

## الگوریتمهای ترکیبی برای دسته‌بندی الگوها (آتاماتونهای یادگیر + روشهای کلاسیک)

محمدرضا میبیدی

غلامرضا رضایی

دانشکده مهندسی کامپیوتر

دانشگاه صنعتی امیرکبیر

تهران - ایران

**چکیده:** تکنیکهای شناسایی الگو به دستبندی یک کلاس الگو از سایر کلاسهای الگوی موجود در حوزه کاربرد مورد نظر می‌پردازد. استفاده از تولع تمایز، یکی از روشهای دستبندی الگوها می‌باشد. برای تعیین تولع تمایز، روشهای مختلفی پیشنهاد شده که گروهی از آنها براساس آتاماتونهای یادگیر می‌باشند [۲۲] [۲۱] [۱۹] [۳۰۰۹]. در [۲۱] کارایی آتاماتونهای یادگیر تکسطحی در تعیین پارامترهای تولع تمایز از طریق بازی آتاماتونهای یادگیر با پاسخ یکسان بررسی شد و آتاماتونهای یادگیر با ساختار ثابت جدیدی که کارایی بالاتری در تعیین تولع تمایز (نزدیک به) بهینه از خود نشان می‌دهند، پیشنهاد گردید. در [۲۲] با توسعه آتاماتونها و الگوریتمهای یادگیری ارفه شده در [۲۱] آتاماتونهای یادگیر سلسله‌مراتبی با ساختار ثابت جدید ارفه گردیده و کارایی آنها در تخمین پارامترهای تولع تمایز نشان داده شده است. در این مقاله، از ترکیب آتاماتونهای یادگیر با روشهای کلاسیک، الگوریتمهایی جدید بمنظور تعیین تولع تمایز ارفه می‌گردد. الگوریتمهای پیشنهادی با استفاده از آتاماتونهای یادگیر تکسطحی یا سلسله‌مراتبی و روشهای کلاسیک، سعی در تعیین خودکار حدود هر کلاس الگو می‌نمایند. از هر یک از آتاماتونهای یادگیر ارفه شده در [۲۱] و [۲۲] می‌توان به عنوان بازیگران بازی (اجزای سیستم یادگیری) استفاده نمود. همچنین، در این مقاله، کاربرد روشهای پیشنهادی بمنظور تشخیص آوهای گفتاری زبان فارسی مورد بررسی قرار گرفته است.

**کلمات کلیدی:** شناسایی الگو، آتاماتونهای یادگیر، تولع تمایز، الگوریتمهای ترکیبی، تشخیص آوا

### ۱ - مقدمه

تکنیکهای شناسایی الگو، با ارفه قواعد تصمیم‌گیری، سعی در جداسازی یک کلاس الگو از سایر کلاسهای الگو می‌نماید. در صورتیکه اطلاعات مسئله بطور کامل موجود باشد، قانون بیز [۱] [۲] [۱۳] می‌تواند قاعده تصمیم‌گیری که احتمال دستبندی غلط را می‌نیم می‌کند را تولید نماید. در شرایطی که اطلاعات کافی درباره مسئله در دست نیست، قواعد تصمیم‌گیری با استفاده از یک سری نمونههای آموزشی موجود در حوزه کاربرد موردنظر، یادگرفته می‌شود. یک روش دستبندی الگوها، استفاده از تولع تمایز می‌باشد. در یک مسئله در کلاسه، اگر  $g(\cdot)$  تولع تمایز باشد، قاعده تصمیم‌گیری به صورت  $\text{if } g(x) \geq 0 \text{ then } x \in W_1 \text{ else } x \in W_2$  می‌باشد.

روشهای متعددی برای تعیین تولع تمایز ارفه شده‌اند که گروهی از آنها بر اساس آتاماتونهای یادگیر می‌باشند. Barto و Andrew روشی بنام  $A_{R-P}$  ارفه کرده‌اند که هدف آن شناسایی تولع تمایز خطی  $g(X) = \theta^T X$  با استفاده از آتاماتونهای یادگیر می‌باشد [۳]. در این روش یک رابطه تجمعی (Associative Relation) بین هر عضو از نمونه ورودی و هر عمل تصمیم‌گیری ایجاد می‌شود و آتاماتونهای یادگیر سعی می‌کنند که با مشاهده هر نمونه ورودی، تصمیم صحیح را اتخاذ کنند. Sastry و Thatachar روشی را جهت یادگیری تولع تمایز بهینه از فضای تولع تمایز با استفاده از بازی آتاماتونهای یادگیر با پاسخ یکسان ارفه کرده‌اند [۴]. الگوریتم ارفه شده در این روش بر اساس آتاماتونهای یادگیر با ساختار متغیر می‌باشد که در آن، احتمال هر عمل با استفاده از تخمینی از ماتریس پاداش (Pay-off Matrix) بازی که در طول یادگیری ساخته می‌شود، به‌هنگام می‌شود. هر آتاماتون شرکت کننده در بازی، یکی از پارامترهای تولع تمایز را تعیین می‌کند. مجموعه اعمال هر آتاماتون، مجموعه مقادیر ممکن برای آن پارامتر می‌باشد. Phananskar و Thatachar روشی را براساس شبکه‌ای پیشخور (Feed Forward Network) از آتاماتونهای یادگیر، جهت پیاده‌سازی یک سیستم دستبندی سطحی براساس دستبندی خطی قطعه به قطعه (3-Layer Linear Piecewise Classification) ارفه کرده‌اند [۵] [۶]. در این روش با استفاده از تولع تمایز خطی قطعه به قطعه محدوده‌های هر کلاس تخمین زده می‌شود. در این روش از آتاماتونهای یادگیر با ساختار متغیر با شیوه به‌هنگام سازی  $L_R-L$  استفاده و نشان داده شده که سیستم می‌تواند به یک نقطه بهینه محلی (Local Optimum) (و نه الزاماً سراسری) اگر شرایط اولیه مناسب باشد، همگرا شود. Sastry و Thatachar با ارفه PLA (Parameterized LA) روشی را جهت شناسایی نقطه بهینه سراسری (Global Optimum) با استفاده از شبکه‌های پیشخور پیشنهاد کرده‌اند [۷] [۸]. در این روش، بجای آتاماتونهای استاندارد از آتاماتون جدیدی به نام PLA که دارای ساختار متغیر بوده و قابلیت دریافت بردار محتوی (Context Vector) از محیط را دارد، استفاده و نشان داده شده است که تحت شرایط ویژه‌ای، مجموعه آتاماتونها می‌تواند به نقطه بهینه سراسری همگرا شود.

رضایی و میبیدی در [۲۱] و [۲۲] کارایی آتاماتونهای یادگیر با ساختار ثابت در تخمین پارامترهای (بهینه) تولع تمایز با استفاده از بازی آتاماتونهای یادگیر با پاسخ

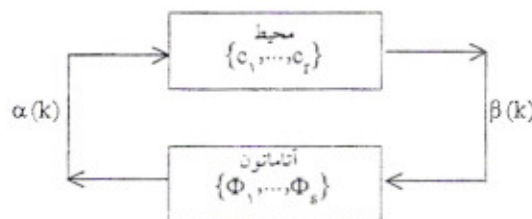
پکسان مورد بررسی قرار داده‌اند. در [۲۱] آتاماتونهای یادگیر تک سطحی با ساختار ثابت جدیدی ارائه شده که قادرند پارامترهای تابع تمایز را با کارایی بالایی تعیین نمایند. در [۲۲] با توسعه آتاماتونهای ارائه شده در [۲۱]، آتاماتونهای یادگیر سلسله‌مراتبی جدیدی که قادرند در محدوده وسیعتری پارامترهای تابع تمایز را تعیین کنند ارائه گردیده است.

روشهای ارائه شده در [۲۱] و [۲۲] برای تعیین پارامترهای تابع تمایزی که فرم کلی آن از قبل معلوم باشد، قابل استفاده می‌باشد. همچنین محدوده پارامترهای تابع تمایز که باید تخمین زده شوند، باید از قبل مشخص باشد. به این ترتیب، باید اطلاعات زیادی (نوع تابع تمایز و حدود پارامترهای آن) درباره مسئله مورد حل در دست باشد. در این مقاله روش آتاماتونهای یادگیر ارائه شده در [۲۱] و [۲۲] با روشهای کلاسیک، محدوده هر کلاس را بطور خودکار تعیین می‌نماییم. روشهای کلاسیک متعددی برای تعیین تابع تمایز خطی ارائه شده‌اند که غالباً برای نوع خاصی از توزیع الگوها (مثلاً توزیع نرمال) بهینه می‌باشند [۱] [۲] و به طور عام نمی‌توان از این روشها برای حل مسائل استفاده نمود. با در نظر گرفتن محدودهای برای هر پارامتر تابع تمایز خطی، می‌توان تابع تمایز اولیه بدست آمده از روشهای کلاسیک را اصلاح نمود. آتاماتون یادگیر به عنوان یک مدل یادگیری از صفر، می‌تواند عمل اصلاح هر پارامتر را انجام دهد. همچنین با ترکیب دسته‌خطیهای تمایز و اصلاح موقعیت آنها با استفاده از بازی آتاماتونهای یادگیر می‌توان بطور خودکار منحنی پوش هر کلاس الگو را تقریب زد و به این ترتیب، هر کلاس را از سایر کلاسهای الگو جدا نمود. همچنین با استفاده از ساختارهای درختی می‌توان حدود تصمیم‌گیری برای جداسازی کلیه کلاسهای الگوی موجود در مسئله مورد نظر را تعیین نمود. در این مقاله روش آتاماتونهای یادگیر با الگوریتمهای کلاسیک ترکیب شده و در قالب الگوریتمهای تکسطحی و سلسله‌مراتبی یک کلاس الگو را از سایر کلاسهای الگو جدا می‌نمایند. سایر بخشهای بعدی مقاله به صورت زیر تنظیم شده است.

در بخش ۲ به معرفی اجمالی آتاماتونهای یادگیر می‌پردازیم و بازی آتاماتونهای یادگیر با پاسخ پکسان را معرفی می‌نماییم. در بخش ۳ الگوریتم «بهبود تابع خطی» ارائه می‌گردد. در این الگوریتم ابتدا یک تابع خطی (غیر دقیق) با استفاده از روشهای کلاسیک، برای جداکردن یک کلاس از سایر کلاسها، تخمین زده می‌شود. سپس پارامترهای تابع خطی بدست آمده بوسیله بازی آتاماتونهای یادگیر با پاسخ پکسان تصحیح می‌شود. همچنین روشی برای افزایش درجه تابع خطی تخمین زده شده (به درجه ۲ و ۳ و ...) ارائه می‌گردد. در بخش ۴ الگوریتم «دستبندی قطعه به قطعه تطبیقی» پیشنهاد می‌گردد. در این الگوریتم پوش توزیع هر کلاس الگو بوسیله مجموعه‌ای از خطوط، تقریب زده می‌شود. این تقریب توسط روشهای کلاسیک انجام می‌گیرد. سپس خطوط غیر ضروری حذف شده و مجموعه خطوط باقی‌مانده، بوسیله آتاماتونهای یادگیر اصلاح می‌شوند. در بخش ۵ الگوریتم «دستبندی درختی» پیشنهاد می‌گردد. توسط این الگوریتم، ابتدا برحسب میزان جدایی‌پذیری کلاسهای موجود در کاربرد مورد نظر، یک ساختار درختی از کلاسها ایجاد می‌گردد. به این ترتیب کلاسهای قرار گرفته در سطوح بالاتر این ساختار، با دقت بیشتر و کلاسهای قرار گرفته در سطوح پایین‌تر، با دقت کمتری قابل جداسازی می‌باشند. سپس تابع تمایز جداکننده هر گره توسط الگوریتمهای ارائه شده در بخشهای ۳ و ۴ تخمین زده شده و تصحیح می‌گردد. در بخش ۶ کارایی الگوریتمهای پیشنهادی در مسئله جداسازی آوای زبان فارسی مورد بررسی قرار می‌گیرند. در این قسمت از آنالیز MFCC جهت استخراج بردارهای ویژگی استفاده گردیده است. بخش ۷ خلاصه مقاله می‌باشد.

## ۲ - آتاماتون یادگیر

یک آتاماتون یادگیر، یک آتاماتون احتمالی (Stochastic Automaton) است که با محیط خود به صورت پسخور (Feedback) تراکنش دارد. خروجی آتاماتون عملی است که به عنوان ورودی، به محیط داده می‌شود و خروجی محیط پاسخ محیط به عمل انجام شده است که به عنوان ورودی به آتاماتون ارسال می‌گردد. آتاماتون یادگیر براساس پاسخ محیط استراتژی خود را در تصمیم‌گیریهای آینده به‌هنگام می‌نماید. هدف از فعالیت آتاماتون، انتخاب اعمالی از مجموعه اعمال آتاماتون است که بهترین پاسخ را از محیط دریافت کند. رابطه محیط و آتاماتون که بصورت پسخور می‌باشد در شکل (۱) نشان داده شده است.



شکل ۱: رفتار متقابل محیط و آتاماتون

از نظر ساختاری، آتاماتونهای یادگیر به دو دسته آتاماتونهای یادگیر با ساختار ثابت (Fixed Structure Learning Automata) و آتاماتونهای یادگیر با ساختار متغیر (Variable Structure Learning Automata) تقسیم می‌شوند. در آتاماتونهای با ساختار ثابت، نحوه تغییر حالت آتاماتون از قبل مشخص می‌شود. براساس حالت آتاماتون در مرحله  $k$  ام، عمل  $\alpha(k)$  با استفاده از تابع انتخاب عمل آتاماتون، به محیط اعمال می‌شود و با دریافت پاسخ  $\beta(k)$  از جانب محیط، حالت بعدی آتاماتون تعیین می‌گردد. در آتاماتونهای یادگیر با ساختار متغیر، نحوه تغییر حالت آتاماتون از قبل مشخص نیست. در این مدل، از بردار احتمال عمل (Action Probability Vector) یا بردار احتمال حالت (State Probability Vector) آتاماتون استفاده می‌گردد. بردار احتمال عمل یا حالت، احتمال انتخاب یک عمل یا قرار گرفتن آتاماتون در یک حالت را در مرحله  $k$  ام تعیین می‌کند. در آتاماتونهای یادگیر با ساختار متغیر، یک توزیع اولیه (معمولاً یکنواخت) به عنوان مقادیر اولیه بردار احتمال عمل در نظر گرفته می‌شود. در هر مرحله، آتاماتون یک عمل را با توجه به احتمالات عملها، انتخاب و به محیط اعلام می‌نماید و پاسخ آن را از محیط دریافت می‌کند. براساس پاسخ دریافت شده، آتاماتون، این احتمالات را به‌هنگام می‌نماید و در مرحله بعدی براساس احتمالات به‌هنگام شده، عمل بعدی را انتخاب می‌کند. هدف از این به‌هنگام سازی، یافتن عمل مناسب با هدف دریافت بیشترین احتمال پاسخ مطلوب از محیط است. به نحوه تغییر احتمالات، الگوریتم یادگیری (Learning Algorithm) یا روش تقویتی (Reinforcement Scheme) گفته می‌شود که می‌تواند تابعی خطی یا غیرخطی از احتمالات باشد.

محیط براساس مجموعه  $\beta$  (مجموعه پاسخ محیط) به سه دسته تقسیم می‌شود: مدل  $P$  که در آن، مجموعه  $\beta$  در عضوی است که یک عضو بیاتر پاسخ مطلوب و عضو دیگر بیاتر پاسخ نامطلوب می‌باشد، مدل  $Q$  که در آن، مجموعه خروجی به صورت مجموعه محدود  $\{\beta_1, \dots, \beta_m\}$  تعریف می‌شود و مدل  $S$  که در آن، مجموعه خروجی، یک مجموعه نامحدود به صورت  $\beta = \{(a, b)\}$  می‌باشد. در این مدل، پاسخ محیط می‌تواند یکی از مقادیر پیوسته در ناحیه  $(a, b)$  را اختیار کند. در مدل‌های  $Q$  و  $S$  معمولاً مقدار بیشتر برای پاسخ محیط، به عنوان پاسخ بهتر در نظر گرفته می‌شود.



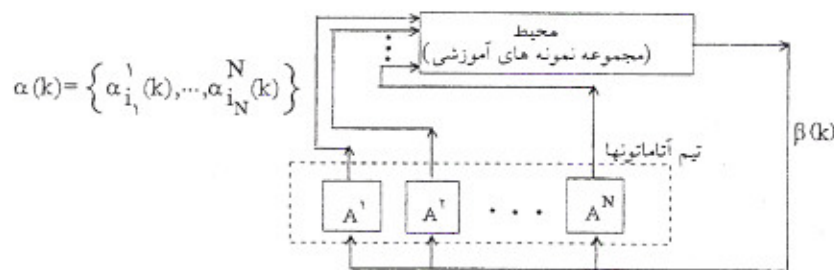
بر اساس مجموعه  $C$  (مجموعه مشخصات داخلی محیط)، محیط به دو دسته ایستا (Stationary) و پویا (Non Stationary) طبقه‌بندی می‌شود. در مدل ایستا مقادیر  $C_i$  ها (احتمال دادن پاسخ نامطلوب به عمل  $(i)$ ) و در نتیجه مشخصات محیط در طول زمان ثابت است، در صورتیکه در مدل پویا، مشخصات داخلی محیط در طول یادگیری در حال تغییر می‌باشد.

از نظر تئوری، یک مسئله را می‌توان با یک آتاماتون منفرد که با محیط در ارتباط است حل کرد که در این صورت، ممکن است مجبور به استفاده از آتاماتونی با تعداد عملهای بسیار زیاد بشویم که این، منجر به سرعت همگرایی بسیار پایین می‌گردد [۹]. برای حل مشکل همگرایی، می‌توان از مجموعه‌ای از آتاماتونها که به صورت یک تیم در حل مسئله شرکت می‌کنند، استفاده کرد. مدل‌های مختلفی از بازی آتاماتونها ارائه شده است. در مدلی که در این مقاله در نظر گرفته شده است، همه بازیگران در انتهای هر بازی پاسخ یکسانی را از محیط دریافت می‌کنند [۹].

فرض کنیم تابع تمایز به فرم  $h(x) = g(\theta_1, \dots, \theta_N, x)$  باشد بطوریکه  $\theta_1, \dots, \theta_N$  پارامترهای تابع  $g$  هستند که باید یاد گرفته شوند و  $x$  بردار نمونه ویژگی‌ها است که باید دسته‌بندی شود. با استفاده از  $N$  آتاماتون یادگیر در قالب یک بازی با پاسخ یکسان مطابق شکل (۲) می‌توان پارامترهای فوق را تخمین زد.

مطابق شکل (۲)، آتاماتونهای  $A^1 \dots A^N$  در بازی شرکت کرده‌اند. هر آتاماتون  $A^i$  دارای  $r_i$  عمل  $\alpha_1^i, \dots, \alpha_{r_i}^i$  می‌باشد. هر عمل آتاماتون  $A^i$  یک مقدار ممکن

برای پارامتر  $\theta_i$  از تابع نمایز در نظر گرفته می‌شود.



شکل ۲: بازی آتاماتونهای یادگیر با پاسخ یکسان

مقادیر ممکن برای پارامتر  $\theta_i$  (و در نتیجه اعمال آتاماتون  $A^i$ ) از قبل تعیین شده‌اند. هدف از این بازی، پیدا کردن مقادیری برای پارامترهای  $\theta_i$  می‌باشد بطوریکه تعداد دسته‌بندیهای غلط به‌کمینه گردد. در هر مرحله، آتاماتون  $A^i$  با توجه به شیوه تصمیم‌گیری خود، یک عمل (مقدار) از مجموعه اعمال خود (مجموعه مقادیر ممکن برای  $\theta_i$ ) را انتخاب و به محیط اعلام می‌نماید. یک (یا چند) نمونه از بردارهای ویژگی موجود در محیط توسط تابع نمایز بدست آمده (با استفاده از پارامترهای انتخاب شده) دسته‌بندی می‌شود و نتیجه بصورت نسبت تعداد دسته‌بندی‌های صحیح به کل نمونه‌های آزمایش شده، به تیم ارسال می‌شود. بر اساس پاسخ بدست آمده، آتاماتونهای شرکت کننده در بازی، شیوه تصمیم‌گیری خود را به‌هنگام می‌نمایند. هدف از بازی، بدست آوردن تابع تمایزی است که بر اساس آن احتمال دریافت پاسخ نامطلوب و یا به عبارت دیگر تعداد دسته بندی‌های غلط نمونه‌ها به‌کمینه گردد. برای اطلاعات بیشتر درباره آتاماتونهای یادگیر، می‌توانید به [۹] [۱۰] [۱۱] [۱۲] [۱۳] [۱۴] [۱۵] مراجعه نمایید. آنچه در ادامه این بخش می‌آید، شرح مختصری است از یکی از آتاماتونهای یادگیر که در [۱۹] پیشنهاد شده است. این آتاماتون که  $SLR-I$  نامیده می‌شود، در این مقاله مورد استفاده قرار گرفته است.

## ۱-۲ - آتاماتون $SLR-I$

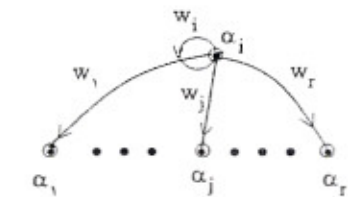
آتاماتونی با  $r$  حالت  $\Phi_1, \dots, \Phi_r$  و  $r$  عمل  $\alpha_1, \dots, \alpha_r$  بطوریکه در هر حالت  $\Phi_j$  عمل  $\alpha_j$  توسط آتاماتون انتخاب شود در نظر بگیرید. برای هر عمل  $\alpha_i$ ، وزن  $w_i$  با توزیع اولیه یکنواخت (مقدار  $\frac{1}{r}$  برای هر  $w_i$ ) در نظر گرفته می‌شود. در صورتیکه در مرحله  $k$  ام، عمل  $\alpha_j(k)$  توسط آتاماتون انتخاب شود (آتاماتون در حالت  $\Phi_j(k)$  باشد) با دریافت پاسخ محیط، آتاماتون تشویق یا تنبیه می‌شود. در صورت تشویق شدن، وزن عملهای آتاماتون مطابق رابطه (۱) به‌هنگام می‌شود بدین معنی که به وزن عمل انتخاب شده مقداری اضافه شده و از وزن سایر عملها کم می‌شود. در صورت تنبیه شدن، بدون اینکه وزن عملهای آتاماتون تغییر کند، با احتمال  $w_j(k)$  در حالت فعلی باقی‌مانده و با احتمال  $w_j(k)$  به حالت  $\Phi_j$  تغییر حالت می‌دهد. عملی که وزن آن از حد مشخصی (مثلاً ۰/۹۹) بیشتر شود، به عنوان عمل بهینه انتخاب می‌شود.  $W$  عددی است صحیح که پارامتر یادگیری نامیده می‌شود.

$$w_i(k+1) = w_i(k) + \frac{1}{W}(1 - w_i(k))$$

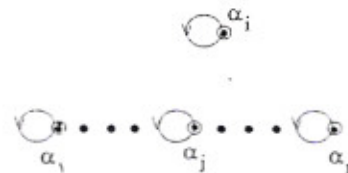
$$w_j(k+1) = \left(1 - \frac{1}{W}\right)w_j(k) \quad j \neq i \quad (1)$$

شکلهای (۳) و (۴) نحوه تغییر حالت آتاماتون با دریافت پاسخ مطلوب و نامطلوب از جانب محیط را نشان می‌دهد. فرض شده است که آتاماتون در حالت  $\Phi_i$  قرار دارد، به این ترتیب عمل  $\alpha_i$  را انتخاب می‌نماید. با دریافت پاسخ مطلوب از جانب محیط، آتاماتون تشویق می‌گردد و وزن عملهای آن مطابق رابطه (۱) به‌هنگام می‌گردد. با دریافت پاسخ نامطلوب، آتاماتون با احتمال  $w_i$  در حالت  $\Phi_i$  باقی می‌ماند و با احتمال  $w_j$  ( $j \neq i$ ) به حالت  $\Phi_j$  تغییر حالت می‌دهد. باید

توجه داشت که همواره  $\sum_{i=1}^r w_i = 1$  می‌باشد. برای کسب اطلاعات بیشتر درباره این آتاماتون می‌توانید به [۱۹] مراجعه نمایید.



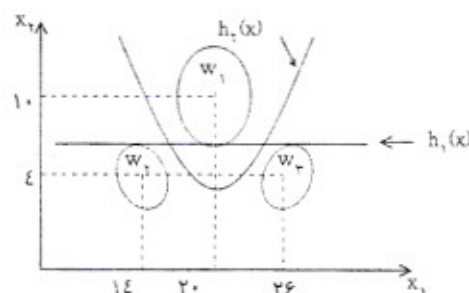
شکل ۲: گراف تغییر حالت آنامتون  $SL_R-1$  با دریافت پاسخ نامطلوب از جانب محیط



شکل ۳: گراف تغییر حالت آنامتون  $SL_R-1$  با دریافت پاسخ مطلوب از جانب محیط

### ۳- بهبود توابع خطی

روشهای کلاسیک متعددی برای تعیین تابع تمایز خطی برای جداسازی یک کلاس از کلاسهای دیگر موجود در حوزه مسئله مورد حل، ارائه شده است. گروهی از این روشها بصورت تکراری (Iterative) عمل میکنند و گروهی دیگر بر اساس مطالعات آماری بین کلاسها میباشند [۱] [۲]. غالباً الگوریتمهای ارائه شده برای جداسازی دو کلاس با تابع توزیع نرمال و اغلب با شرط یکسان بودن ماتریسهای کوواریانس آنها، بهینه میباشند. شکل (۵) شکل توزیع کلاسهای یک مسئله سه کلاسه فرضی را نشان می دهد. همانطور که در شکل (۵) دیده میشود، تابع تمایز  $h_1(x)$  تابع تمایز خطی است که میتواند کلاس  $W_1$  را از دو کلاس دیگر جداسازد، اما تابع تمایز  $h_p(x)$  که یک تابع درجه ۲ می باشد - با دقت بسیار بیشتری کلاس  $W_1$  را دستبندی می نماید. با در نظر گرفتن تابع  $h_1(x)$  به عنوان یک حدس اولیه برای جداسازی کلاس  $W_1$  و تصحیح آن، می توان تابع تمایز  $h_p(x)$  را بدست آورد.



شکل ۵: توزیع کلاسهای مسئله نمونه شماره ۱

اگر  $h_1(x)$  تابع تمایز اولیه خطی برای جداسازی کلاس  $W_1$  از سایر کلاسها باشد، فرم عمومی  $h_1(x)$  بصورت (۲) می باشد:

$$h_1(x) = \begin{cases} v^T x + v_0 < 0 \Rightarrow x \in W_1 \\ v^T x + v_0 > 0 \Rightarrow x \notin W_1 \end{cases}, \quad v = (v_1, \dots, v_n)^T \in \mathbb{R}^n, \quad x = (x_1, \dots, x_n)^T \in \mathbb{R}^n \quad (2)$$

و یا:

$$h_1(x) = \sum_{i=1}^n v_i x_i + v_0 \quad (3)$$

فضای مسئله،  $\mathbb{R}^n$  فرض میشود و با در نظر گرفتن  $v_0$  به عنوان حد آستانه، تابع  $h_1(x)$  دارای  $n+1$  پارامتر می باشد. در صورتیکه هدف تعیین تابع درجه ۲ باشد، فرم عمومی تابع بصورت (۴) خواهد بود.

$$h_p(x) = \sum_{i=1}^n \sum_{j=1}^n \theta_{ij} x_i x_j + \sum_{i=1}^n \theta_i x_i + \theta_0 \quad (4)$$

بنابراین تابع  $h_p(x)$  دارای  $\frac{n(n+1)}{2} + n + 1$  پارامتر می باشد. برای تعیین  $h_p(x)$  با استفاده از حدس اولیه  $h_1(x)$ ، می توان مقدار پارامترهای  $\theta_0, \dots, \theta_n$  را حول  $v_0, \dots, v_n$  و مقدارهای  $\theta_{11}, \dots, \theta_{nn}$  را حول مقدار  $\frac{1}{2}$  در نظر گرفت. الگوریتم (۱)، نحوه تبدیل تابع  $h_1(x)$  به  $h_p(x)$  را با استفاده از بازی آنامتونها یادگیر با پاسخ یکسان، بیان میکند.

در الگوریتم (۱) رویه Learn-h2 پارامترهای تابع تمایز  $h_p(x)$  را تعیین می نماید. تابع Discriminate در هر مرحله از یادگیری، بردار نمونه ویژگی ورودی را دستبندی نموده و پاسخ صحیح یا غلط را باز می گرداند. رویه UpdateGame با توجه به نتیجه دستبندی (DiscRes) وضعیت آنامتونها را به هنگام می نماید. در رویه Learn-h2، ابتدا تابع  $h_1(x)$  با یک روش آماری (غیر دقیق) تعیین می گردد. آنامتونها  $LA_{ij}$ ،  $1 \leq i, j \leq n$  که در الگوریتم ایجاد میشوند، وظیفه یادگیری پارامترهای  $\theta_{ij}$  (ضرایب  $x_i x_j$  در رابطه ۴) از تابع تمایز  $h_p(x)$  را برعهده دارند. از آنجا که  $h_p(x)$  با استفاده از  $h_1(x)$  ساخته میشود و این ضرایب در  $h_1(x)$  وجود ندارند (صفر می باشند)، مقدارهای ممکن برای این پارامترها حول مقدار صفر در نظر گرفته میشود. تعداد مقادیر (تعداد عملهای آنامتون  $LA_{ij}$ ) برابر  $1 + 2r' + \frac{1}{K}$  در نظر گرفته میشود و مقدار صفر به عنوان یکی از عملهای آنامتون در نظر گرفته میشود.  $r'$  و  $K$  پارامترهای الگوریتم می باشند. آنامتونها

$h_i(x)$  نیز مامور یادگیری ضرایب  $\theta_i$  در تابع  $h_i(x)$  می‌باشد. مجموعاً  $r = 2r' + 1$  مقدار ممکن با قدمهای  $\frac{v_i}{K}$  حول مقدار پارامتر  $v_i$  از تابع  $h_i(x)$  (رابطه ۳) برای پارامتر  $\theta_i$  در نظر گرفته می‌شود. به این ترتیب یک فضا برای پارامترهای تابع تمایز ایجاد می‌شود و مجموعه آتاماتونهای یادگیر سعی در پیدا کردن پارامترهای بهینه برای تابع تمایز مورد نظر در فضای تعریف شده می‌نمایند.

#### Procedure Learn-h2

```

Guess  $h_i(x)$  using training samples;
for  $i := 1$  to  $n$  do
  for  $j := 1$  to  $n$  do
     $LA_{ij} :=$  create a learning automata;
    for  $k := -r'$  to  $r'$  do  $LA_{ij}.Actions.Add(k/K)$ ;
  end for  $\{j\}$ 
end for  $\{i\}$ 
for  $i := 0$  to  $n$  do
   $LA_i :=$  create a learning automata;
  for  $j := -r'$  to  $r'$  do  $LA_i.Actions.Add(v_i * (1 + j/K))$ ;
end for  $\{i\}$ 

Repeat
   $x :=$  Next training sample;
  DiscResult := Discriminate( $x$ );
  UpdateGame(DiscResult);
Until EndOfGame;
Optimal Actions are parameters of function  $h_i(x)$ ;
End Learn-h2;
```

#### Function Discriminate( $x$ ): Boolean;

```

 $hx := \sum_{i=1}^n \sum_{j=1}^n (LA_{ij}.Action) \cdot x_i x_j +$ 
 $\sum_{i=1}^n (LA_i.Action) \cdot x_i + LA_0.Action$ 
Result := (( $hx \leq 0$ ) and ( $x.ClassID = 1$ )) or
          (( $hx > 0$ ) and ( $x.ClassID \neq 1$ ));
End Discriminate;
```

#### الگوریتم ۱

الگوریتم (۱) تابع تمایز خطی  $h_1(x)$  را به تابع درجه ۲  $h_2(x)$  تبدیل می‌نماید. بطور کلی برای تبدیل تابع خطی  $h_1(x)$  به یک تابع چند جمله‌ای با درجه  $n > 1$  می‌توان از روشی مشابه الگوریتم (۱) استفاده نمود، به عنوان مثال برای تابع تمایز درجه ۳، فرم کلی بصورت (۵) می‌باشد:

$$h_2(x) = \sum_{i=1}^n \sum_{j=1}^n \sum_{k=1}^n \theta_{ijk} x_i x_j x_k + \sum_{i=1}^n \sum_{j=1}^n \theta_{ij} x_i x_j + \sum_{i=1}^n \theta_i x_i + \theta_0 \quad (5)$$

برای تعیین پارامترهای (۵)، مقدارهای ممکن برای  $\theta_{ij}$ ،  $\theta_{ijk}$  حول صفر و مقدارهای ممکن برای  $\theta_i$  حول  $v_i$  در نظر گرفته می‌شود. در قسمت یادگیری تابع تمایز در الگوریتم (۱)، رویه UpdateGame وضعیتهای داخلی آتاماتونها را به‌هنگام می‌نماید. آتاماتونها می‌توانند هر یک از انواع آتاماتونهای ساختار ثابت و یا متغیر که قادر به همگرا شدن در بازی آتاماتونها با پاسخ یکسان باشند، انتخاب شوند. در صورتیکه تعداد عملها زیاد باشد، می‌توان از ساختارهای سلسله‌مراتبی نیز استفاده نمود.

مسئله نمونه ۱: محاسبه تابع تمایز برای جداسازی کلاس  $W_1$  از دو کلاس دیگر با توزیعهای نرمال با مشخصات زیر:

$$W_1: \Sigma = \begin{pmatrix} 4 & 0 \\ 0 & 6 \end{pmatrix}, m = [20, 10]^T \quad W_2: \Sigma = \begin{pmatrix} 2 & -0.5 \\ -0.5 & 2 \end{pmatrix}, m = [14, 2]^T \quad (6)$$

$$W_3: \Sigma = \begin{pmatrix} 2 & -0.5 \\ -0.5 & 2 \end{pmatrix}, m = [13, 10]^T$$

$\Sigma$  ماتریس کوواریانس توزیعها و  $m$  بردار میانگین توزیعها می‌باشد. جدول (۱) نتیجه ۱۰ بار اجرای الگوریتم (۱) را برای حل مسئله نمونه ۱ نشان می‌دهد.

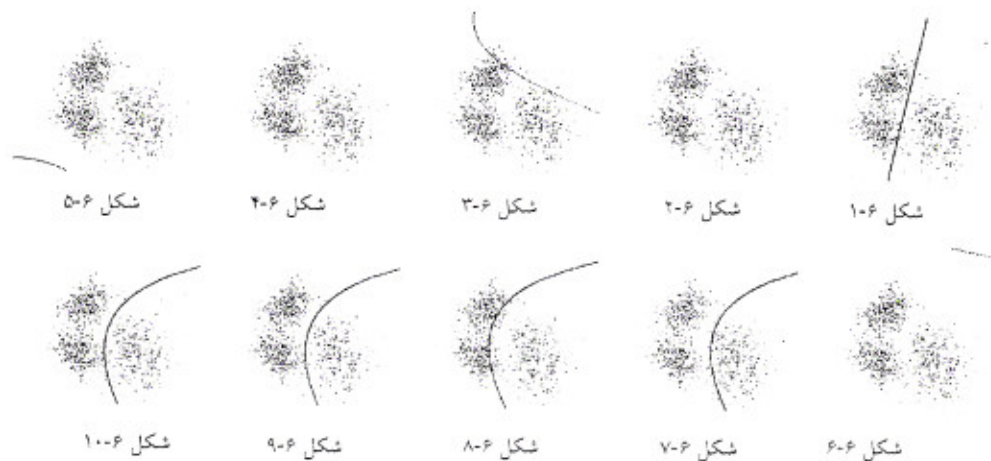
جدول ۱

شماره آزمایش	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰
دقت اولیه	۹۶/۳	۹۵/۸	۹۶/۱	۹۵/۱	۹۷/۱	۹۵/۷	۹۶/۵	۹۶/۵	۹۷/۱	۹۶/۵
دقت نهایی	۹۸/۵	۹۸/۳	۹۷/۸	۹۸/۱	۹۸/۱	۹۷/۲	۹۸/۱	۹۸/۷	۹۸/۱	۹۸/۵
تعداد مراحل	۱۵۹۱۵	۲۳۲۲۶	۱۲۱۸۴	۱۵۳۰۲	۲۳۲۳۸	۱۷۴۷۲	۳۰۲۰۷	۱۳۲۱۷	۱۲۰۹۹	۲۴۶۴۸

در پیاده‌سازی انجام شده، مقدار  $K = 5$  و  $r' = 3$  در نظر گرفته شده و برای یادگیری هر آتاماتون از الگوریتم  $SL_{R-T}$  با پارامتر  $W = 1000$  استفاده گردیده است. در قسمت یادگیری تابع تمایز درجه دوم با استفاده از تابع خطی اولیه ساخته می‌گردد. در جدول (۱) دقت اولیه، دقت تابع تمایز خطی اولیه حدس زده شده و دقت نهایی، دقت تابع تمایز درجه ۲ اصلاح شده می‌باشد. شکل (۱-۶) وضعیت خط اولیه تخمین زده شده بوسیله تابع فشر [۱] برای یک سری نمونه آموزشی با توزیع



مسئله نمونه ۱ را نشان می‌دهد. شکل (۲-۶) وضعیت اولیه منحنی درجه ۲ در ابتدای یادگیری و شکلهای دیگر، وضعیت خط یادگرفته شده را بعد از هر ۳۰۰۰ مرحله نشان می‌دهد.



#### ۴ - الگوریتم دسته‌بندی قطعه به قطعه تطبیقی (Adaptive Piecewise Classification)

الگوریتم ارائه شده در بخش ۳ قادر به بهبود تابع تمایز خطی به فرم تابع تمایز موردنظر می‌باشد. در این الگوریتم فرم عمومی تابع تمایز نهایی باید معلوم باشد. با استفاده از یک الگوریتم دسته‌بندی قطعه به قطعه و تقریب زدن منحنی پوش هر دسته الگو، می‌توان این عمل را نیز بطور خودکار انجام داد. الگوریتم (۲) اسکلت اصلی این روش را نشان می‌دهد.

Procedure Adaptive-Piecewise  
Estimate-Basic-Lines;  
Discard-Extra-Lines;  
Learn-Modified-Lines;  
End Learn-Piecewise;

الگوریتم ۲

در الگوریتم (۲)، رویه Estimate-Basic-Lines، مجموعه توابع خطی اولیه برای جداسازی تقریبی کلاس  $W_1$  از سایر کلاسهای موجود در حوزه مسئله را تعیین می‌کند. این توابع می‌توانند به تعداد زیاد و با دقت پایین باشند. رویه Discard-Extra-Lines خطوط غیرضروری را حذف می‌کند به این ترتیب که مجموعه خطی که بیشترین دقت جداسازی را از خود نشان می‌دهند، حفظ شده و سایر خطوط کنار گذاشته می‌شود. در قدم سوم، رویه Learn-Modified-Lines پارامترهای خطوط تمایز بدست آمده را اصلاح می‌کند. در این قسمت حتی می‌توان درجه خطوط تمایز قطعه به قطعه بدست آمده از دو قدم قبلی را نیز با استفاده از الگوریتم (۱) افزایش داد. در الگوریتم‌هایی که در ادامه بخش خواهند آمد، فرض شده است که کلاس الگو  $W_1$  توسط یک منحنی محدب از سایر کلاسها قابل جداسازی باشد. در ادامه بحث، مراحل مختلف الگوریتم فوق به تفصیل آمده است.

#### ۴-۱ - تعیین خطوط اولیه

در این قسمت الگوریتمی برای تقریب زدن خطوط اولیه تشکیل دهنده پوش کلاس الگوی  $W_1$  از سایر کلاسها ارائه می‌گردد. برای انجام این عمل، از نقاط کناره‌ای توزیع کلاس الگوی  $W_1$  استفاده می‌گردد. نقاط کناره‌ای، نقاطی درنظر گرفته می‌شوند که فاصله آنها از نقاط متعلق به سایر کلاسهای الگو از حد مشخصی کمتر باشد. منظور از نقاط همان بردار ویژگی متعلق به مجموعه آموزشی حوزه مسئله می‌باشد. الگوریتم (۳) مجموعه خطوط اولیه برای جداسازی کلاس  $W_1$  از سایر کلاسها را تعیین می‌نماید. در الگوریتم (۳) مقادارهای MBD (Max Between Distance) و MWD (Max Within Distance) در پارامتر الگوریتم می‌باشد. آرایه Samples نمونه‌های آموزشی موجود در مجموعه نمونه آموزشی و آرایه  $W_1$  نمونه‌های مربوط به کلاس  $W_1$  را نشان می‌دهد. آرایه G مجموعه قطعات نقاط تقریبی برای شناسایی لبه کلاس  $W_1$  را ذخیره می‌کند و هر عضو آرایه V قطعه خط تقریب زده شده برای جداسازی کلاس  $W_1$  با استفاده از اطلاعات موجود در عضو متناظر در آرایه G را ذخیره می‌کند. بطور خلاصه الگوریتم فوق بصورت زیر عمل می‌کند: ابتدا به ازای هر نمونه آموزشی موجود در کلاس  $W_1$ ، کلیه نمونه‌های با فاصله کمتر از مقدار MBD که متعلق به سایر کلاسها باشند را جدا می‌کند و در صورتیکه چنین نمونه‌هایی وجود داشته باشد، گروهی در آرایه G ساخته می‌شود که اعضای آن نمونه اولیه کلاس  $W_1$  و سایر نمونه‌های بدست آمده می‌باشد. در قدم بعد گروههای نزدیک به هم با یکدیگر ترکیب می‌شوند، به این ترتیب که اگر فاصله نمونه‌های متعلق به کلاس  $W_1$  از هر گروه از مقدار MWD کمتر باشد، اعضای دو گروه در قالب یک گروه ترکیب می‌گردند. به این ترتیب مجموعه نمونه‌های آموزشی موجود در حوزه مسئله به گروههای کوچکتر تقسیم می‌گردد. در قدم سوم خط جدا کننده کلاس  $W_1$  از سایر کلاسها با استفاده از اطلاعات محلی موجود در گروهها تعیین و در آرایه V ذخیره می‌گردد. خروجی الگوریتم، آرایه V می‌باشد که اعضای آن ضرایب خطوط تمایز اولیه می‌باشد.

# Procedure Learn-Basic-Lines

```

for i := 1 to Wi.Count do
    SampleAdded := False;
    for j := 1 to Samples.Count do
        if Samples[j].ClassID <> Samples[i].ClassID and
            Distance(Samples[j], Samples[i]) < MBD then
            if not SampleAdded then
                SampleAdded := True;
                Gi := Make-A-Group-For-Samples[i];
            end if;
            Gi.AddSample (Samples[j]);
        end if;
    end for {j};
end for {i};

for i := 1 to G.Count do
    for j := i+1 to G.Count do
        if Distance (Gi.Sample-Wi, Gj.Sample-Wj) < MWD then
            Gi := CombineGroups (Gi, Gj);
        end if;
    end for {j}
    Vi := Estimate-Line (Gi)
end for {i}
end Learn-Basic-Lines;

```

## الگوریتم ۳

در الگوریتم فوق با در نظر گرفتن پارامتری برای تعیین لبه، می توان تعداد خطوط اضافی را کم کرد به این ترتیب که تنها خطوطی در مجموعه  $V$  ذخیره گردند که دقت جداسازی آنها برای برای جداسازی نمونه های کلاس  $W_i$  از حد مشخص BP (Border Parameter) بیشتر باشد.

## ۲-۴ - کاهش تعداد خطوط

اگر تابع تمایز خطی با بردار ضرایب  $V$  به صورت

$$\text{if } v^T \underline{x} < 0, \text{ then } \underline{x} \in W_i \text{ else } \underline{x} \notin W_i \quad (V) \quad (7)$$

در نظر گرفته شود، در جداسازی قطعه به قطعه توسط آرایه خطوط  $V$  با  $m$  عضو  $(V_i)$  خواهیم داشت :

(۸)

تعدادی از خطوط در مجموعه خطوط بدست آمده توسط الگوریتم (۳) غیر ضروری می باشند و همچنین بدلیل دقیق بودن الگوریتم تقریب لبه، این مجموعه حاوی خطوط با دقت پایین نیز می باشد. در قدم بعدی، خطوط غیر ضروری و خطوطی که دارای دقت پایین می باشند، حذف می شوند و یا به عبارت دیگر بهترین زیر مجموعه  $n$  عضوی از مجموعه  $m$  عضوی که بهترین دقت جداسازی را بدست می دهد باید تعیین گردد. بهترین جواب با آزمایش کلیه زیرمجموعه های مجموعه  $V$  روی مجموعه نمونه های آموزشی حاصل می گردد که دارای پیچیدگی  $O(2^n)$  می باشد. الگوریتم (۴) زیرمجموعه ای خوب (نه الزاماً بهینه) را از بین کلیه زیرمجموعه ها، با پیچیدگی  $O(m^2)$  جدا می کند.

در الگوریتم (۴)، آرایه های  $Masks$  و  $Max-Masks$  دارای اعضای صفر یا یک می باشند. هر عضو از این دو آرایه، متناظر با عضوی از آرایه  $V$  می باشد. تنها اعضای از آرایه خطوط  $V$  در دستبندی الگوها شرکت می نمایند که عضو متناظر آن در آرایه  $Masks$  برابر با «یک» باشد. مقادیر اولیه این دو آرایه صفر فرض می شود. متغیر  $Precision$  دقت جداسازی در هر مرحله و متغیر  $Max-Precision$  حداکثر دقت جداسازی بدست آمده تا آن مرحله را در خود ذخیره می کنند. اعضای آرایه  $Max-Masks$  مجموعه خطوطی که بیشترین دقت جداسازی را تولید کرده اند را نشان می دهند. همانطور که در الگوریتم مشاهده می شود، الگوریتم دارای  $m$  (تعداد خطوط آرایه  $V$ ) قدم (حلقه ای که با متغیر  $i$  مشخص شده است) می باشد. در هر قدم فرض می شود که عضو متناظر آن قدم (عضو  $i$ ام)، در جداسازی شرکت می نماید و در نهایت با توجه به الگوریتم ارائه شده، زیرمجموعه ای از سایر خطوط که بیشترین دقت جداسازی را از خود نشان می دهند، تعیین می گردد. دقت جداسازی با توجه به آرایه  $Masks$  در هر مرحله توسط تابع  $DiscriminateByMasks$  تعیین می گردد. بهترین دقت بدست آمده در مراحل مختلف به عنوان نتیجه الگوریتم در نظر گرفته می شود. الگوریتم (۵) تابع  $DiscriminateByMasks$  می باشد.

پیچیدگی الگوریتم (۴) از مرتبه  $O(m^2)$  می باشد و در صورتیکه مجموعاً  $M$  بردار نمونه در مجموعه نمونه های آموزشی وجود داشته باشد، تعداد مقایسه ها و مرتبه  $O(m^2 M)$  خواهد بود. خروجی الگوریتم (۴) بردار  $Max-Masks$  می باشد که نشان می دهد چه خطوطی در تعیین لبه های کلاس الگوی  $W_i$  شرکت خواهند داشت.

```

Procedure Discard-Extra-Lines (v : List of Lines)
    Max-Precision := 0;
    for i := 1 to v.Count do
        for j := 1 to Masks.Count do
            Masks[j] := 0;
        end for {j};
        Masks[i] := 1;
        Precision := DiscriminateByMasks;
        for (j := 1 to Masks.Count) and (j < i) do
            Masks[j] := 1;
            if DiscriminateByMasks > Precision then
                Precision := DiscriminateByMasks
            else
                Masks[j] := 0;
            end if;
        end for {j}
        if Precision > Max-Precision then
            Max-Precision := Precision;
            Max-Masks := Masks;
        end if;
    end for {i}
End Discard-Extra-Lines
    
```

الگوریتم ۲

```

Function DiscriminateByMasks : Real;
    N-Correct := 0;
    for i := 1 to Samples.Count do
        V-Side := True;
        for (j := 1 to v.Count) and (Masks[j] = 1) do
            v-Side := v-Side and
                (DiscriminateByV(v[i], Sample) <= 0)
        end for {j}
        if (v-Side and (Samples[i] in Wi)) or
            (not v-Side and not (Samples[i] in Wi)) then
            inc (N-Correct);
        end for {i}
        Result :=  $\frac{N-Correct}{Samples.Count}$ 
    End DiscriminateByMasks;
    
```

الگوریتم ۵

### ۳-۴ - تصحیح خطوط

مجموعه خطوط بدست آمده از الگوریتمهای (۳) و (۴)، بدلیل دقیق نبودن الگوریتمهای ایجاد کننده آنها، خطوط غیر دقیقی هستند. در این بخش، الگوریتمی براساس آتاماتونهای یادگیر برای تصحیح خطوط بدست آمده ارائه می‌گردد. روش کلی به اینصورت می‌باشد که برای هر خط موجود در آرایه V که فیلد متناظر آن در آرایه Max-Masks برابر یک باشد، یک تیم از آتاماتونها مطابق الگوریتم (۱) تشکیل می‌شود. در هر مرحله، نمونه‌ای بطور تصادفی انتخاب و توسط تیمی آتاماتونهای شرکت کننده در بازی دستمندی می‌گردد. سپس آتاماتونهای شرکت کننده، براساس نتیجه دستمندی، استراتژی تصمیم‌گیری خود را برای مرحله بعدی بهنگام می‌نمایند. الگوریتمهای (۶) تا (۸) نحوه تصحیح خطوط توابع تمایز بدست آمده از الگوریتمهای (۳) تا (۵) را نشان می‌دهند.

```

Procedure Learn-Modified-Lines
    for (i := 1 to v.Count) and if (Max-Masks[i] = 1) then
        LA-List := Create LAs for modifying
            line v[i] like Algorithm (1);
        GCi.LA-List := LA-List;
    end for {i}
    Repeat
        Sample := Select-A-Random-Sample;
        DiscRes := DiscriminateByGC (Sample);
        for i := 1 to GC.Count do GCi.UpdateGame (DiscRes);
    until EndOfLearn;
End Learn-Modified-Lines;
    
```

الگوریتم ۶

```

Function DiscriminateByGC (Sample) : Boolean
    LA-Side := True;
    for i := 1 to GC.Count-1 do
        LA-Side := LA-Side and
            (Discriminate(GC.LA-List, Sample) <= 0);
    end for {i}
    Result := (LA-Side and (Sample in Wi)) or
        (not LA-Side and not (Sample in Wi));
    End DiscriminateByGC;
    
```

الگوریتم ۷

```

Function EndOfLearn : Boolean;
    Result := True if for All GC : GCi.EndOfGame = True;
End EndOfLearn;
    
```

الگوریتم ۸

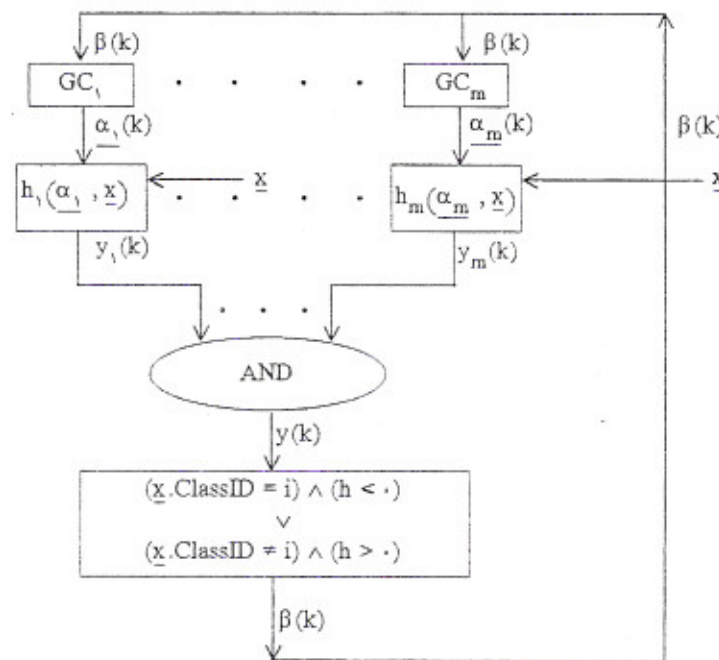


هر عضو  $GC_i$  از آرایه  $GC$  (Game Controller) تیم آتاماتونهای اصلاح کننده خط  $V_i$  را نگهداری می‌نماید. به ازای هر خط  $V_i$  تولید شده توسط الگوریتمهای (۳) تا (۵) و با توجه به روش تصحیح خط تمایز (اصلاح ضرایب تابع خطی، افزایش درجه تابع و ...)، یک تیم تصحیح کننده مشابه الگوریتم (۲) ایجاد و در  $GC_i$  ذخیره می‌گردد. سپس در قسمت یادگیری پارامترهای تابع تمایز، در هر مرحله، یک نمونه تصادفی از مجموعه نمونه‌های آموزشی انتخاب و بوسیله کلیه تیمهای شرکت کننده در تعیین پوش کلاس  $W_i$  دستبندی می‌گردد و با توجه به نتیجه دستبندی، آتاماتونهای شرکت کننده در تیمهای مختلف استراتژی تصمیم‌گیری خود را به‌هنگام می‌نمایند. هر مرحله یادگیری پارامترهای تابع تمایز در الگوریتم (۶) توسط حلقه Repeat...Until نشان داده شده است.

الگوریتم (۷) نحوه دستبندی یک بردار نمونه را نشان می‌دهد. در این الگوریتم تابع Discriminate در هر مرحله، با استفاده از پارامترهای یادگرفته شده برای تابع تمایز تا آن مرحله، مقدار تابع را برای بردار ویژگی نمونه ورودی تعیین می‌کند. خروجی الگوریتم (۷) به صورت دستبندی صحیح یا غلط می‌باشد.

الگوریتم (۸)، شرط خاتمه یادگیری را معین می‌کند. یادگیری زمانی خاتمه پیدا می‌کند که کلیه تیمهای آتاماتونهای شرکت کننده در تعیین خطوط تمایز، همگرا شده باشند یعنی آتاماتونهای شرکت کننده در هر تیم، عمل بهینه خود را پیدا کرده باشند.

الگوریتمهای (۶) تا (۸) شبکه‌ای از تیمهای آتاماتونهای یادگیر را بوجود می‌آورند. در هر مرحله، پاسخ محیط که همان نتیجه دستبندی می‌باشد، برای کلیه آتاماتونهای شرکت کننده در هر تیم و برای کلیه تیمهای شرکت کننده در شبکه یادگیری، ارسال می‌شود. شکل (۷) نمایی از این شبکه در لحظه  $k$  را نشان می‌دهد. تیمهای  $GC_1, \dots, GC_m$  پارامترهای هر یک از  $m$  خط تابع تمایز اصلاح شده را تعیین می‌نمایند. هر تیم، بردار عمل  $\alpha_i(k)$  را که بیانگر مقدارهای تقریب زده شده برای مجموعه پارامترهای خط  $V_i$  در مرحله  $k$  می‌باشد را انتخاب می‌نماید (در شکل، خط  $V_i$  بصورت تابع  $h_i(\alpha_i, x)$  نشان داده شده است). نتیجه دستبندی هر تیم بصورت  $y_i(k)$  برای بردار نمونه تصادفی  $x$  در هر مرحله محاسبه شده و نتیجه کلی بازی با اجرای عملگر AND روی نتایج تیمها بدست می‌آید. این نتیجه به عنوان پاسخ برای شبکه یادگیری در نظر گرفته شده و به همه تیمها ارسال می‌گردد.



شکل ۷

**مسئله نمونه ۲:** پیدا کردن توابع تمایز برای جداسازی کلاس  $W_p$  از بین چهار کلاس  $W_1, W_2, W_3$  و  $W_4$  با توزیعهای نرمال. مشخصات کلاسهای  $W_1, W_2, W_3$  و  $W_4$  مطابق مسئله نمونه ۱ و ماتریس کوواریانس و بردار میانگین توزیع  $W_p$  مطابق رابطه (۸) می‌باشد.

$$W_p: \Sigma = \begin{pmatrix} 2 & 0.5 \\ 0.5 & 2 \end{pmatrix}, \quad m = [20, 2]^T \quad (9)$$

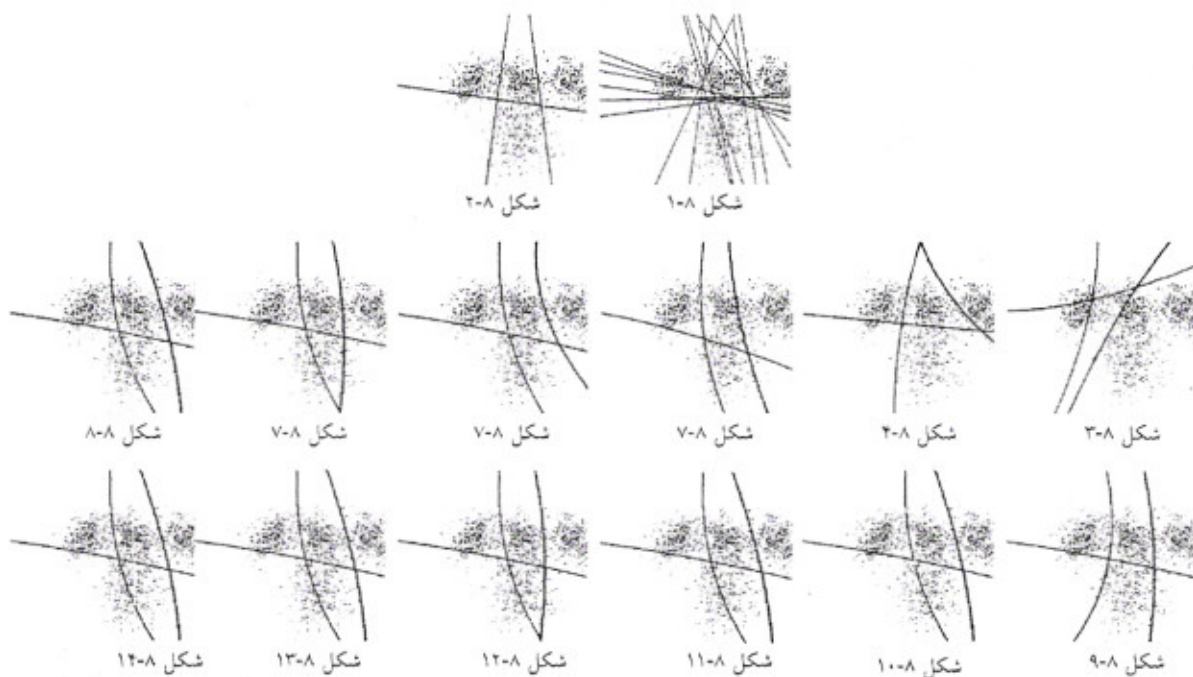
الگوریتم (۲) به ازای  $K = 100, T = 3, MBD = 1, MWD = 0.5, BP = 0.8$  و با استفاده از آتاماتونهای  $SLR-I$  با پارامتر یادگیری  $W = 2000$  برای حل مسئله نمونه ۲ پیاده‌سازی شد. جدول (۲) نتایج ۱۰ بار اجرای الگوریتم را با پارامترهای ذکر شده نشان می‌دهد. در جدول (۲) سطر مربوط به تعداد خطوط اولیه تعداد خطوط اولیه حدس زده شده در مرحله اول الگوریتم را نشان می‌دهد که با مقدار مشخص شده توسط «دقت اولیه» کلاس الگوی  $W_p$  را از سایر کلاسها جدا می‌کند. تعداد خطوط ثانویه، تعداد خطوط باقی مانده پس از انجام مرحله حذف خطوط غیر ضروری را نشان می‌دهد که با دقت مشخص شده توسط «دقت ثانویه» کلاس  $W_p$  را جدا می‌کنند. دقت نهایی، بیانگر دقت خطوط اصلاح شده توسط آتاماتونهای یادگیر می‌باشد.

شکل (۱-۸) تا (۱۲-۸) مراحل یادگیری توابع تمایز برای جداسازی کلاس  $W_p$  در مسئله نمونه ۲ را نشان می‌دهد. شکل (۱-۸) خطوط اولیه تخمین زده شده را نشان می‌دهد. شکل (۲-۸) خط باقی مانده پس از حذف خطوط غیر ضروری را نشان می‌دهد. شکلهای (۳-۸) تا (۱۴-۸) مراحل مختلف تصحیح توابع تمایز که توسط آتاماتونهای  $SLR-I$  انجام می‌گیرد را نشان می‌دهد. شکل (۳-۸) وضعیت اولیه خطوط در شروع یادگیری و شکلهای (۴-۸) تا (۱۴-۸) توابع تمایز اصلاح شده پس از

هر ۲۰۰۰۰ مرحله را نشان می دهد.

جدول ۲

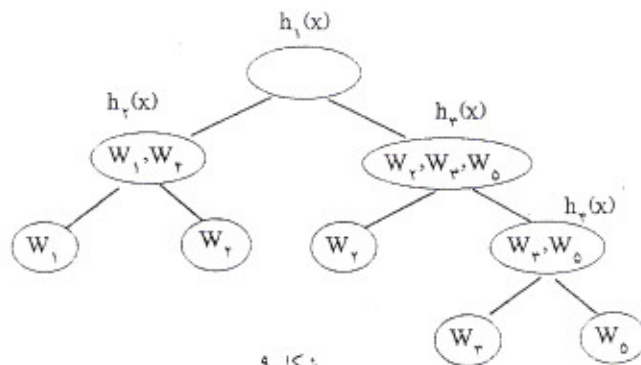
شماره آزمایش	۱	۲	۳	۴	۵	۶	۷	۸	۹	۱۰
تعداد خطوط اولیه	۱۷	۱۶	۱۷	۱۷	۲۰	۲۰	۱۵	۱۵	۱۷	۱۷
دقت اولیه	۸۸/۷	۹۰	۸۷/۳	۹۰/۷	۹۰/۲	۸۹/۲	۹۱/۴	۸۷/۹	۸۶/۱	۹۰/۳
تعداد خطوط ثانویه	۴	۳	۴	۴	۵	۳	۳	۳	۴	۳
دقت ثانویه	۹۵/۸	۹۲/۶	۹۵/۵	۹۵/۹	۹۵/۶	۹۶/۳	۹۲/۷	۹۵/۲	۹۵/۵	۹۵/۲
دقت نهایی	۹۵/۹	۹۶/۱	۹۵/۹	۹۶/۱	۹۵/۸	۹۶/۵	۹۵/۵	۹۶/۲	۹۵/۲	۹۵/۳
تعداد مراحل	۲۹۵۳۷۸	۱۲۷۷۹۱	۳۹۳۹۶۲	۳۲۲۵۳۶	۲۷۲۵۷۱	۲۱۲۲۲۹	۲۸۶۶۹۱	۲۳۱۰۰۷	۲۲۳۹۲۵	۳۹۱۳۱۷



## ۵ - دسته بندی درختی

الگوریتم ارائه شده در بخش (۲) یک کلاس الگو را از سایر کلاسهای الگو جدا می نماید. در صورتیکه  $n$  کلاس  $W_1, \dots, W_n$  در حوزه مسئله مورد نظر وجود داشته باشد، با اجرای الگوریتم (۲) برای کلاسهای مختلف مسئله، می توان هر کلاس را از سایر کلاسها جدا نمود. با استفاده از دسته بندی درختی، به شکل موثرتری می توان عمل دسته بندی را انجام داد. روش کار بدین صورت است که ابتدا توسط یک الگوریتم قطعه بندی (Clustering Algorithm)، از طریق ترکیب برخی از کلاسهای مسئله با یکدیگر و ساختن کلاسهای جدید، قطعات (Cluster) بزرگتری که قابلیت جداسازی بیشتری دارند، ساخته می شود و در مراحل بعدی، کلاسهای موجود در هر قطعه از یکدیگر جدا می گردد. به این ترتیب یک درخت قطعه بندی برای جداسازی کلاسهای مسئله مورد نظر تشکیل می شود بطوریکه در سطوح بالایی، قطعات بزرگتر و در سطوح پایینی، قطعات کوچکتر و در برگها، کلاسهای مسئله جدا می شوند. شکل (۹) یک مسئله با ۵ کلاس که بصورت درختی قطعه بندی شده است را نشان می دهد. قاعده تصمیم گیری هر گره، دو قطعه سمت چپ و راست هر گره را از یکدیگر جدا می نماید.

تقسیم بندی کلاسها در هر گره به گونه ای انجام می گیرد که دقت جداسازی در آن گره بیشترین مقدار ممکن باشد. به این ترتیب، کلاسهای مشابه، در یک قطعه قرار می گیرند. با استفاده از الگوریتم دسته بندی درختی، دسته بندی های واضح تر (کلاسهای که به راحتی از یکدیگر جدا می گردند) در سطوح بالاتر و دسته بندی های غامض تر (دسته بندی کلاسهایی که بطور ساختاری به یکدیگر نزدیکتر می باشند) در سطوح پایتر انجام می گیرد. الگوریتم (۹)، الگوریتم دسته بندی درختی را نشان می دهد. در الگوریتم (۹)، رویه Make-Class-Tree درخت قطعات کلاسهای مسئله را بوجود می آورد. رویه Estimate-Node-Lines خطوط تمایز اولیه مربوط به جداسازی قطعه کلاسها در هر گره از درخت را ایجاد می کند. قطعات قرار گرفته در دو طرف هر گره می توانند توسط یک خط تمایز و یا دسته ای از خطوط تمایز (مانند الگوریتم (۲) از یکدیگر جدا شوند. رویه Modify-Node-Lines خطوط اولیه تقریب زده شده در هر گره را تصحیح می نماید.



شکل ۹

Procedure Learn-By-Tree  
 Make-Class-Tree;  
 Estimate-Node-Lines;  
 Modify-Node-Lines;  
 End Learn-By-Tree;

الگوریتم ۹

## ۱-۵- ایجاد درخت دسته‌بندی

فرض کنید  $n$  کلاس  $W_1, \dots, W_n$  در مسئله وجود داشته باشد. در صورتیکه بخواهیم این کلاسها را به دو گروه با  $p$  و  $n-p$  کلاس تقسیم کنیم. مجموعاً  $\binom{n}{p}$  حالت را می‌توان در نظر گرفت. در صورتیکه بخواهیم به هر شکل ممکن مجموعه کلاسها را به دو گروه تقسیم نماییم بطوریکه در هر گروه حداقل یک کلاس وجود داشته باشد  $(1 < p < n)$  تعداد حالتها ممکن دستمندی برابر با  $\sum_{p=1}^{n-1} \binom{n}{p} = 2^n - 1$  می‌باشد. بنابراین پیچیدگی الگوریتم محاسبه کننده بهترین قطعه‌بندی (تقسیم کلاسها به دو قطعه بطوریکه بهترین دقت جداسازی بوجود آید) در هر گره از مرتبه  $O(2^n)$  می‌باشد. الگوریتم (۱۰)، الگوریتمی است که در هر گره با مرتبه  $O(n^2)$  اجرا می‌شود و کلاسهای موجود در هر گره را به دو دسته که بطور خطی از یکدیگر جدایی پذیر باشند، تقسیم بندی می‌نماید.

Procedure Make-Class-Tree (ClassTree : TClassTree; ClassSet : TClassLabelSet);

if ClassSet.Count = 1 then Exit;

ClassTree.LeftSet = [];

ClassTree.Precision := 0;

Repeat

NextIndex := -1;

for i := 1 to ClassSet.Count do

LeftSet := ClassTree.LeftSet + ClassSet.Item[i];

V := Estimate-Line (LeftSet);

Precision := DiscriminateSamples (V);

if Precision > ClassTree.Precision then

NextIndex := i;

ClassTree.Precision := Precision;

end if

end for

if NextIndex > 0 then

ClassTree.LeftSet := ClassTree.LeftSet + ClassSet.Item[NextIndex];

ClassSet := ClassSet - ClassSet.Item[NextIndex];

end if

Until NextIndex < 0;

Make-Class-Tree (ClassTree.Left, ClassTree.LeftSet);

Make-Class-Tree (ClassTree.Right, ClassSet);

End Make-Class-Tree;

الگوریتم ۱۰

الگوریتم (۱۰) یک الگوریتم بازگشتی است و دارای دو ورودی می‌باشد. در پارامتر ClassTree، درخت تجزیه کلاسها ساخته و ذخیره می‌گردد. مجموعه ClassSet، مجموعه کلاسهایی که باید تقسیم‌بندی شوند، می‌باشد. اگر تعداد کلاسها  $n$  باشد، حلقه داخلی الگوریتم حداکثر  $\frac{n(n+1)}{2}$  بار اجرا می‌گردد. توسط این الگوریتم مجموعه کلاسها به دو گروه سمت چپ و سمت راست تقسیم می‌گردند. ابتدا کلاسی که به بهترین شکل بوسیله یک تابع خطی قابل جداسازی باشد، از بقیه کلاسها جدا شده و به عنوان عضوی از گروه کلاسهای سمت چپ در نظر گرفته می‌شود. در هر بار اجرای بعدی تا زمانی که با برداشتن کلاسی از کلاسهای باقی‌مانده و



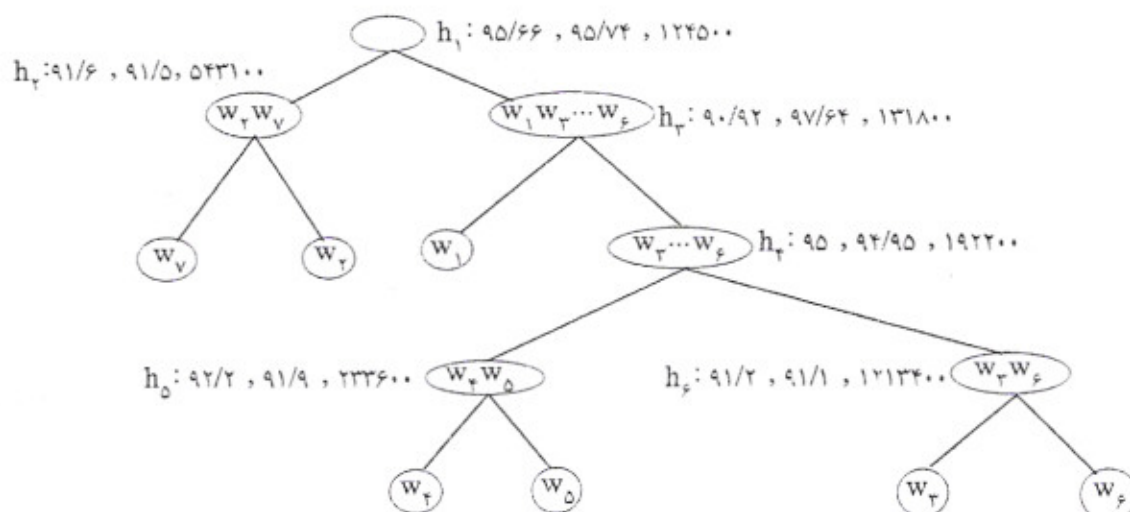
اضافه نمودن آن به گروه کلاسه‌های سمت چپ دقت بالاتری بدست بیاید. الگوریتم ادامه پیدا می‌کند. در غیر اینصورت کلاسه‌های جدا شده به عنوان کلاسه‌های گروه سمت چپ و کلاسه‌های باقی مانده به عنوان کلاسه‌های گروه سمت راست در نظر گرفته می‌شوند. الگوریتم بطور بازگشتی برای گروه سمت چپ و سمت راست اجرا می‌گردد. شرط خاتمه الگوریتم زمانی است که در هر گره انتهایی فقط یک کلاس وجود داشته باشد.

مسئله نمونه ۳: بدست آوردن توابع تمایز برای جداسازی ۷ کلاس  $w_1 \dots w_7$  با توزیع‌های نرمال. توزیع کلاسه‌های  $w_1 \dots w_7$  مطابق مسئله نمونه ۲ می‌باشد. ماتریس کوواریانس و بردار میانگین کلاسه‌های  $w_1 \dots w_7$  در زیر آمده است.

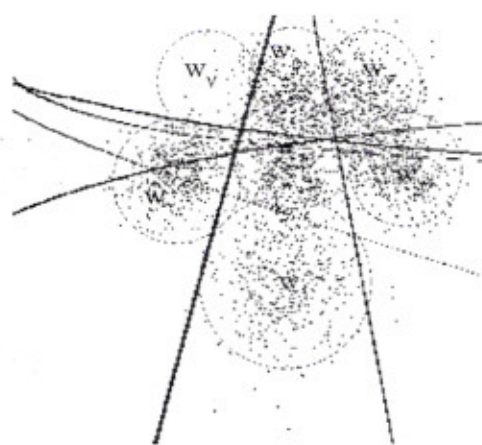
$$W_0: \Sigma = \begin{pmatrix} 2 & 0.5 \\ 0.5 & 2 \end{pmatrix}, m = [20, 0]^T \quad W_5: \Sigma = \begin{pmatrix} 3 & 0.5 \\ 0.5 & 3 \end{pmatrix}, m = [24, 0]^T$$

$$W_7: \Sigma = \begin{pmatrix} 3 & 0.5 \\ 0.5 & 3 \end{pmatrix}, m = [160, 0]^T$$

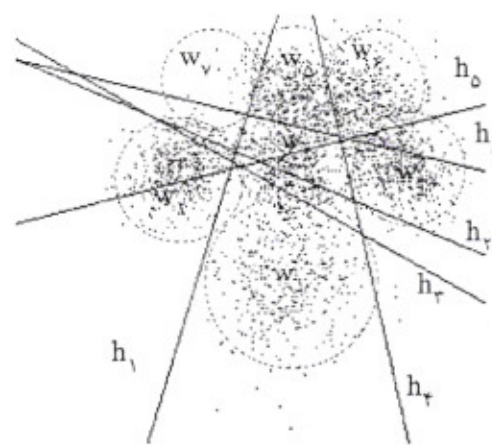
شکل (۱۰) نتیجه یک بار اجرای الگوریتم (۹) برای حل مسئله نمونه ۳ با پارامترهای  $K=10$  و  $r'=3$  و با در نظر گرفتن یک خط تمایز در هر گره و تصحیح آن با استفاده از آتامنوهای  $SLR-1$  با پارامتر یادگیری  $W=2000$  را نشان می‌دهد. کنار هر گره میانی، به ترتیب از چپ به راست، دقت تابع تمایز اولیه حدس زده شده، دقت تابع تمایز اصلاح شده و تعداد مراحل مورد نیاز برای اصلاح تابع، نشان داده شده است. دقت جداسازی کلیه کلاسه‌ها توسط خطوط اولیه حدس زده شده،  $79/91\%$  بود که پس از اصلاح خطوط، این دقت به  $84/54\%$  افزایش یافت. شکل (۱۱-۱) وضعیت اولیه خطوط و شکل (۱۱-۲) وضعیت نهایی خطوط تمایز را نشان می‌دهد.



شکل ۱۰



شکل ۱۱-۲



شکل ۱۱-۱

## ۶- جداسازی آواهای فارسی

در این بخش، کاربرد روش‌های ارائه شده در بخش‌های قبلی برای جداسازی آواهای گفتاری در زبان فارسی، مورد بررسی قرار می‌گیرند. به دلیل ویژگی‌های خاصی که در سیگنال صحبت وجود دارد، پردازش سیگنال صحبت و تشخیص صدا یکی از مسائل بسیار پیچیده محسوب می‌گردد. وجود این ویژگی‌ها باعث بوجود آمدن تکنیک‌های ویژه‌ای در تشخیص سیگنال صحبت شده است [۱۷] [۱۸]. درک گفتار فرآیندی بسیار پیچیده محسوب می‌گردد. بطوریکه علاوه بر تشخیص جنبه‌های سیگنال صدا (مثلاً تشخیص آواها) تکنیک‌های مختلفی از هوش ماشین و

زبان طبیعی نیز باید مورد استفاده قرار گیرد. روشهای تشخیص سیگنال صحبت به دو دسته کلی تقسیم می‌شوند که عبارتند از: روشهای پارامتریک و روشهای غیر پارامتریک. در روشهای غیر پارامتریک، معمولاً نتیجه انجام یک تبدیل روی سیگنال صحبت، جهت آنالیز مورد استفاده قرار می‌گیرد. تبدیل فوری یکی از این نوع تبدیلات می‌باشد. در روشهای پارامتریک، سیگنال صحبت توسط یک مدل پارامتریک مدل می‌شود. پارامترهای مدل مورد نظر محاسبه گردیده و نتیجه آن جهت آنالیز گفتار بکار برده می‌شود. در این فصل از روشی پارامتریک و از آنالیز MFCC استفاده گردیده است.

مجرای گفتار توسط چندین لوله صوتی باز و بسته که با طولها و قطرهای مختلف به یکدیگر متصل شده‌اند، می‌تواند مدل شود. لوله‌های صوتی به هم متصل شده باید بتوانند تخمینی از شکل مجرای گفتار انسان در هنگام تلفظ آوای مورد نظر را حاصل نمایند. تعداد لوله‌های در نظر گرفته شده در مدل، باید بقدری باشد که بتواند دقت مورد نظر را ایجاد کند و با توجه به کاربرد و دقت مورد نظر این تعداد می‌تواند متغیر باشد. در مدل استفاده شده در این بخش، از ۱۲ لوله صوتی برای مدل کردن مجرای گفتار استفاده شده است.

آزمایشهای انجام شده در این بخش برای جداسازی ۱۵ آوای مختلف /z/, /v/, /u/, /sh/, /s/, /r/, /o/, /l/, /kh/, /i/, /f/, /e/, /aa/, /a/ و /zh/ طراحی و پیاده‌سازی شده‌اند. هر آوای بصورت کلاسی مشکل از مجموعه‌ای از نمونه‌های آموزشی و تست و بصورت بردارهای ویژگی در فضای  $R^{12}$  در نظر گرفته شده و مجموعه کلیه بردارهای ویژگی استخراج شده از نمونه‌های صدا، پایگاه داده‌ای مسئله را تشکیل می‌دهد.

در مجموع از ۱۵ فرد مختلف با سن و جنس‌های مختلف و با درجه تحصیلات مختلف خولسته شد که هر یک از آوای فوق را بصورت عادی و به مدت یک ثانیه تلفظ نمایند و صدای هر فرد بصورت یک فایل wave با فرمت ۲۲ kHz/s ۱۶ بیتی و بصورت مونو با استفاده از یک میکروفن سلفی ۶۰۰ اهمی از طریق ورودی Mic کارت صوتی در رایانه ذخیره گردید. در قدم بعد، آوای مختلف موجود در صدای ضبط شده افراد مختلف بصورت دستی جدا گردید و مجموعه شکل موج مربوط به هر آوای بصورت یک فایل wave ذخیره گردید. در این قدم، مدت نیم ثانیه از آوای تلفظ شده توسط هر فرد جدا گردید، این نیم ثانیه از قسمتهای وسطی هر آوای تلفظ شده جدا گردید.

در مرحله بعد، مجموعه بردارهای ویژگی هر آوا با انجام آنالیز MFCC روی فایل‌های بدست آمده از بخش، قبل استخراج گردید. هر بردار ویژگی بصورت ضرایب آنالیز MFCC با ۱۲ مقدار ویژگی در نظر گرفته شد. ضرایب MFCC روی پنجره‌های ۳۰ میلی‌ثانیه‌ای از فایل wave مربوط به هر آوا محاسبه گردید و بین هر دو پنجره متوالی، روی هم افتادگی ۱۰ میلی‌ثانیه در نظر گرفته شد. به این ترتیب از هر آوای تلفظ شده توسط هر فرد تعداد ۲۵  $\frac{0.05 \text{ ms}}{20 \text{ ms}} = \frac{50 \text{ ms}}{20 \text{ ms}} = 2.5$  نمونه بردار ویژگی و مجموعاً برای هر آوا حدود ۴۰۰ بردار ویژگی استخراج گردید. در پایگاه اطلاعاتی، بردارهای ویژگی مربوط به آوای هر فرد به دنبال یکدیگر و بردارهای افراد مختلف پشت سر یکدیگر ذخیره گردید. مجموعه بردارهای ویژگی هر آوا به دو مجموعه آموزشی و تست تقسیم گردید. ۳۰۰ بردار نمونه ویژگی اول به عنوان مجموعه نمونه آموزشی هر آوا و بقیه بردارها (حدود ۱۰۰ بردار) به عنوان مجموعه نمونه‌های تست در نظر گرفته شد. به این ترتیب عمل یادگیری و تست روی صداهای مربوط به افراد مختلف انجام می‌گیرد. لیست کامل نمونه‌های آزمایش شده را می‌توانید در مرجع [۱۹] ببینید.

## ۱-۶ - دسته‌بندی دودویی

در این بخش برای هر دو کلاس موجود در حوزه مسئله یک تابع تمایز در نظر گرفته می‌شود. به این ترتیب اگر  $M$  کلاس الگو در حوزه مسئله وجود داشته باشد، برای جداسازی هر کلاس از سایر کلاسها،  $M-1$  تابع تمایز در نظر گرفته می‌شود و مجموعاً تعداد  $\frac{M(M-1)}{2}$  تابع تمایز تخمین زده می‌شود.

اگر  $f_{ij}(x) = \begin{cases} \geq 0 & \text{if } x \in w_i \\ < 0 & \text{if } x \in w_j \end{cases}, i = 1 \dots M, j = 1 \dots M-1$ ، برای دسته‌بندی

یک بردار الگوی ورودی، این بردار توسط کلیه توابع تمایز دسته‌بندی می‌شود. به این ترتیب به بردار ورودی تعداد  $\frac{M(M-1)}{2}$  برجسب نسبت داده می‌شود که بیانگر تعلق داشتن بردار ورودی به یک کلاس الگو توسط هر تابع تمایز از دید هر یک از توابع تمایز می‌باشد. در حالت ایده‌آل، بردار ورودی  $x$  به کلاس  $w_i$  تعلق دارد اگر  $f_{ij}(x) \geq 0, j = 1 \dots M-1$ ، به دلیل اینکه دقت هر تابع تمایز ۱۰۰٪ نمی‌باشد، باید ملاکی برای مقایسه نتیجه دسته‌بندی توابع تمایز تعریف گردد. این ملاک می‌تواند تعداد نسبت دهی الگو به هر کلاس در نظر گرفته شود، بدین معنی که بردار  $x$  که توسط بیشترین تعداد توابع تمایز به کلاس  $w_i$  نسبت داده شود، به عنوان عضوی از کلاس  $w_i$  در نظر گرفته می‌شود.

برای حل مسئله دسته‌بندی آوای صوتی، از آنجا که ۱۵ آوای مختلف مورد بررسی قرار می‌گیرند، تعداد  $\frac{15 \times 15}{2} = 105$  تابع تمایز با استفاده از تابع فیشر [۱] تخمین زده شد. جدول (۴) درصد دسته‌بندی مجموعه نمونه‌های تست توسط توابع تمایز بدست آمده را نشان می‌دهد.

هر خانه از جدول (۴)، درصد دسته‌بندی مجموعه نمونه مربوط به هر کلاس مشخص شده توسط سطر مربوط به آن خانه، در کلاس مشخص شده با برجسب ستون آن خانه را نشان می‌دهد. به این ترتیب درصد‌های مشخص شده توسط خانه‌های مربوط به قطر اصلی، درصد دسته‌بندی‌های صحیح مربوط به هر کلاس آوا می‌باشد. در حالت ایده‌آل، عناصر قطر اصلی باید ۱۰۰ و عناصر سایر خانه‌ها باید صفر باشد.

توابع تمایز خطی اولیه بدست آمده، با استفاده از روش پیشنهاد شده در بخش (۳)، بوسیله بازی آتاماتونهای یادگیر با پاسخ یکسان اصلاح گردید. برای اصلاح توابع  $K = 25$  و  $\lambda = 1$  در نظر گرفته شد. به این ترتیب برای هر آتاماتون ۳ عمل تعریف می‌گردد. برای اصلاح از آتاماتونهای  $SLR-I$  با پارامتر یادگیری  $W = 1000$

استفاده گردید. برای تصحیح هر تابع تمایز، ۱۰۰ مرتبه آزمایش انجام گردید و تابع تصحیح شده با بیشترین دقت جداسازی، به عنوان تابع تمایز بهینه در نظر گرفته شد. اصلاح توابع تمایز برای جداسازی کلاسهای مختلف مسئله تا دو قدم دیگر نیز ادامه داده شد. در قدم اول، توابع تمایزی که دقت کمتر از ۹۰٪ تولید می‌کردند، با پل ۱ مترهای  $K = 50$  و  $\lambda = 1$  باریدگر مورد اصلاح قرار گرفتند و در قدم بعد توابع خطی اصلاح شده‌ای که دقت کمتر از ۸۰٪ را از خود نشان می‌دادند، به درجه ۲ ارتقا داده شدند و بوسیله بازی آتاماتونهای یادگیر  $SLR-I$  با پارامتر یادگیری  $W = 1000$  ضرایبشان تعیین گردید. به این ترتیب با توجه به ۴ مرحله انجام شده در

تعیین توابع تمایز (مرحله اول حدس اولیه با استفاده از تابع فیشر، مرحله دوم الگوریتم ۲ و مراحل سوم و چهارم، نتایج اصلاحهای ذکر شده) برای جداسازی دو کلاس  $w_1$  و  $w_2$  حداکثر ۴ تابع تمایز بدست آمد و تابعی که حداکثر دقت جداسازی را برای جداسازی دو کلاس  $w_1$  و  $w_2$  از خود نشان داد، به عنوان تابع تمایز نهایی انتخاب گردید. جدول (۵) درصد دسته‌بندی نمونه‌های آوای مختلف توسط توابع تعیین شده را نشان می‌دهد.



جدول ۴

	/a/	/aa/	/e/	/f/	/i/	/kh/	/l/	/o/	/r/	/s/	/sh/	/u/	/v/	/z/	/zh/
/a/	۶۶	۰	۱	۱	۰	۰	۱	۱۱	۱۵	۰	۱	۱	۲	۱	۰
/aa/	۰	۸۸	۰	۰	۰	۳	۰	۰	۰	۰	۰	۰	۰	۰	۹
/e/	۴	۰	۲۵	۲	۹	۰	۳	۳	۵۲	۰	۰	۱	۰	۰	۰
/f/	۰	۰	۰	۸۵	۰	۰	۰	۰	۱	۰	۰	۰	۹	۵	۰
/i/	۱	۰	۵	۰	۵۰	۰	۲۳	۰	۱۱	۰	۱	۰	۸	۰	۰
/kh/	۰	۱۸	۰	۰	۰	۶۹	۰	۰	۰	۰	۲	۰	۱	۰	۸
/l/	۵	۰	۰	۰	۲۰	۰	۳۳	۴	۱۲	۰	۰	۱۸	۸	۰	۰
/o/	۳	۰	۲۰	۰	۰	۰	۱	۷۰	۲	۰	۰	۵	۰	۰	۰
/r/	۸	۰	۱۶	۲	۹	۰	۱۵	۸	۱۵	۰	۰	۱۶	۷	۲	۰
/s/	۰	۰	۰	۱۰	۱	۰	۰	۳	۱	۷۶	۰	۰	۶	۳	۰
/sh/	۰	۳	۰	۰	۰	۲۰	۰	۰	۰	۰	۷۵	۰	۰	۰	۲
/u/	۰	۰	۴	۰	۱۲	۰	۱۷	۲	۱۰	۰	۰	۲۹	۵	۰	۰
/v/	۱	۰	۶	۱۶	۹	۰	۱۷	۹	۲	۰	۰	۳	۱۹	۱۸	۰
/z/	۰	۰	۲۵	۶	۰	۰	۱	۵	۱۱	۰	۰	۱	۲۶	۲۵	۳
/zh/	۰	۲۴	۶	۰	۰	۱۱	۰	۴	۰	۰	۱۵	۲	۰	۰	۳۸

جدول ۵

	/a/	/aa/	/e/	/f/	/i/	/kh/	/l/	/o/	/r/	/s/	/sh/	/u/	/v/	/z/	/zh/
/a/	۷۳	۰	۱	۱	۰	۰	۰	۰	۱۸	۰	۱	۲	۳	۰	۰
/aa/	۰	۹۲	۰	۰	۰	۵	۰	۰	۰	۰	۰	۰	۰	۰	۳
/e/	۳	۰	۴۱	۵	۴	۰	۱	۰	۲۲	۰	۰	۳	۰	۰	۰
/f/	۰	۰	۰	۸۹	۰	۰	۰	۰	۵	۰	۰	۰	۰	۵	۰
/i/	۰	۰	۹	۰	۳۳	۰	۱۶	۰	۱۸	۰	۱	۲	۹	۲	۰
/kh/	۰	۳	۰	۰	۰	۸۵	۰	۰	۰	۰	۵	۰	۰	۰	۷
/l/	۰	۰	۰	۰	۱۱	۰	۲۶	۰	۲۸	۰	۰	۲۵	۸	۳	۰
/o/	۰	۰	۱۵	۰	۰	۰	۰	۷۷	۳	۰	۰	۵	۰	۰	۰
/r/	۵	۰	۹	۳	۳	۰	۵	۱۲	۲۵	۰	۰	۸	۹	۲	۰
/s/	۰	۰	۰	۱۰	۰	۰	۰	۰	۹۰	۰	۰	۰	۰	۰	۰
/sh/	۰	۱	۰	۰	۰	۲۱	۰	۰	۰	۰	۷۷	۰	۰	۰	۱
/u/	۰	۰	۲	۰	۶	۰	۳	۰	۲۶	۰	۰	۵۹	۵	۰	۰
/v/	۰	۰	۴	۲۱	۴	۰	۵	۰	۲۰	۱	۰	۷	۲۶	۱۱	۰
/z/	۰	۰	۷	۸	۵	۰	۱	۰	۵	۰	۰	۶	۱۲	۵۷	۰
/zh/	۰	۲	۴	۰	۱	۳	۰	۲	۰	۰	۱۳	۴	۰	۰	۷۲

از آنجا که اعضای قطر اصلی، بیانگر درصد دسته‌بندی صحیح برای هر کلاس آوا می‌باشد و از آنجا که تعداد نمونه‌های آموزشی و تست آواهای مختلف تقریباً یکسان می‌باشد، میانگین قطر اصلی هر جدول را می‌توان به عنوان درصد دقت دسته‌بندی کل سیستم دسته‌بندی در نظر گرفت. در این صورت، دقت جدول (۵) برابر ۶۳/۴۷٪ می‌باشد.

**توضیح ۱:** در دسته‌بندی دودویی، برای مقایسه نتایج دسته‌بندی توابع تمایز مختلف برای نسبت‌دهی بردار نمونه ورودی به یک کلاس، یک سیستم استنتاج باید تعریف گردد. سیستم استنتاجی که در این فصل استفاده گردید، یک سیستم ایستا می‌باشد (به عنوان مثال، نسبت‌دهی بردار نمونه به کلاسی که توسط بیشترین تعداد توابع تمایز بردار نمونه به آن نسبت داده شده باشد). شاید بتوان با استفاده از یک سیستم یادگیر پویا در این قسمت، نتایج دقیقتری بدست آورد.

**توضیح ۲:** توسط دسته‌بندی دودویی می‌توان ملاکی برای تشابه ساختاری دو آوا (کلاس الگو) تعریف نمود. در صورتی که تشابه یک کلاس با خودش را ۱ فرض نماییم و اگر  $t_{ij}$  عضو  $i$ ام جدول دسته‌بندی باشد، مقدار کمی تشابه ( $s_{ij}$ ) را می‌توان بصورت  $s_{ij} = \frac{t_{ij} + t_{ji}}{t_{ii} + 1}$  تعریف نمود.

**توضیح ۳:** برای شناسایی کلمه از طریق آواشناسی، می‌توان سیگنال ورودی را به فرم‌هایی تقسیم نموده و آواها را با در نظر گرفتن محدودیت‌هایی که به حوزه کاربرد مورد نظر مربوط می‌شود، بصورت نویزی مشخص نمود. به این ترتیب یک رشته آوای نویزی بدست می‌آید. با داشتن یک سیستم استنتاج برای شناسایی رشته‌های نویزی، می‌توان کلمه مورد نظر را شناسایی نمود. مرجع [۲۰] یک الگوریتم شناسایی رشته کلمات نویزی با استفاده از آتوماتونهای یادگیر ساختار ثابت را ارائه نموده است.



## ۷ - خلاصه

در این مقاله روشهای ترکیبی مبتنی بر اتوماتونهای یادگیر و روشهای کلاسیک، برای دستبندی الگوها ارائه شد. در اولین روش، ابتدا یک تابع خطی (غیر دقیق) با استفاده از روشهای کلاسیک برای جداسازی یک کلاس از سایر کلاسها تخمین زده میشود و سپس پارامترهای تابع خطی بدست آمده بوسیله بازی اتوماتونهای یادگیر تصحیح میگردد. در قسمت بعد، روشی برای افزایش درجه تابع تخمین زده شده به منظور افزایش دقت تابع تمایز، ارائه گردید. در بخش چهارم الگوریتمی به نام «الگوریتم دستبندی قطعه به قطعه تطبیقی» پیشنهاد گردید. در این روش، پوش توزیع هر کلاس الگو بوسیله مجموعه‌ای از خطوط توسط روشهای کلاسیک تقریب زده میشود. بعد از حذف خطوط غیر ضروری (خطوطی که با حذف آنها، دقت دستبندی تغییر نمیکند یا حتی افزایش می‌یابد) خطوط باقی مانده بوسیله اتوماتونهای یادگیر اصلاح می‌گردند. در بخش پنجم، الگوریتم «دستبندی درختی» پیشنهاد گردیده است. در این الگوریتم، ابتدا با توجه به میزان جدایی‌پذیری کلاسهای الگوی موجود در حوزه کاربرد مورد نظر، یک ساختار درختی از کلاسهای ایجاد میگردد و سپس توابع تمایز جداکننده در هر گره از ساختار درختی توسط روشهایی که پیشتر به آنها اشاره گردید، تخمین زده میشود. کاربرد روشهای پیشنهادی در مسئله جداسازی آوارهای گفتاری فارسی نیز بررسی شد.

## ۸ - منابع

- [1] Fukunaga, An Introduction to Statistical Pattern Recognition, New York, Academic Press, 1972.
- [2] Pierre A. Devijver and Joseph Kittler, Pattern Recognition Theory and Applications, NATO ASI Series, Series F: Vol. 30, 1986.
- [3] Andrew G. Barto and P. Anandan, "Pattern Recognition Stochastic Learning Automata", IEEE Trans. Sys. Man and Cybern., Vol. SMC-15, No. 3, May/January 1985.
- [4] Mandayam A. L. Thatachar and P. S. Sastry, "Learning Optimal Discriminant Functions Through a Cooperative Game of Automata", IEEE Trans. Sys. Man and Cybern., Vol. SMC - 17, No. 1, January/February 1987.
- [5] M. A. L. Thatachar and V. V. Phananskar, "A feedforward Network of Learning Automata for Pattern Recognition", in Proc. Int. Joint. Conf. Neural Networks, Singapore, Nov. 1991.
- [6] M. A. L. Thatachar and V. V. Phananskar, "Convergence of Teams and Hierarchies of Learning Automata in Connectionist Systems", IEEE Trans. Sys. Man and Cybern., Vol. 25, No. 11, Nov. 1995.
- [7] V. V. Phananskar and M. A. L. Thatachar, "Global Convergence of Teams of Learning Automata", in Symp. Intell. Syst., Bangalore, Dec. 1991.
- [8] M. A. L. Thatachar and V. V. Phananskar, "Learning Global Maximum with Parameterized Learning Automata", IEEE Trans. Neural Networks, Vol. 6, No. 2, March 1995.
- [9] Kumpan S. Narendra and M. A. L. Thatachar, Learning Automata An Introduction, New Jersey, Prentice Hall, 1989.
- [10] K. Najim and A. S. Poznyak, Learning Automata Theory and Applications, Elsevier Science Ltd, 1994.
- [11] P. Mars, J.R. Chen and R. Nombiar, Learning Algorithms Theory and Applications in Signal Processing, Control and Communications, CRC Press, New York, 1998.
- [12] S. Lakshimavarhan, Learning Algorithms Theory and Applications, New York, Springer - Verlag, 1981.
- [13] J. T. Tou and R. C. Gonzalez, Pattern Recognition Principals, Reading, MA: Addison-Welsey, 1974.
- [14] M. R. Meybodi and S. Lakshimavarhan, "On a class of Learning Algorithms which have a symmetric Behavior under Success and Failure", Springer - Verlag Lecture Notes in Statistics, PP. 145-155, 1984.
- [15] M. R. Meybodi, "Results on Strongly Absolutely Expedient Learning Automata", Proceedings of Inference Conference 86, ed. D.R. Moates and R. Butrick (Athens, Ohio: Ohio University Press, 1987), pp. 197-209.
- [16] N. Baba and H. Handa, "Utilization of Hierarchical Structure Stochastic Automata for the Back Propagation Method with Momentum", proc. of IEEE ICNN-95, pp. 389-393, 1995.
- [17] John R. Deller, John G. Proakis, and John H. L. Hansen, Discrete Time Processing of Speech Signal, Mc Millan, New York, 1993.
- [18] Sadaoki Furui, Digital Speech Processing, Synthesis and Recognition, Marsel Dekker inc., New York and Basel, 1989.
- [19] G. R. Rezaei, Study the Performance of Learning Automata in Pattern Recognition, Msc. Thesis, Computer Engineering Department, Amir Kabir University of Technology, 1999. (in Farsi)
- [20] B. John Oemmen and Edward V. de St. Croix, "String Taxonomy Using Learning Automata", IEEE Trans. on Sys., Man and Cybern, Vol. 27, No. 3, April 1997.
- [21] G. R. Rezaei and M.R. Meybodi, "Automatic Determination of Discriminant Functions for Pattern Recognition", Proceedings of Fourth International Conference of Computer Society of Iran, pp. 298-309, 1998 (in Farsi).
- [22] G. R. Rezaei and M.R. Meybodi, "Determining Discriminant Functions for Pattern Classification Using Hierarchical Learning Automata", Proceedings of Seventh International Conference on Electrical Engineering of Iran, pp. 9-16, 1999 (in Farsi).