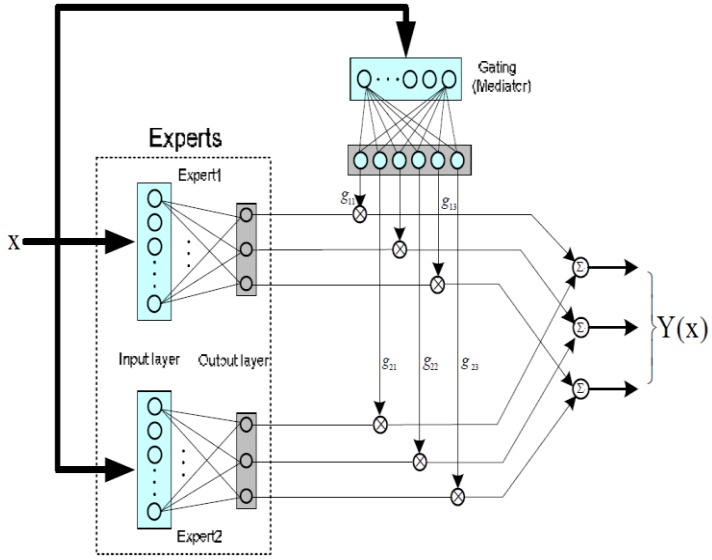
شکل ۴-۲: دقت تشخیص هویت متناظر با تعداد ویژگیها برای روش استخراج ویژگی‌ها توسط فیلتر سوبل و با

۴-۴ کلاس‌بندی و اختلاط خبره ها

کلاسه‌بندی به عنوان آخرین مرحله شناسایی نویسنده است. در این تحقیق از اختلاط خبره ها(ME) جهت طبقه بندی نویسنده‌ها و تعیین هویت آنها استفاده گردیده است [۴]. برای ایجاد شبکه مرکب از دو ساختار استفاده گردیده است در ساختار اول که در شکل ۴-۳ نمایش داده شده است، به نظر هر طبقه بند یک وزن یا ضریب اعمال می‌کنیم که در سه آزمایش نخست از این ساختار استفاده شده است که در آزمایش اول این وزنها از طریق آزمایش به دست آمده‌اند و در آزمایش دوم با کمینه کردن خطا در مرحله طبقه‌بندی وزنهای بهینه را با الگوریتم PSO به دست آورده‌ایم. در ساختار دوم که در شکل ۴-۴ نشان داده شده است به هر کدام از نرون-های خروجی شبکه‌های عصبی که خبره‌های(Experts) ما در این تحقیق هستند وزن داده ایم که در آزمایش چهارم از این ساختار استفاده کرده‌ایم و برای به دست آوردن وزن‌های بهینه از الگوریتم PSO استفاده کرده‌ایم. برای الگوریتم PSO پارامترهای c1 و c2 به ترتیب برابر در نظر گرفته شده‌اند تعداد جمعیت ۵۰ ذره و تعداد تکراربرای ساختار اول ۱۰۰تکرار و برای ساختار دوم ۲۰۰ تکرار در نظر گرفته شده است پارامترهای الگوریتم وزنهای مورد نیاز شبکه می‌باشند که به صورت بهینه با مینیمم کردن خطا برای داده‌های آموزش به دست آمده‌اند، که برای ساختار اول ۳ و برای ساختار دوم ۱۲۰ پارامتر می‌باشد. خبره های ما شبکه‌های عصبی پیش-رونده (feed forward) با یک لایه میانی و تعداد نرونهای لایه میانی (مخفی) ۲۵ نرون می‌باشد و از روش گرادیان نزولی برای مینیمم کردن خطا استفاده شده و از تابع سیگموئید در نرون های میانی و خروجی استفاده کرده ایم. پارامتر K در تابع سیگموئید، 0.3 در نظر گرفته شده و نرخ آموزش و مومنتم به ترتیب 0.5 و 0.8 در نظر گرفته شده اند.



شکل ۴-۳: ساختار یک شبکه مرکب. ساختار اول[۳]



شکل ۴-۴: ساختار یک شبکه مرکب. ساختار دوم[۳]

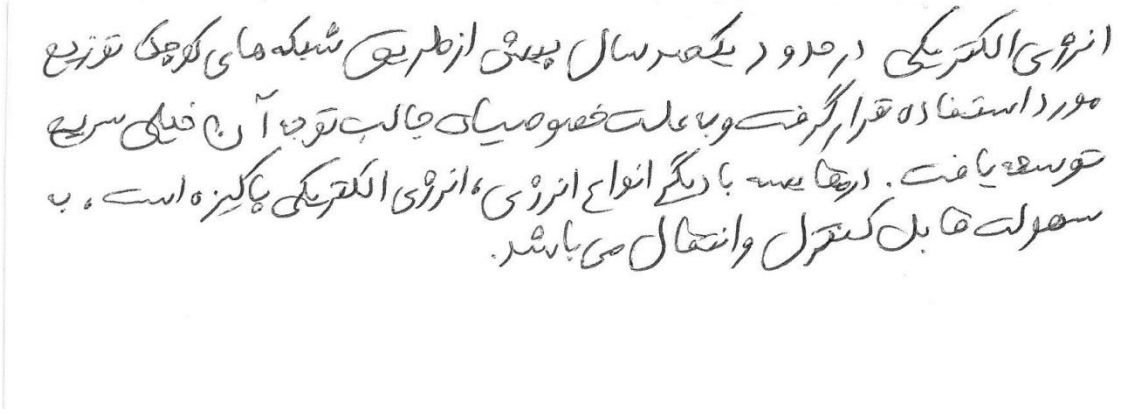
۴-۵ آزمایش اول

در آزمایش اول هر کدام از طبقه‌بندها را به صورت جداگانه آموزش دادیم و در نهایت نظر آنها را با هم ترکیب کردیم در این آزمایش نظر تمام طبقه‌بندها را با هم برابر گرفتیم برای محاسبه نتایج ده مرتبه شبکه‌های عصبی را روی مجموعه داده‌ها آموزش داده و نتیجه شبکه مرکب را بر روی داده‌های آموزش به دست آورده و در

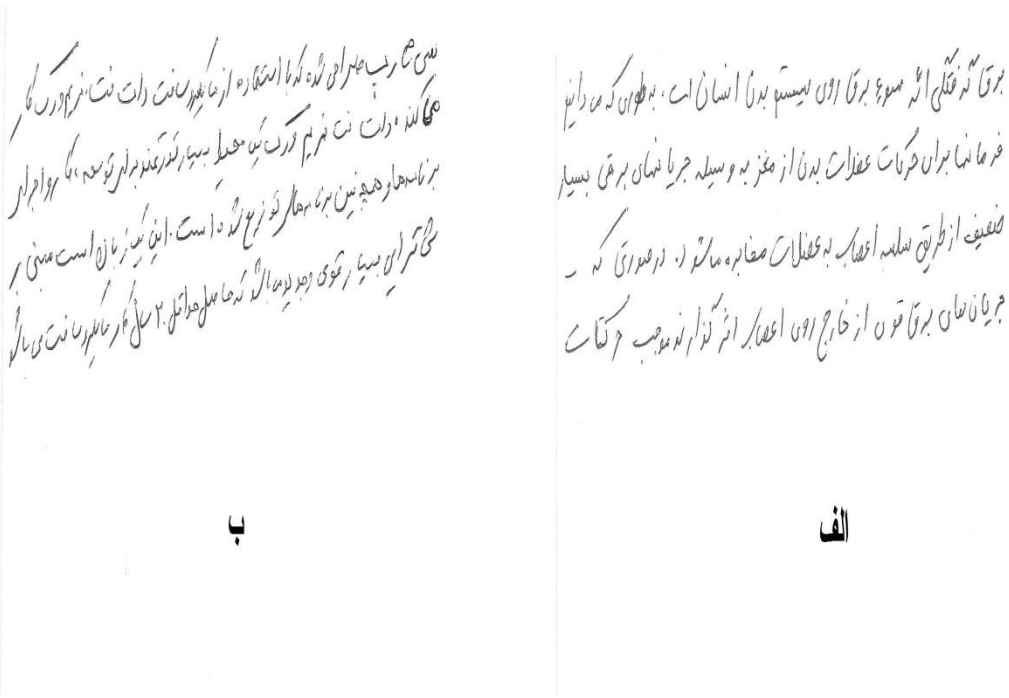
نهایت میانگین نتایج به دست آمده را به عنوان میزان دقت به دست آمده ثبت کرده‌ایم نحوه عملکرد شبکه مرکب به این شکل می‌باشد که در مرحله آموزش هر کدام از شبکه‌های عصبی به صورت جداگانه بر روی مجموعه ویژگی‌هایی که به عنوان ورودی در اختیار شبکه قرار گرفته است آموزش می‌بینند و در مرحله آزمایش خروجی نرونهای لایه آخر شبکه‌ها را در ضریب یک سوم ضرب میکنیم و خروجی نرون‌های متناظر را با هم جمع می‌کنیم. در نهایت چهل نرون در شبکه مرکب به عنوان نرونهای لایه خروجی وجود خواهند داشت که در مقدار خروجی آن‌ها هر کدام از شبکه‌ها به اندازه یکسان نقش دارند. ساختار این شبکه همانند شکل ۴-۳ می‌باشد که در این ساختار مقادیر g_i ها برابر یک سوم در نظر گرفته شده‌اند نتایج این آزمایش در جدول ۴-۱ آورده است با توجه به جدول ۴-۱ می‌توان پی برد که ترکب شبکه ها باعث بهبود در میزان دقت در تعیین هویت بر اساس دستخط افراد گردیده است. برای درک بیشتر یک نمونه از دست‌خط‌های طبقه‌بندی شده به صورت درست را در شکل ۴-۴ و یک نمونه که اشتباه با شبکه مرکب طبقه‌بندی شده و به نویسنده اشتباه نسبت داده شده را در شکل ۴-۵ الف آورده ایم و برای مقایسه نمونه ای از دستخط نویسنده‌ای که این نمونه اشتباها توسط شبکه مرکب به وی نسبت داده شده را در شکل ۴-۶ آورده‌ایم

جدول ۴-۱:نتایج به دست آمده از آزمایش (۱)

روش استخراج ویژگی	دقت مرحله آموزش قبل از کاهش ویژگی‌ها با PCA	دقت مرحله آزمایش قبل از کاهش ویژگی‌ها با PCA	زمآن اجرای الگوریتم آموزش قبل از کاهش ویژگی‌ها با PCA	دقت مرحله آموزش بعد از کاهش ویژگی‌ها با PCA	دقت مرحله آزمایش بعد از کاهش ویژگی‌ها با PCA	زمآن اجرای الگوریتم آموزش بعد از کاهش ویژگی‌ها با PCA
فیلتر سوبل	۹۹,۹۶ درصد	۹۴,۷۵ درصد	۱۳۰,۶۳ ثانیه	۹۹,۹۶ درصد	۹۵,۱۲۵ درصد	۷۱,۷۳ ثانیه
فیلتر روبرتز	۹۹,۸ درصد	۹۳,۶۲۵ درصد	۱۷۹,۷۲ ثانیه	۹۹,۹۶ درصد	۹۴,۲۵ درصد	۱۱۱,۲۱ ثانیه
فیلتر گابور	۹۸,۹ درصد	۸۹,۷۵ درصد	۲۸۳۵,۶۶ ثانیه	۹۹,۵۳ درصد	۹۱,۸۷۵ درصد	۲۲۶۷,۶ ثانیه
شبکه مرکب	*	۹۷,۲۵ درصد	۳۱۴۶ ثانیه	*	۹۷,۴۲ درصد	۲۴۵۰,۵۴ ثانیه



شکل ۴-۵: یک نمونه از دست‌خط‌هایی که توسط شبکه مرکب آزمایش ۱ درست تشخیص داده شده است.این دستخط مربوط به نویسنده دوازده‌ام بوده که به درستی تشخیص داده شده است.



شکل ۴-۶: الف) یک نمونه از دست‌خط‌های اشتباه تشخیص داده شده توسط شبکه مرکب آزمایش اول. کلاس صحیح ۱۳بوده که به اشتباه کلاس ۳ تشخیص داده شده است. ب)یک نمونه از دستخط‌های نویسنده سوم

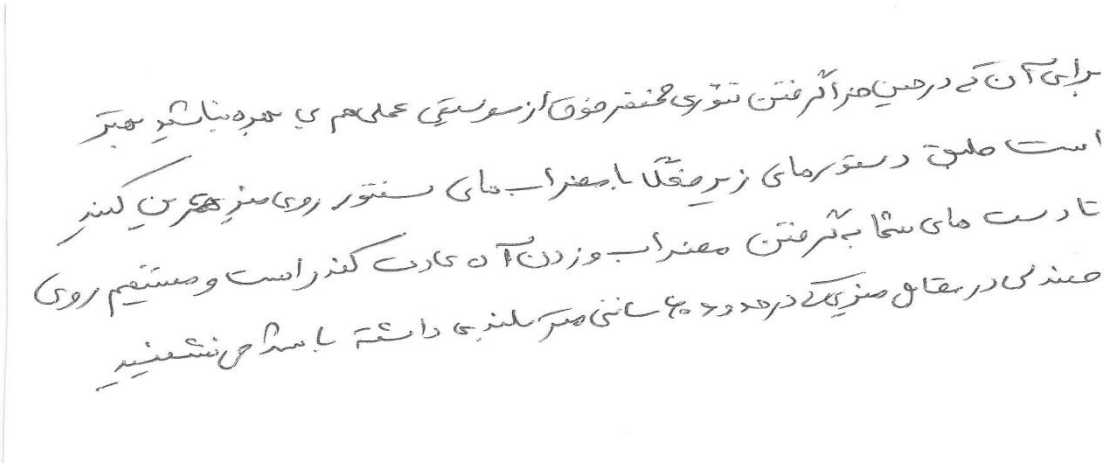
۴-۶ آزمایش دوم

در آزمایش دوم ابتداهر کدام از مجموعه ویژگی‌های استخراج شده را توسط شبکه‌های عصبی به صورت جداگانه کلاس‌بندی کردیم و در نهایت از میزان دقت هر کدام جهت میزان اهمیت(شایستگی) هر کدام استفاده نموده و نظر هر کدام از شبکه‌های عصبی را با هم ترکیب کرده ایم که نتایج آن در جدول ۴-۲ آورده شده است لازم به ذکر است که در این آزمایش برای به دست آوردن میزان شایستگی هر طبقه‌بند(میزان دقت هرکدام) ده مرتبه هر کدام از شبکه‌ها را آموزش دادیم و میزان دقت را بر روی داده‌های آزمایش به دست آوردیم و در نهایت معدل آنها را به عنوان شایستگی طبقه‌بند مورد نظر در نظر گرفتیم. بقیه مراحل انجام این آزمایش شبیه به مراحل انجام شده در آزمایش قبل می‌باشد و ساختار شبکه نیز همانند شکل ۴-۳ می‌باشد با این تفاوت که در این

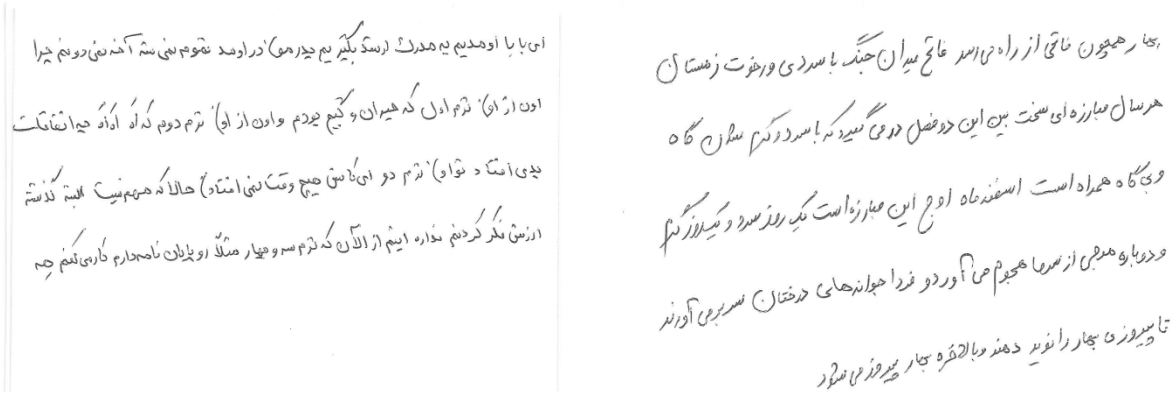
آزمایش به میزان برابر به نظر هر کدام از شبکه ها اهمیت نداده و میزان دقتی که در آزمایشی که برای به دست آوردن شایستگی هر شبکه انجام گرفته است تعیین کننده میزان شایستگی هر کدام می‌باشد. و مقادیر gi ها در ساختار شبکه نمایش داده شده در شکل ۴-۳ را به صورت تجربی و از طریق آزمایش به دست آورده‌ایم. نتایج به دست آمده از این آزمایش در جدول ۴-۲ آورده شده است. نتایج حاصل از این آزمایش نیز بهبود دقت را در تعیین هویت نویسنده نسبت به کلاسه‌بند های تکی نشان می‌دهد. برای درک بیشتر یک نمونه از دست‌خط‌های طبقه‌بندی شده به صورت درست را در شکل ۴-۴ و یک نمونه که اشتباه با شبکه مرکب طبقه‌بندی شده و به نویسنده اشتباه نسبت داده شده را در شکل ۴-۵ الف آورده ایم و برای مقایسه نمونه ای از دستخط نویسنده‌ای که این نمونه اشتباها توسط شبکه مرکب به وی نسبت داده شده را در شکل ۴-۶ آورده‌ایم

جدول ۴-۲: نتایج به دست آمده از آزمایش (۲)

روش استخراج ویژگی	دقت مرحله آموزش قبل از کاهش ویژگی‌ها با PCA	دقت مرحله آزمایش قبل از کاهش ویژگی‌ها با PCA	زمآن اجرای الگوریتم آموزش قبل از کاهش ویژگی‌ها با PCA	دقت مرحله آموزش بعد از کاهش ویژگی‌ها با PCA	دقت مرحله آزمایش بعد از کاهش ویژگی‌ها با PCA	زمآن اجرای الگوریتم آموزش بعد از کاهش ویژگی‌ها با PCA
فیلتر سوبل	۹۹,۹۶ درصد	۹۴,۷۵ درصد	۱۳۰,۶۳ ثانیه	۹۹,۹۶ درصد	۹۵,۱۲۵ درصد	۷۱,۷۳ ثانیه
فیلتر روبرتز	۹۹,۸ درصد	۹۳,۶۲۵ درصد	۱۷۹,۷۲ ثانیه	۹۹,۹۶ درصد	۹۴,۲۵ درصد	۱۱۱,۲۱ ثانیه
فیلتر گابور	۹۸,۹ درصد	۸۹,۷۵ درصد	۲۸۳۵,۶۶ ثانیه	۹۹,۵۳ درصد	۹۱,۸۷۵ درصد	۲۲۶۷,۶ ثانیه
شبکه مرکب	*	۹۷,۵ درصد	۳۱۴۶ ثانیه	*	۹۷,۷۵ درصد	۲۴۵۰,۵۴ ثانیه



شکل ۴-۶: یک نمونه از دست‌خط‌های صحیح تشخیص داده شده توسط شبکه مرکب آزمایش ۲



الف **ب**

شکل ۴-۷: الف) یک نمونه از دست‌خط‌های اشتباه تشخیص داده شده توسط شبکه مرکب کلاس صحیح ۲ بوده که به اشتباه کلاس ۵ تشخیص داده شده است. ب)یک نمونه از دستخط‌های نویسنده پنجم

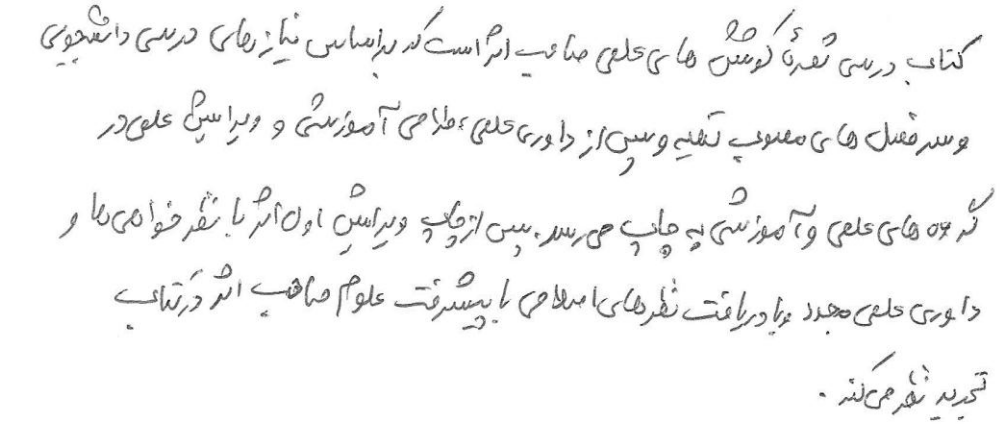
۴-۷ آزمایش سوم

آزمایش سوم مشابه آزمایش دوم از ساختار شبکه مرکب نشان داده شده در شکل ۴-۳ استفاده شده با این تفاوت که در این آزمایش برای تعیین وزن‌ها از الگوریتم PSO استفاده کرده‌ایم و وزنهای بهینه را به گونه‌ای که

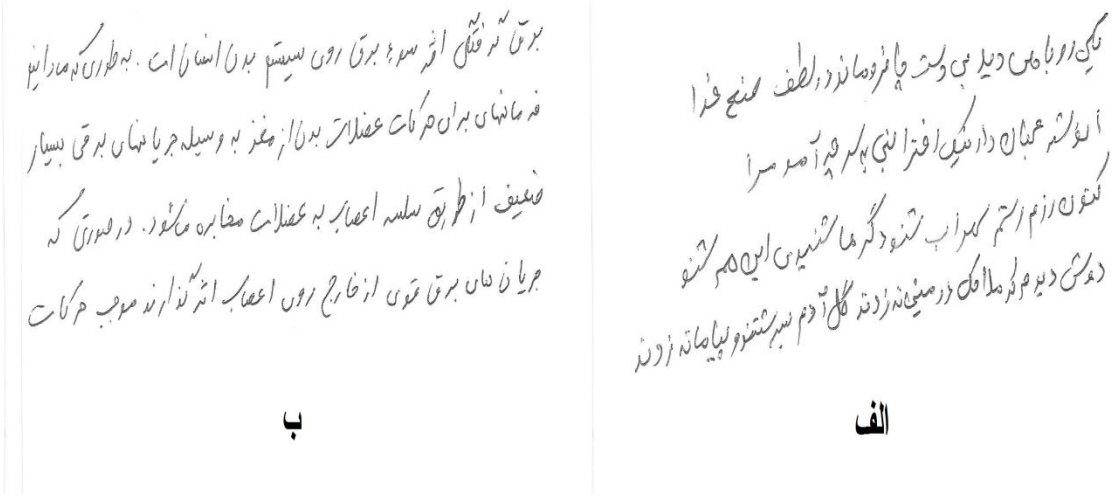
خطای مرحله آموزش مینیمم شود از این روش به دست آورده‌ایم در این آزمایش برای هر کدام از خبره‌ها همانند آزمایش قبل یک وزن به دست آورده‌ایم که در مقدار نرونهای خروجی شبکه ها ضرب می‌گردد و در نهایت خروجی هر سه شبکه با هم ترکیب می‌گردد. نتایج این آزمایش در جدول ۴-۳ آورده شده است. با توجه به جدول میتوان مشاهده کرد که ترکیب طبقه‌بندها باعث بهبود دقت نسبت به بهترین طبقه‌بند گردیده است. برای درک بیشتر یک نمونه از دست‌خط‌های طبقه‌بندی شده به صورت درست را در شکل ۴-۴ و یک نمونه که اشتباه با شبکه مرکب طبقه‌بندی شده و به نویسنده اشتباه نسبت داده شده را در شکل ۴-۵ الف آورده ایم و برای مقایسه نمونه ای از دستخط نویسنده‌ای که این نمونه اشتباها توسط شبکه مرکب به وی نسبت داده شده را در شکل ۴-۶ آورده‌ایم.

جدول ۴-۳: نتایج به دست آمده از آزمایش (۳)

روش استخراج ویژگی	دقت مرحله آموزش قبل از کاهش ویژگی‌ها با PCA	دقت مرحله آزمایش قبل از کاهش ویژگی‌ها با PCA	زمآن اجرای الگوریتم آموزش قبل از کاهش ویژگی‌ها با PCA	دقت مرحله آموزش بعد از کاهش ویژگی‌ها با PCA	دقت مرحله آزمایش بعد از کاهش ویژگی‌ها با PCA	زمآن اجرای الگوریتم آموزش بعد از کاهش ویژگی‌ها با PCA
فیلتر سوبل	۹۹,۹۶ درصد	۹۴,۷۵ درصد	۱۳۰,۶۳ ثانیه	۹۹,۹۶ درصد	۹۵,۱۲۵درصد	۷۱,۷۳ثانیه
فیلتر روبرتز	۹۹,۸درصد	۹۳,۶۲۵در صد	۱۷۹,۷۲ ثانیه	۹۹,۹۶درصد	۹۴,۲۵ درصد	۱۱۱,۲۱ ثانیه
فیلتر گابور	۹۸,۹ درصد	۸۹,۷۵ درصد	۲۸۳۵,۶۶ثانیه	۹۹,۵۳درصد	۹۱,۸۷۵ درصد	۲۲۶۷,۶ ثانیه
شبکه مرکب	*	۹۷,۳۲درصد	۳۲۴۵,۱۴ ثانیه	*	۹۷,۵۹درصد	۲۵۶۲,۰۴ثانیه



شکل ۴-۸: یک نمونه از دست‌خط‌های صحیح تشخیص داده شده توسط شبکه مرکب آزمایش۳



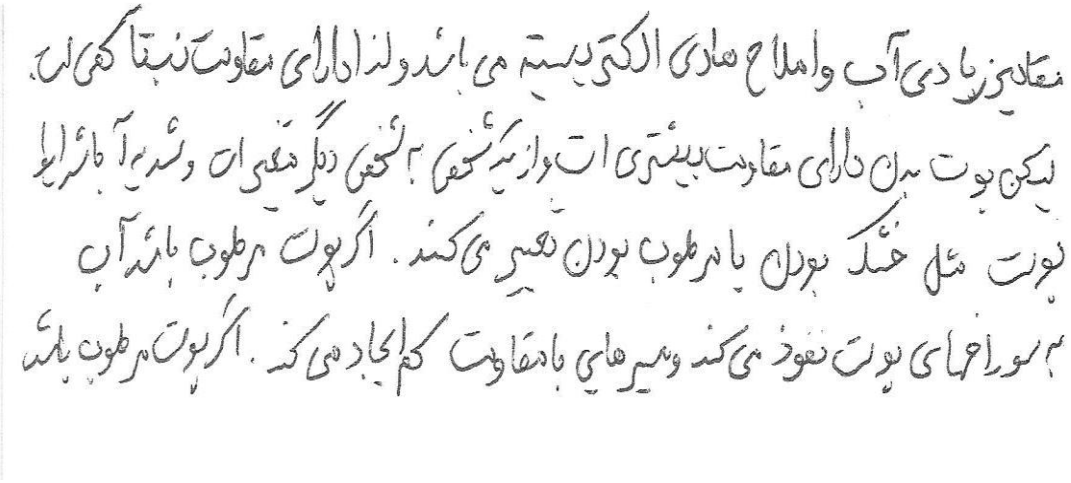
شکل ۴-۹: الف) یک نمونه از دست‌خط‌های اشتباه تشخیص داده شده توسط شبکه مرکب آزمایش سوم کلاس صحیح ۱۸ بوده که به اشتباه کلاس ۱۳ تشخیص داده شده است. ب)یک نمونه از دستخط‌های نویسنده سیزدهم

۴-۸آزمایش چهارم

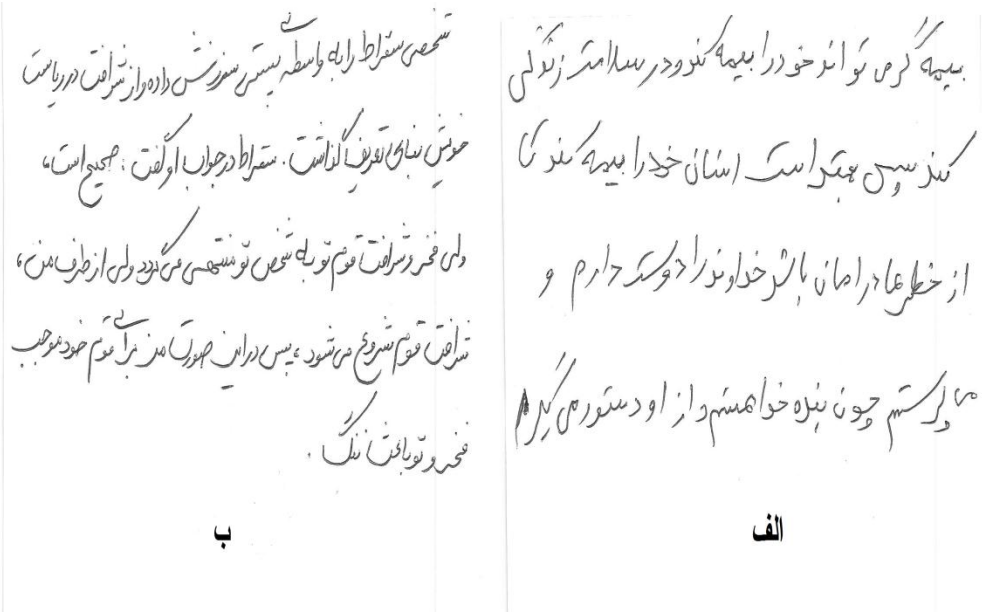
در این آزمایش از ساختار شکل ۴-۴ استفاده کرده‌ایم و برای خروجی هر نرون لایه خروجی شبکه های پایه یک وزن در نظر گرفته‌ایم که در این آزمایش نیز شبیه به آزمایش قبل این وزنها را با استفاده از الگوریتم PSO به دست آورده‌ایم در این آزمایش به تعداد نرونهای لایه خروجی سه شبکه پایه یعنی ۱۲۰ تا وزن نیاز داریم که آنها را توسط الگوریتم PSO به نحوی می‌یابیم که خطای شبکه مرکب در مرحله آموزش مینیمم گردد. نتایج به دست آمده از این آزمایش در جدول ۴-۴ آورده شده است با توجه به این جدول مشاهده میگردد که ترکیب طبقه‌بندهای پایه با این روش نیز باعث بهبود در دقت طبقه‌بندی نویسنده ها یا به عبارت دیگر افزایش دقت در تعیین هویت نویسنده‌ها در مقایسه با طبقه‌بند تکی می‌باشد.

جدول ۴-۴: نتایج به دست آمده از آزمایش (۴)

روش استخراج ویژگی	دقت مرحله آموزش قبل از کاهش ویژگی‌ها با PCA	دقت مرحله آزمایش قبل از کاهش ویژگی‌ها با PCA	زمآن اجرای الگوریتم آموزش قبل از کاهش ویژگی‌ها با PCA	دقت مرحله آموزش بعد از کاهش ویژگی‌ها با PCA	دقت مرحله آزمایش بعد از کاهش ویژگی‌ها با PCA	زمآن اجرای الگوریتم آموزش بعد از کاهش ویژگی‌ها با PCA
فیلتر سوبل	۹۹,۹۶ درصد	۹۴,۷۵ درصد	۱۳۰,۶۳ ثانیه	۹۹,۹۶ درصد	۹۵,۱۲۵ درصد	۷۱,۷۳ ثانیه
فیلتر روبرتز	۹۹,۸ درصد	۹۳,۶۲۵ درصد	۱۷۹,۷۲ ثانیه	۹۹,۹۶ درصد	۹۴,۲۵ درصد	۱۱۱,۲۱ ثانیه
فیلتر گابور	۹۸,۹ درصد	۸۹,۷۵ درصد	۲۸۳۵,۶۶ ثانیه	۹۹,۵۳ درصد	۹۱,۸۷۵ درصد	۲۲۶۷,۶ ثانیه
شبکه مرکب	*	۹۷,۵ درصد	۳۲۴۸,۷۶ ثانیه	*	۹۸,۱۷ درصد	۲۵۹۴,۵۴ ثانیه



شکل ۴-۱۰: یک نمونه از دست‌خط‌های صحیح تشخیص داده شده توسط شبکه مرکب آزمایش۴



شکل ۴-۱۱: الف) یک نمونه از دست‌خط‌های اشتباه تشخیص داده شده توسط شبکه مرکب آزمایش چهارم کلاس صحیح ۳۳ بوده که به اشتباه کلاس ۲۸ تشخیص داده شده است. ب)یک نمونه از دستخط‌های نویسنده بیست و هشتم

فصل پنجم:

نتیجه‌گیری

۵-۱ نتیجه‌گیری

در این تحقیق با ترکیب چند شبکه عصبی و آموزش آنها بر روی یک مجموعه داده که به روشهای مختلفی جهت گوناگون بودن در خطا استخراج ویژگی گردید توانسته‌ایم در دقت تعیین هویت افزایش ایجاد کنیم و یک روش هوشمند برای افزایش دقت تعیین هویت نویسنده بر اساس دستخط که از ویژگی‌های بیومتریک انسان می‌باشد و اثبات گردیده که برای هر فرد منحصر به فرد می‌باشد را به کار بگیریم و با توجه به نتایج به دست آمده نشان دهیم که ترکیب طبقه بندها و استفاده آنها جهت طبقه‌بندی نویسنده‌ها و تعیین هویت آنها می‌تواند دقت تعیین هویت افراد را افزایش دهد. در این تحقیق همچنین با استفاده از انتخاب مناسب ویژگی‌ها و حذف ویژگی‌های اظافی علاوه بر اینکه در زمان مرحله آموزش توانستیم کاهش ایجاد کنیم در بعضی مواقع بهبود در دقت را نیز به همراه داشته است. به دلیل نبود پایگاه داده استاندارد برای این منظور مقایسه دقیقی نمی‌توان با کارهای صورت گرفته در قبل انجام داد. به همین دلیل بیشتر سعی می‌گردد مقایسه‌ای بین نتایج این تحقیق با تحقیق صورت گرفته توسط آقای دهقان که بر روی همین پایگاه داده انجام شده است را به طور خاص انجام دهیم و همچنین نتایج برخی از کارهای صورت گرفته در گذشته را برای مقایسه شهودی و کیفی آورده‌ایم. در جدول ۵-۱ نتایج آزمایشهای صورت گرفته در این تحقیق با نتایج به دست آمده توسط آقای دهقان مقایسه گردیده است. و در جدول ۵-۲ نتایج به دست آمده توسط محققین گذشته که در زمینه تعیین هویت نویسنده تحقیق انجام داده اند را برای مشاهده و مقایسه کیفی خواننده آورده‌ایم. با توجه به جدول ۵-۱ مشاهده می‌گردد که تمامی آزمایش‌های انجام شده در این تحقیق نتیجه بهتری نسبت به نتیجه حاصل از تحقیق دهقان حاصل شده است. و در سه آزمایش نخست که از ساختار اول برای ترکیب طبقه‌بندها استفاده شده برابر در نظر گرفتن وزنها هرچند نسبت به بهترین طبقه‌بند تکی نتیجه بهتری داشته اما در بین آزمایشهای این تحقیق پایین‌ترین دقت را داشته و بعد از آن از نظر دقت شبکه مرکبی قرار گرفته است که توسط سه وزن که به هر شبکه توسط

الگوریتم PSO داده شده است، شبکه‌های عصبی با هم ترکیب گردیده‌اند و شبکه مرکبی که به هر شبکه بر اساس شایستگی آن شبکه وزن داده‌ایم بهترین نتیجه را در سه آزمایش اول که درآن از ساختار اول جهت ترکیب شبکه های عصبی با هم استفاده شده است،را داشته است. و در نهایت آزمایش چهارم که در آن از ساختار دوم استفاده گردیده که به هر نرون یک وزن تخصیص یافته است و وزنها توسط PSO به صورت بهینه پیدا شده‌اند بهترین نتیجه را به خود اختصاص داده است.

جدول ۵-۱: مقایسه نتایج آزمایشهای انجام شده در این تحقیق با نتایج به دست آمده توسط

روش به کار رفته	آزمایش اول	آزمایش دوم	آزمایش سوم	آزمایش چهارم	روش به کار رفته توسط [۲]
دقت	۹۷,۴۲ درصد	۹۷,۷۵ درصد	۹۷,۵۹ درصد	۹۸,۱۷ درصد	۹۶,۲۵ درصد
روش کلاس-بندی	ME	ME	ME	ME	KNN

جدول۵-۲: مقایسه روش پیشنهادی با کارهای گذشته

روش به کار رفته	[۷۲]	[۷۳]	[۷۴]	[۱۰]	[۱۵]	[۲]	روش پیشنهادی
کلاسه‌بند به کار رفته	WED	WED	LCS	WED,KNN, LDC,SVM	NEURAL NETWORK	KNN	ME
دقت	۸۲ درصد	۹۲ درصد	۹۵ درصد	۹۰ درصد	۹۴ درصد	۹۶/۲۵ درصد	۹۸/۱۷ درصد
تعداد افراد	۷۰	۲۵	۱۰۰	۲۰	۵۰	۵۰	۴۰

مراجع

[۱] S. N. Srihari, H. Arora, S. H. Cha *et al.*, "Individuality of handwriting," *Journal of Forensic*, vol. 47.

[۲] مسعود دهقان, "تعیین هویت شخص از روی دستنوشته فارسی," پایان‌نامه کارشناسی ارشد، دانشگاه صنعتی شاهرود, ایران,۱۱۴ص، ۱۳۹۰.

[۳] سید حسن نبوی کریزی, "ترکیب طبقه بند ها با تاکید بر گوناگونی آنها," پایان‌نامه دکتری، دانشگاه تربیت مدرس, ۱۷۲ص, ۱۳۸۵.

[۴] L. I. Kuncheva, *Combining Pattern Classifiers Methods and Algorithms*, Canada: A JOHN WILEY & SONS, INC., PUBLICATION, 2004.

[۵] M. Dash, and R. Liu, "Feature selection for classification",*Intelligent data analysis*, vol. 1, pp. 131-156, 1997.

[۶] S. Srihari, M. Beal, K. Bandi *et al.*, "A Statistical Model for Writer Verification." pp. 1105-1109.

[۷] B. Zhang, S. Srihari, and S. Lee, "Individuality of Handwritten Characters." pp. 1086-1090.

[۸] S. Srihari, C. Tomai, B. Zhang *et al.*, "Individuality of Numerals." pp. . 1096-1100.

[۹] C. Tomai, B. Zhang, and S. Srihari, "Discriminatory Power of Handwritten Words for Writer Recognition." pp. 638-641.

[۱۰] H. Said, G. Peake, T. Tan *et al.*, "Writer Identification from Non-Uniformly Skewed Handwriting Images," in Ninth British Machine Vision Conf, 1998, pp. 478-487.

[۱۱] H. Said, T. Tan, and K. Baker, "Personal Identification Based on Handwriting," *Pattern Recognition*, vol. 33, pp. 149-160, 2000.

[۱۲] G. S. Peake, and T. N. Tan, "Script and language identification from document images." pp. 610-619.

[۱۳] T. N. Tan, "Rotation invariant texture features and their use in automatic script identification," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol. 20, pp. 751-756, 1998.

[۱۴] Y. Zhu, T. N. Tan, and Y. Wang, "Font recognition based on global texture analysis," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol. 23, no. 1, pp. 1192-1200, 2001.

[۱۵] E. Zois, and V. Anastassopoulos, "Morphological Waveform Coding for Writer Identification," *Pattern Recognition,* vol. 33, no. 3, pp. . 385-398, 2000.

[۱۶] A. Bensefia, T. Paquet, and L. Heutte, "A Writer Identification and Verification System," *Pattern Recognition Letters*, vol. 26 ,no. 10, pp. 2080-2092, 2005.

[۱۷] A. Bensefia, T. Paquet, and L. Heutte, “Handwritten Document Analysis for Automatic Writer Recognition,” *Electronic Letters on Computer Vision and Image Analysis*, vol. 5, no. 2, pp. 72-86, 2005.
[۱۸] U.-V. Marti, R. Messerli, and H. Bunke, "Writer IdentificationUsing Text Line Based Features." pp. 101-105.
[۱۹] C. Hertel, and H. Bunke, "A Set of Novel Features for Writer Identification." pp. 679-687.
[۲۰] A. Schlapbach, V. Kilchherr, and H. Bunke, "Improving Writer Identification by Means of Feature Selection and Extraction." pp. 131-135.
[۲۱] A. Schlapbach, and H. Bunke, "Using HMM-Based Recognizers for Writer Identification and Verification." pp. 167-172.
[۲۲] U.-V. Marti, and H. Bunke, “Using a Statistical Language Model to Improve the Performance of an HMM-Based Cursive Handwriting Recognition System,” *Pattern Recognition and Artificial Intelligence*, vol. 15, pp. 65-90, 2001.
[۲۳] M. Bulacu, L. Schomaker, and L. Vuurpijl, "Writer Identification Using Edge-Based Directional Features." pp. 937-941.
[۲۴] M. Bulacu, and L. Schomaker, "Writer Style from Oriented Edge Fragments." pp. 460-469.
[۲۵] L. Schomaker, M. Bulacu, and K. Franke, "Automatic Writer Identification Using Fragmented Connected-Component Contours." pp. 18.۱۹۰-۵
[۲۶] M. Bulacu, and L. Schomaker, “Text-Independent Writer Identification and Verification Using Textural and Allographic Features,” *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 29, no. 4, 2007
[۲۷] A. Al-Dmour, and R. A. Zitar, “Arabic writer identification based on hybrid spectral-statistical measures,” *Journal of Experimental & Theoretical Artificial Intelligence*, 2007.
[۲۸] محمد رحمتی و فاطمه شهابی‌نژاد، "مقایسه ویژگیهای مبتنی بر فیلترهای گابور و ارایه روشی جدید برای تعیین هویت نویسنده بر اساس دستنوشته فارسی."چهارمین کنفرانس ماشین‌بینایی و پردازش تصویر،۱۳۸۵.
[۲۹] S. Sadeghi, Ram, and M. E. Moghadam, “A Persian Writer Identification method based on Gradient Features and Neural Networks,” in IEEE conf. on Image and signal processing, 2009, pp. 1-4.
[۳۰] R. O. Duda, P. E.hart, and D. G. stork, *pattern classification*, new york: Wiley-Interscience Publication, 2001.
[۳۱] B. V. Dasarathy, and B. V. Sheela, “A composite classifier system design: concepts and methodology,” *Proceedings of IEEE*, vol. 67, pp. 708–713, 1978.
[۳۲] R .H. Erenstein, and L. A. Rastrigin, “Method of collective recognition,” *Energoizdat,Moscow,*, 1981.
[۳۳] L. I. Kuncheva, *Pattern Recognition Letters,* vol. 14, pp. 619-623, 1993.

[۳۴] L. Lam, "Classifier combinations: implementations and theoretical issues ", *Multiple classifier systems*, pp. 77-86, 2000.
[۳۵] C. Kaynak, and E. Alpaydin, "Cascading classifiers," *KYBERNETIKA-PRAHA*, vol. 34, pp. 369-374, 1998.
[۳۶] K. Bowyer, W. P. Kegelmeyer Jr, and K. Woods, "Combination of multiple classifiers using local accuracy estimates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 405-410, 1997.
[۳۷] F. Roli, and G. Giacinto, "Design of effective neural network ensembles for image classification purposes," *Image and Vision Computing*, vol ,۱۹ .pp. 699-707, 2001.
[۳۸] K. Woods, W. P. Kegelmeyer, and K. Bowyer, "Combination of multiple classifiers using local accuracy estimates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 405–410, 1997.
[۳۹] L. I. Kuncheva" ,Switching between selection and fusion in combining classifiers: An experiment," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 32,, pp. 146-156, 2002.
[۴۰] L. I. Kuncheva, "Clustering-and-selection model for classifier combination," in Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000, pp. 185-188.
[۴۱] B. Yuan, and R. Liu, "Multiple classifiers combination by clustering and selection," *Elsevier*, vol. 2, pp. ۱۶۸-۳.
[۴۲] L. Breiman, "Bagging predictors," *Machine learning,Springer*, vol. 24, pp. 123-140, 1996.
[۴۳] B. Efron, and R. J. Tibshirani, "An introduction to the bootstrap," *Chapman & Hal1*, 1993.
[۴۴] S. Suen, and L. Lam, "Application of majority voting to pattern recognition: An analysis of its behavior and performance," *Systems and Humans, IEEE Transactions on*, vol. 27, no. IEEE, pp. 553-568, 1997.
[۴۵] L. Breiman, "Random forests," *Machine learning,Springer*, vol. 45, pp. 5-32, 2001.
[۴۶] L. Breiman, "Pasting small votes for classification in large databases and on-line," *Machine learning,Springer*, vol. 36, pp. 85-103, 1999.
[۴۷] W. Kegelmeyer, T. Moore, K. Bowyer *et al.*, "Distributed pasting of small votes," *Multiple Classifier Systems, Springer*, pp.۲۰۰۲ ,۶۱-۵۲ .
[۴۸] R. Schapire, and Y. Freund, "A desicion-theoretic generalization of on-line learning and an application to boosting," *Computational learning theory*, pp. 23-37, 1995.
[۴۹] L. Breiman, "Arcing classifier (with discussion and a rejoinder by the author)," *The Annals of Statistics, Institute of Mathematical Statistics*, vol. 26, pp. 801-849, 1998.

[۵۰] J. C. Schlimmer, “Efficiently inducing determinations: A complete and systematic search algorithm that uses optimal pruning,” *Proceedings of the Tenth International Conference on Machine Learning*, pp. 284-290, 1993.
[۵۱] D. M., and H. Liu, “Consistency-based search in feature selection,” *Artificial intelligence, Elsevier*, vol. 151, pp. 155-176, 2003.
[۵۲] W. Megchelenbrink, “Relief-based feature selection in bioinformatics:detecting functional speci city residues from multiple sequence alignments,” Radboud University Nijmegen, Nijmegen, 2011.
[۵۳] I. Kononenko, and M. Robnik-Šikonja, "An adaptation of Relief for attribute estimation in regression." pp. 296-304.
[۵۴] T. Chang, P. Yan, and L. Xu, “Best first strategy for feature selection,” in 9th International Pattern Recognition Conference, 1988, pp. 706-708.
[۵۵] L. Bobrowski, “Feature selection based on some homogeneity coefficient,” in 9th International Pattern Recognition Conference, 1988,, pp. 544-546.
[۵۶] C. Cardie, "Using decision trees to improve case-based learning." pp. 32-37.
[۵۷] M. Sahami, and D. Koller, "Toward optimal feature selection." pp. 284-292.
[۵۸] J. Novovicova, and A. Malik, "Information-theoretic feature selection algorithms for text classification." pp. 3272-3277.
[۵۹] E. E. Gose, and A. N. Mucciardi, “A comparison of seven techniques for choosing subsets of pattern recognition properties,” *IEEE Transactions on Computers*, vol. 100, pp. 1023-1031, 1971.
[۶۰] A. Sangiovanni, Vincentelli, and A. L. Oliveira, "Constructive induction using a non-greedy strategy for feature selection." pp. 355-360.
[۶۱] R. Setiono, and H. Liu, “A probabilistic approach to feature selection-a filter solution,” in Machine Lerning International Workshop Then Conference, 1996, pp. 319-327.
[۶۲] L. V. Fausett, *Fundamentals of neural networks*, 1 ed.: Prentice-Hall Englewood Cliffs, NJ, 1994.
[۶۳] R. Caruana, and D. Freitag, "Greedy attribute selection." pp. 28-36.
[۶۴] J. Doak, “An evaluation of feature selection methods and their application to computer security,” *University of California at Davis, California, Technical Report CSE-92-18*, 1992.
[۶۵] M. S. Lee, and A. W. Moore, "Efficient algorithms for minimizing cross validation error." pp. 190-198.
[۶۶] A. W. Moore, and O. Maron, “The racing algorithm: Model selection for lazy learners,” *Artificial Intelligence Review*, vol. 11, no. Springer, pp. 193-225, 1997.
[۶۷] L. Parsons et al., “Evolving feature selection,” *IEEE Intelligent systems*, vol. 20, no. IEEE, pp. 64-76, 2005.
[۶۸] N. Milic-Frayling, M. Grobelnik, J. Brank *et al.*, "Feature selection using linear classifier weights: interaction with classification models." pp. 234-241.

[۶۹] J. Kittler, J. Novovicova, and P. Pudil, “Floating search methods in feature selection,” *Pattern recognition letters,* vol. 15, no. Elsevier, pp. 1119-1125, 1994.
[۷۰] I. Foroutan, and J. Sklansky, “Feature selection for automatic classification of non-gaussian data,” *IEEE Transactions on Systems, Man and Cybernetics,* vol. 17, pp. 187-198, 1987.
[۷۱] F. Shahabi nejad, and M. Rahmati, “A New Method for Writer Identification and Verification Based on Farsi/Arabic Handwritten Texts,” *Document Analysis and Recognition* ,pp. 23-26, 2007.
[۷۲] A. Schlapbach, and H. Bunke, "Using HMM based recognizers for writer identification and verification." pp. 167-172.
[۷۳] S. N. Srihari, H. Arora, S. H. Cha *et al.*, “Individuality of Handwriting: a validation study,” in 6th IEEE Conf. on Document Analysis and Recognition, 2001, pp. 106-109.
[۷۴] L. Schomaker, and M. Bulacu, “Automatic writer identification using connected-component contours and edge-based features of upper-case western script,” *IEEE Trans on Pattern Analysis and Machine Intelligence,* vol. 26, no. 6, pp. 787-798, 2004.

Abstract:

One of the most important research fileds in the world is person identification. In this reserch mixture of expert network is suggested for writer classification and identification in Persian handwriting, which is a technique for person identification based of biometrics features. It is suggested to combine three neural networks that are trained based on three different text features extracted by Roberts, Sobel and Gabor filters. The numbers of extracted features by different approaches are reduced using principal component analysis algorithm, and k nearest neighbor algorithm is also used to find the optimal number of confident features. Three neural networks are trained based on gradient descent and error back propagation procedures. Outputs of these networks are combined by using a set of weights which are obtained by four different methods. In the first experiment we combine neural network outputs by equal weights so each neural network which is trained by different feature sets has equal effect in the result of classification and writer identification. In the second experiment confidence of each network has been used as weight of that neural network. In the third experiment the PSO optimization algorithm is used to minimize the mean square error in training stage and provide a set of weights for combining the networks. In the last experiment the PSO is used again with this difference that we assign different weigths to outputs of each network so instead of 3 weights in third method we must find 120 weigths for output neurons of neural networks and combining corresponding neurons of each neural networks together to obtain finally 40 output neurons. Simulation results show the high accuracy of writer identification up to 97.42%, 97.75%, 97.59% and 98.17% respectively for different proposed methods which are evidently higher than accuracy obtained in each single neural network.

Keywords:

MLP, mixture of experts, writer identification, PSO, PCA, manuscript

Shahrood University of Technology

Faculty of Electrical and Robotic Engineering

Introducing new features for writer identification using mixture of experts

By:

Vahid Zareii

Supervisors:

Dr. Omidreza Maruzi

Assistance:

Dr. Hosseyn Khosravi

FOR THE DEGREE OF

MASTER OF SCIENCE

September 2012