

تعیین خودکار توابع تمایز برای دسته‌بندی الگوها

غلامرضا رضایی

محمد رضا امیدی

دانشکده مهندسی کامپیوتر

دانشگاه صنعتی امیرکبیر

تهران - ایران

چکیده

مسئله یادگیری قاعده تصمیم‌گیری جهت دسته‌بندی الگوهای با نظارت^(۱) با هدف مینیم کردن تعداد دسته‌بندی‌های غلط، یکی از مسائل قدیمی محسوب می‌شود. یکی از روشهای دسته‌بندی الگوها، استفاده از توابع تمایز^(۲) می‌باشد. در این مقاله روشهای جدیدی براساس بازی آناتامونهای یادگیر جهت تعیین توابع تمایز ارائه شده است. در این روشها، تابع تمایز از طریق یک جستجوی احتمالی^(۳) در فضای پارامترهای تابع تمایز بدست می‌آید. آنچه ما در این مقاله بر آن تاکید داریم، استفاده از آناتامونهای یادگیر با ساختار ثابت به عنوان بازیگران بازی و بررسی کارایی این آناتامونها در یادگیری توابع تمایز بهینه خطی یا غیر خطی می‌باشد.

کلمات کلیدی: شناسایی الگو، توابع تمایز، آناتامونهای یادگیر

۱- مقدمه

این مقاله به مسئله دسته‌بندی الگوهای با نظارت با هدف مینیم کردن تعداد دسته‌بندی‌های غلط می‌پردازد. اگر $P(w_i|x)$ احتمال شرطی تعلق داشتن بردار x به کلاس w_i باشد، با استفاده از قانون:

$$x \in w_i \text{ if } P(w_i|x) > P(w_j|x), j \neq i$$

بهترین قاعده تصمیم‌گیری که تعداد دسته‌بندی‌های غلط را مینیم می‌کند، حاصل می‌شود [۱] [۲] [۱۳]. چون غالباً توزیع $P(w_i|x)$ معلوم نیست، این قانون باید با استفاده از یک سری نمونه‌های آموزشی با دسته‌بندی شناخته شده از الگوها، یادگرفته شود. برای یادگیری این قاعده تصمیم‌گیری، دو رویکرد کلی وجود دارد. یک رویکرد، تخمین تابع چگالی شرطی $f(x|w_i)$ از روی مجموعه نمونه‌های آموزشی می‌باشد که در این صورت، با داشتن احتمال هر کلاس $P(w_i)$ ، قانون بیز^(۴) می‌تواند قاعده تصمیم‌گیری را تعیین کند [۱].

رویکرد دیگر استفاده از توابع تمایز است. در یک مسئله دو کلاسه، اگر $g(\cdot)$ تابع تمایز باشد، قاعده تصمیم‌گیری به صورت زیر خواهد بود:

$$\begin{aligned} \text{if } g(x) > 0 \text{ then } x \in w_1 \\ \text{if } g(x) < 0 \text{ then } x \in w_2 \end{aligned}$$

روشهای متعددی جهت تعیین تابع تمایز ارائه شده است. در صورتیکه کلاسه‌های الگوها به طور خطی جدایی پذیر باشند، با استفاده از الگوریتمهای پرسپترون، می‌توان توابع تمایز به فرم $g(X) = \theta^T X$ (علامت ترفاده است) را تعیین کرد [۱] [۱۳]. در صورتیکه کلاسه‌ها به طور خطی جدایی پذیر نباشند، روش Ho-Kashaplike با استفاده از مینیم کردن متوسط خطای مربعات، تابع تمایز را تعیین می‌کند [۱۳]. برخی از تکنیکهای یادگیری توابع تمایز، براساس تقریبهای احتمالی^(۵) بنا شده‌اند [۱۳] [۳]. در این روشها، هدف، پیدا کردن بردار وزن θ به گونه‌ای است که $g(X) = \theta^T X$ مجموع خطای

مربعات را مینیم کند و یا به نوعی تقریبی از $p(w_i|x)$ را حاصل کند. هیچ کدام از روشهای فوق الزاماً احتمال دسته‌بندی‌های غلط را مینیم نمی‌کنند، زیرا برآوردن شرطی مانند مینیم کردن متوسط خطای مربعات، لزوماً به معنی مینیم کردن احتمال دسته‌بندی غلط نیست.

روشهایی نیز براساس آناتامونهای یادگیر^(۶) جهت تعیین تابع تمایز ارائه شده است. Barto و Andrew روشی بنام AR_p ارائه کرده‌اند که هدف آن شناسایی تابع تمایز $g(X) = \theta^T X$ با استفاده از آناتامونهای یادگیر می‌باشد [۳]. در این روش یک رابطه لجنی^(۷) بین هر عضو از نمونه ورودی و هر عمل تصمیم‌گیری ایجاد می‌شود و آناتامونهای یادگیر سعی می‌کنند که با مشاهده هر نمونه ورودی، تصمیم صحیح را اتخاذ کنند. در این روش از آناتامونهای یادگیر با ساختار متغیر با دو عمل استفاده شده و از الگوریتم شبیه الگوریتم Robins - Monro [۳] با هدف مینیم کردن متوسط خطای مربعات استفاده شده است. این روش محدودیت خطی بودن تابع تمایز را دارد و همچنین به دلیل برقرای یک رابطه لجنی بر روی مجموعه نمونه‌های آموزشی، مجموعه نمونه‌ها باید محدود باشد.

Sastry و Thatachar روشی را جهت یادگیری تابع تمایز بهینه از فضای توابع تمایز با استفاده از بازی آناتامونهای یادگیر با پاسخ پکان ارائه کرده‌اند [۴]. الگوریتم ارائه شده در این روش بر اساس آناتامونهای یادگیر با ساختار متغیر می‌باشد که احتمال هر عمل با استفاده از تخمین از ماتریس پاداش^(۸) بازی که در طول یادگیری ساخته می‌شود، بهنگام می‌شود. هر آناتامون شرکت کننده در بازی، یکی از پارامترهای تابع تمایز را تعیین می‌کند. مجموعه اعمال هر آناتامون، مجموعه مقادیر ممکن برای آن پارامتر می‌باشد. در این روش، کلیه ترکیبهای ممکن عملهای آناتامونهای شرکت کننده در بازی به منظور تخمین ماتریس پاداش بازی، در حافظه ذخیره می‌شود. بازی ارائه

Bays Rule (۲)
Pay Off Matrix (۸)

Probabilistic Search (۲)
Associative Relation (۷)

Discriminate Functions (۲)
Learning Automata (LA) (۶)

Supervised Pattern Recognition (۱)
Stochastic Approximations (۵)

نشد در این روش، اساس کار این مقاله را تشکیل می‌دهد با این تفاوت که ما از آتوماتونهای یادگیر با ساختار ثابت استفاده می‌کنیم.

Phananskar, Thatachar روشی را براساس شبکه‌ای پیشخور^(۱) و آتوماتونهای یادگیر، جهت پیاده‌سازی یک سیستم دستبندی سطحی براساس دستبندی خطی قطعه به قطعه^(۲) ارائه کرده‌اند [۵] [۶]. در این روش با استفاده از توابع تمایز خطی قطعه به قطعه محدوده‌های هر کلاس تخمین زده می‌شود. در این روش از آتوماتونهای یادگیر با ساختار متغیر با شیوه به‌هنگام سازی L_{R-1} استفاده و نشان داده شده که سیستم می‌تواند به یک نقطه بهینه محلی^(۳) (و نه فرا-سراسری) اگر شرایط اولیه مناسب باشد، همگرا شود.

Sastry و Thatachar با ارائه PLA (Parameterized LA) روشی را جهت شناسایی قطعه بهینه سراسری^(۴) با استفاده از شبکه‌های پیشخور پیشنهاد کرده‌اند [۷] [۸]. در این روش بجای آتوماتونهای استاندارد از آتوماتون جدیدی به نام PLA که دارای ساختار متغیر بوده و قابلیت دریافت بردار محتوی^(۵) از محیط را دارد، استفاده و نشان داده شده است که تحت شرایط ویژه‌ای، مجموعه آتوماتونهای می‌تواند به نقطه بهینه سراسری همگرا شود.

آنچه در کلیه روشهای ارائه شده قبلی بر اساس آتوماتونهای یادگیر مشترک است، استفاده از آتوماتونهای یادگیر با ساختار متغیر به عنوان اجرای بازی یا سیستم یادگیری می‌باشد. در این مقاله برای یادگیری توابع تمایز بجای استفاده از آتوماتونهای یادگیر با ساختار متغیر، از آتوماتونهای یادگیر با ساختار ثابت استفاده گردیده و نشان داده شده است که آتوماتونهای یادگیر با ساختار ثابت، کارایی بالاتری نسبت به آتوماتونهای یادگیر با ساختار متغیر دارا می‌باشند. بخشهای بعدی مقاله به صورت زیر تنظیم شده است.

در بخش ۲ به معرفی اجمالی آتوماتونهای یادگیر می‌پردازیم و در بخش ۳ بازی آتوماتونهای یادگیر با پاسخ یکسان را معرفی می‌کنیم. در بخش ۴ یک مسئله نمونه سه کلاسه را تعریف می‌کنیم که در طول مقاله، روشهای مختلف روی آن پیاده می‌شود. در بخش ۵ روش ارائه شده توسط Sastry و Thatachar را بررسی می‌کنیم. در بخش ۶ به بررسی آتوماتونهای یادگیر با ساختار متغیر استاندارد و در بخش ۷ به بررسی آتوماتونهای یادگیر با ساختار ثابت استاندارد می‌پردازیم. در بخشهای ۸ و ۹ آتوماتونهای یادگیر با ساختار ثابت جدیدی که دارای کارایی بالاتری هستند پیشنهاد می‌گردد و در بخش ۱۰ به نتیجه‌گیری می‌پردازیم.

۲ - آتوماتون یادگیر

ایده آتوماتونهای یادگیر به مدل‌سازی رفتار تصمیم‌گیری سیستمهای بی‌نویکی بازمی‌گردد. این مدل، بر اساس انتخاب یک عمل از مجموعه‌ای محدود از اعمال با توجه به تجربیات گذشته، به‌نشد است. آتوماتون یادگیر در یک محیط تصادفی ناشناس فعالیت می‌کند، عملی را به محیط اعلام می‌کند و پاسخ آن را از محیط دریافت می‌نماید و براساس پاسخ دریافت شده سعی در یادگیری عمل بهینه دارد.

یک آتوماتون یادگیر، یک آتوماتون احتمالی^(۶) است که با محیط خود به صورت پیشخور^(۷) تراکنش دارد. خروجی آتوماتون 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 آتوماتون ارسال می‌گردد. آتوماتون یادگیر براساس پاسخ محیط، استراتژی خود را در تصمیم‌گیریهایی آینده اصلاح می‌کند. هدف از فعالیت آتوماتون، انجام اعمالی از مجموعه اعمال آتوماتون است که بهترین پاسخ را از محیط دریافت کند.

به طور رسمی، آتوماتون یادگیر بوسیله ۵ تایی

- | | | | |
|--|-------------------------|---|---------------------------------------|
| Global Optimum Point (۴) | Local Optimum Point (۳) | 3-Layer Linear Piecewise Classification (۲) | Feed Forward Network (۱) |
| | Feed Back (۵) | Stochastic Automaton (۶) | Context Vector (۵) |
| Variable Structure Learning Automata (۹) | | | Fixed Structure Learning Automata (۸) |
| | | | State Probability Vector (۱۱) |
| | | | Action Probability Vector (۱۰) |



شکل ۱: رابطه آتوماتون یادگیر با محیط

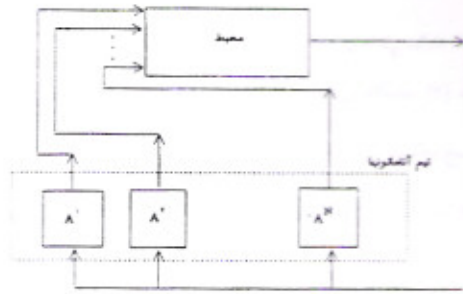
از نظر ساختاری، آتوماتونهای یادگیر به دو دسته تقسیم می‌شوند: آتوماتونهای یادگیر با ساختار ثابت^(۸) و آتوماتونهای یادگیر با ساختار متغیر^(۹). در آتوماتونهای یادگیر با ساختار ثابت، نحوه تغییر حالت آتوماتون بوسیله تابع $F(\cdot, \cdot)$ از قبل مشخص می‌شود. براساس حالت آتوماتون در مرحله k ام، عمل $\alpha(k)$ با استفاده از تابع $G(\cdot)$ به محیط اعمال می‌شود و با دریافت پاسخ $\beta(k)$ از جانب محیط، حالت بعدی آتوماتون تعیین می‌گردد. برخی از مدل‌های آتوماتونهای یادگیر با ساختار ثابت در بخش ۷ این مقاله بررسی شده‌اند.

در آتوماتونهای یادگیر با ساختار متغیر، نحوه تغییر حالت از قبل مشخص نیست. در این مدل، بردار احتمال عمل^(۱۰) یا بردار احتمال حالت^(۱۱) آتوماتون مورد توجه است. بردار احتمال عمل یا حالت، احتمال انتخاب یک عمل یا قرار گرفتن آتوماتون در یک حالت را در مرحله k ام تعیین می‌کند. می‌توان در حل مسائل این دو بردار را با حفظ شرایط معادل گرفت [۹]. بردار احتمال عمل به صورت زیر تعریف می‌شود:

$$P(k) = \{P_1(k), \dots, P_r(k)\}$$

$$P_i(k) = \Pr \{ \alpha(k) = \alpha_i \mid \beta(\cdot) \dots \beta(k-1) \}, i = 1, \dots, r$$

در آتوماتونهای یادگیر با ساختار متغیر، یک توزیع اولیه (معمولاً یکنواخت) به عنوان مقادیر اولیه بردار احتمال عمل در نظر گرفته می‌شود. در هر مرحله، آتوماتون یک عمل را با توجه به احتمالات عملها، انتخاب و به محیط اعلام



شکل ۲. بازی آتامتونی یادگیر با پاسخ مشخص

مقادیر ممکن برای پارامتر θ_i (و در نتیجه اعمال آتامتون A^i) از قبل تعیین شده‌اند. هدف از این بازی، پیدا کردن مقادیری برای پارامترهای θ_i می‌باشد بطوریکه تعداد دستبندیهای غلط مینیمم گردد. در هر مرحله، آتامتون A^i با توجه به شیوه تصمیم‌گیری خود، یک عمل (مقدار) از مجموعه اعمال خود (مجموعه مقادیر ممکن برای θ_i) را انتخاب می‌کند و به محیط اعلام می‌نماید. یک (یا چند) نمونه از بردارهای ویژگی موجود در محیط توسط تابع نمایش بدست آمده (با استفاده از پارامترهای انتخاب شده) دستبندی می‌شود و نتیجه بصورت نسبت تعداد دستبندیهای صحیح به کل نمونههای آزمایش شده، به تیم ارسال می‌شود. بر اساس پاسخ بدست آمده، آتامتونهای شرکت کننده در بازی، شیوه تصمیم‌گیری خود را به‌هنگام می‌نمایند. هدف از بازی، بدست آوردن تابع نمایشی است که بر اساس آن احتمال دریافت پاسخ نامطلوب و یا به عبارت دیگر تعداد دسته‌بندی‌های غلط نمونه‌ها مینیمم گردد.

۴- یک مسئله نمونه

کلیه روش‌های مختلف پیاده‌سازی شده در این مقاله، روی یک مسئله نمونه انجام شده است که در این بخش معرفی می‌گردد.

مسئله: پیدا کردن تابع نمایش بهینه برای سه کلاس W_1 ، W_2 و W_3 با توزیعهای نرمال با مشخصات زیر:

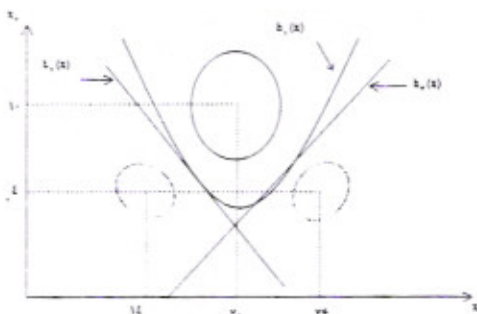
$$W_1: M = (20, 10)^T, \quad \Sigma = \begin{pmatrix} 4 & 0 \\ 0 & 6 \end{pmatrix}$$

$$W_2: M = (14, 2)^T, \quad \Sigma = \begin{pmatrix} 2 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

$$W_3: M = (26, 2)^T, \quad \Sigma = \begin{pmatrix} 2 & 0.5 \\ 0.5 & 2 \end{pmatrix}$$

بردار M ، بردار میانگین و ماتریس Σ ، ماتریس کوواریانس توزیعها

می‌باشد. در شکل ۳، شکل توزیع کلاسه‌ها و تابع نمایش هر کلاس نشان داده شده است.



شکل ۳. توزیع کلاسه‌های مسئله نمونه

می‌نماید و پاسخ آن را از محیط دریافت می‌کند. بر اساس پاسخ دریافت شده، آتامتون، این احتمالها را به‌هنگام می‌نماید و در مرحله بعدی بر اساس 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 احتمالها الگوریتم یادگیری (۱) با روش تشویق (۲) گفته می‌شود که می‌تواند تلمی خطی یا غیرخطی از احتمالها باشد. برخی از این الگوریتمهای یادگیری در بخش ۶ این مقاله بررسی شده‌اند.

آتامتونهای یادگیر با ساختار ثابت بر اساس توابع $G(\cdot)$ ، $F(\cdot)$ به دو دسته قطعی (۳) و غیرقطعی (۲) تقسیم می‌شوند. در مدل قطعی، بر اساس حالت ورودی فعلی آتامتون، به طور قطع می‌توان خروجی فعلی و حالت بعدی آتامتون را تعیین کرد اما در مدل غیرقطعی، یکی از توابع (یا هر دو) $G(\cdot)$ ، $F(\cdot)$ غیر قطعی می‌باشند.

محیط بر اساس مجموعه β (مجموعه پاسخ محیط) به سه دسته تقسیم می‌شود:

- مدل P : در این حالت مجموعه β دو عضوی است که یک عضو بیانگر پاسخ مطلوب و عضو دیگر بیانگر پاسخ نامطلوب می‌باشد و معمولاً به صورت $\beta = \{0, 1\}$ بیان می‌شود.

- مدل Q : در این مدل مجموعه خروجی به صورت مجموعه محدود $\{\beta_1, \dots, \beta_m\}$ تعریف می‌شود. خروجی محیط می‌تواند یکی از مقادیر گسسته β_1, \dots, β_m را اختیار کند.

- مدل S : در این مدل مجموعه خروجی، یک مجموعه نامحدود به صورت $\beta = \{(a, b)\}$ تعریف می‌شود که a ، b اعداد حقیقی هستند. پاسخ محیط می‌تواند یکی از مقادیر پیوسته در ناحیه (a, b) را اختیار کند.

در مدل‌های Q ، S معمولاً مقدار بیشتر برای پاسخ محیط به عنوان پاسخ بهتر در نظر گرفته می‌شود

بر اساس مجموعه C ، محیط به دو دسته ایستا (۵) و پویا (۶) طبقه‌بندی می‌شود. در مدل ایستا مقدار C_i ها 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.

برای اطلاعات بیشتر درباره آتامتونهای یادگیر، می‌توانید به [۹] [۱۰] [۱۱] [۱۲] [۱۳] [۱۴] [۱۵] مراجعه کنید.

۳- بازی آتامتونها

از نظر تئوری، یک مسئله را می‌توان با یک آتامتون منفرد که با محیط در ارتباط است حل کرد اما در این صورت ممکن است مجبور به استفاده از آتامتونی با تعداد عملهای بسیار زیاد بشویم که می‌تواند منجر به سرعت همگرایی بسیار پایین گردد [۹]. برای حل این مشکل، می‌توان از مجموعه‌ای از آتامتونها که به صورت یک تیم در حل مسئله شرکت می‌کنند، استفاده کرد. مدل‌های مختلفی از بازی آتامتونها 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].

همانطور که قبلاً اشاره شد، هدف از این مقاله، شناسایی پارامترهای تابع نمایش از طریق آتامتونهای یادگیر می‌باشد. فرض کنیم تابع نمایش 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 دستبندی شود. با استفاده از N آتامتون یادگیر در قالب یک بازی با پاسخ یکسان مطابق شکل (۲) می‌توان پارامترهای فوق را تخمین زد.

مطابق شکل (۲)، آتامتونهای A^1, \dots, A^N در بازی شرکت کرده‌اند. هر

آتامتون A^i دارای r_i عمل $\alpha_1^i, \dots, \alpha_{r_i}^i$ می‌باشد. هر عمل آتامتون A^i یک

مقدار ممکن برای پارامتر θ_i از تابع نمایش در نظر گرفته می‌شود.

توسط ماشین A^n که دارای π_n عمل است، انتخاب شده باشد، بردار احتمال عمل ماشین توسط رابطه (۲) به‌نگام می‌شود. در رابطه (۲)، $E_j^n(k)$ بهترین احتمال دریافت پاسخ مطلوب از طرف محیط است اگر آتاماتون A^n عمل α_j^n را در مرحله k انتخاب کرده باشد. $E_j^n(k)$ با استفاده از ماتریس پاداش بازی در طول یادگیری ساخته می‌شود. تابع $S_{ts}^n(k)$ ملاکی جهت تشویق یا تنیبه کردن آتاماتون A^n است که طبق رابطه (۳) تعریف می‌شود. تابع $f(\cdot)$ یک تابع اکیدا صعودی بین $[0, 1]$ می‌باشد و λ_{ts} پارامتر یادگیری است. با توجه به تحلیل الگوریتم در [۴] با انتخاب λ_{ts} به حد کافی کوچک، تیم آتاماتونها با احتمال نزدیک به یک، به سمت استراتژی بهینه همگرا می‌شود.

$$P_j^n(k) = P_j^n(k) - \lambda_{ts} [f(E_{ts}^n(k)) - f(E_j^n(k))]$$

$$\left[S_{ts}^n(k) P_j^n(k) + S_j^n(k) (1 - P_j^n(k)) \frac{P_{ts}^n(k)}{r_{ts}-1} \right], j = i$$

$$P_{ts}^n(k+1) = 1 - \sum_{j=1}^n P_j^n(k+1) \quad (2)$$

$$S_{ts}^n(k) = 1 \text{ if } E_{ts}^n(k) > E_j^n(k) \\ = 0 \text{ otherwise} \quad (3)$$

الگوریتم (۱) اسکلت برنامه یادگیری را برای یادگرفتن یک تابع نمایان نشان می‌دهد. این اسکلت در کلیه روشهای پیادسازی شده در این مقاله یکسان است. در این الگوریتم، رویه GenerateSamples مجموعه نمونه‌های آموزشی و تست را تولید می‌کند. رویه SelectActions عمل هر آتاماتون را انتخاب می‌کند. رویه Discriminate بوسیله عملهای انتخاب شده یک (یا چند) نمونه را دسته‌بندی می‌نماید و رویه UpdateGame با توجه به نتیجه دسته‌بندی، شیوه انتخاب عمل را به‌نگام می‌کند. در آتاماتونها با ساختار متغیر، انتخاب عمل با استفاده از بردار احتمال عمل مطابق الگوریتم ۲ انجام می‌شود. رویه EndOfGame پایان بازی را مشخص می‌کند. در آزمایشهای انجام شده در این بخش و بخش بعد، رسیدن احتمال انتخاب عمل بهینه به بیش از ۹۹٪ به عنوان شرط خاتمه بازی در نظر گرفته شده است.

برای پیادسازی روش فوق، تابع $f(x) = x$ در نظر گرفته شده است و به ازاء مقادیر مختلف λ_{ts} برای $0.001, 0.002, 0.005, 0.01, 0.02, 0.05$ نمودارهای (۱-الف) و (۱-ب) رسم شده‌اند. نمودار (الف) دقت جداسازی (درصد دسته‌بندی صحیح) توسط سه تابع نمایان یاد گرفته شده و نمودار (۱-ب) تعداد مراحل مورد نیاز برای یادگیری تابع h_1 را برای مقادیر مختلف λ_{ts} نشان می‌دهند (از آنجا که توابع نمایان مستقل از هم تعیین می‌شوند، تعداد مراحل مورد نیاز برای تعیین هر تابع نمایان، می‌تواند متفاوت از دیگری باشد). ملاحظه می‌شود که با کم شدن λ_{ts} ، دقت بالا می‌رود اما سرعت همگرایی کاهش می‌یابد. به ازاء $\lambda_{ts} = 0.001$ ، دقت متوسط به دست آمده ۹۵/۶۵٪ با تعداد متوسط مراحل ۱۰۸۲۸۴ بوده است.

```
for i = 1 to 100 do
  GenerateSamples;
  Repeat
    SelectActions;
    DiscResult = Discriminate;
    UpdateGame(DiscResult);
  Until EndOfGame;
End for
SaveResults;
```

الگوریتم ۱

توابع $h_1(x)$ ، $h_2(x)$ و $h_3(x)$ ، توابع نمایان هستند که باید یاد گرفته شوند بطوریکه:

$$h_1(x) > 0 \Rightarrow x \in W_1 \\ h_1(x) < 0 \Rightarrow x \in W_i$$

با توجه به شکل توزیع، توابع:

$$h_1(x) = x_1 - 0.19x_1^2 + 0.09x_1 - 0.00 > 0 : x \in W_1$$

$$h_2(x) = -x_1 - x_1 + 0.23 > 0 : x \in W_2$$

$$h_3(x) = -x_1 + x_1 - 0.17 > 0 : x \in W_3$$

می‌توانند تقریبهای خوبی از توابع نمایان کلاسه‌های مسئله باشند.

همانطور که مشاهده می‌شود $h_1(x)$ دارای ۳ پارامتر و $h_2(x)$ و $h_3(x)$ دارای ۲ پارامتر هستند که باید یاد گرفته شوند. بنابراین از سه تیم آتاماتون مجزا شامل ۲، ۲، ۳ تیم آتاماتون که به طور مستقل کار می‌کنند، برای یادگیری توابع نمایان استفاده می‌کنیم:

$$h_1 : A_{11}, A_{12}, A_{13} \rightarrow h_1 = x_1 - \alpha_{11}x_1^2 + \alpha_{12}x_1 - \alpha_{13}$$

$$h_2 : A_{21}, A_{22} \rightarrow h_2 = -x_1 - \alpha_{21}x_1 + \alpha_{22}$$

$$h_3 : A_{31}, A_{32} \rightarrow h_3 = -x_1 + \alpha_{31}x_1 - \alpha_{32}$$

آتاماتونها یادگیری مستقیم که پارامترهای تابع نمایان $h_1(x)$ را تعیین می‌کنند. α_{ij} پارامترهای تابع نمایان $h_i(x)$ هستند. مجموعه مقادیر ممکن برای پارامترهای این سه تابع نمایان بصورت زیر در نظر گرفته شده‌اند. مقادیر ممکن برای هر پارامتر، در جلوی نام آتاماتون تعیین کننده آن پارامتر، آمده است.

$$A_{11}: 0.15, 0.17, 0.19, 0.21, 0.23 \quad (1)$$

$$A_{12}: 0.6, 0.7, 0.8 \quad A_{13}: 0.7, 0.8, 0.9$$

$$A_{21}: 0.5, 0.75, 1, 1.25, 1.5 \quad A_{22}: 0.19, 0.21, 0.23, 0.25, 0.27$$

$$A_{31}: 0.5, 0.75, 1, 1.25, 1.5 \quad A_{32}: 0.19, 0.21, 0.23, 0.25, 0.27$$

در قسمت یادگیری، برای هر کلاس ۵۰۰ نمونه تصادفی آموزشی در نظر گرفته می‌شود. پس از انجام یادگیری، تست و گزارش کارایی روی مجموعه‌های ۵۰۰ تایی که متفاوت با مجموعه‌های آموزشی هستند، انجام می‌گیرد. در هر آزمایش، نتایج اعلام شده، میانگین ۱۰۰ شبیه‌سازی می‌باشد.

مقادیر ممکن برای پارامترهای توابع نمایان مسئله نمونه که در رابطه (۱) آمده‌اند، براساس مشخصات توزیع نمونه‌های مسئله که مشخص می‌باشند، انتخاب شده‌اند. اما در عمل، بدلیل اینکه توزیع نمونه‌ها مشخص نیست، چنین امکانی وجود ندارد. برای رفع این مشکل، می‌توان بازه بزرگی برای هر پارامتر در نظر گرفت و با در نظر گرفتن یک ساختار درختی از آتاماتونها یادگیر برای یادگیری هر پارامتر، پارامتر مورد نظر را تعیین کرد. در این ساختار سطوح بالاتر (نزدیک به ریشه) بیانگر بازه‌های بزرگتر و سطوح پایینتر، بیانگر بازه‌های کوچکتر هستند و برگها بیانگر مقادیر نهایی پارامترها می‌باشند. با توجه به دقت مورد انتظار برای هر پارامتر، می‌توان تعداد سطوح ساختار درختی را انتخاب کرد. مقدار هر پارامتر با توجه به مسیر فعال شده در هر درخت تعیین می‌گردد [۴].

۵ - روش Thatachar و Sastry [۴]

در این قسمت بشرح روش پیشنهاد شده توسط Thatachar و Sastry می‌پردازیم. این روش، یک روش براساس آتاماتونها یادگیر با ساختار متغیر است. گروه آتاماتونها در یک بازی همکار با پاسخ یکسان با هدف یافتن تابع نمایان بهینه از فضای توابع نمایان موجود شرکت می‌کنند. در این روش تقریبی از ماتریس پاداش بازی در طول یادگیری ذخیره می‌شود. ماتریس پاداش، یک ماتریس چند بعدی است که اعضای آن احتمالات دسته‌بندی صحیح برای هر مجموعه مقدار ممکن برای پارامترهای آتاماتونها شرکت کننده در بازی می‌باشند. عمل یادگیری، سعی در همگرا کردن تیم به سمت انتخاب مجموعه اعمال بهینه (تابع نمایان بهینه) می‌نماید. اگر فرض کنیم در مرحله k ام، عمل α_{ts}^n

مسئله نمونه با استفاده از سه الگوریتم به‌هنگام سازی فوق پیاده‌سازی شد. بر اساس نتایج به‌دست آمده، نکات زیر قابل توجه است:

۱- الگوریتم L_{R-P} قادر به همگرا شدن نمی‌باشد.
۲- الگوریتم L_{R-I} همواره همگرا شده و قادر به تعیین مقادیر مناسبی برای پارامترهای تولید تمایز می‌باشد.

۳- همگرایی الگوریتم L_{R-EP} بستگی به نسبت $\frac{a}{b}$ دارد. این نسبت باید از یک مقدار مشخص γ_0 که به مسئله مورد نظر و محدوده پارامترهای a و b بستگی دارد، بزرگتر باشد.

۴- از بین دو الگوریتم L_{R-EP} و L_{R-I} ، الگوریتم L_{R-EP} دقت بیشتری تولید می‌کند اما نسبت به الگوریتم L_{R-I} ، دارای سرعت همگرایی پایین‌تری می‌باشد.

در آزمایش انجام شده مقادیر a و b مقادیر $0.005, 0.003, 0.002, 0.001, 0.0005, 0.0001$ و مقادیر $0.005, 0.003, 0.002, 0.001, 0.0005, 0.0001$ برای b در نظر گرفته شده‌اند و به ازاء کلیه زوجهای (a, b) با شرط $\frac{a}{b} > 10$ ، الگوریتم اجرا گردید. نتایج به‌دست آمده نشان می‌دهد که برای الگوریتم L_{R-EP} با کوچکتر شدن a و b دقت افزایش می‌یابد اما تعداد مراحل مورد نیاز می‌تواند بطور چشمگیری افزایش یابد تا آنجا که سیستم عملاً ناپایدار شده و بجای همگرا شدن به یک نقطه بهینه، بین دو یا چند نقطه بهینه محلی نوسان کند. در آزمایش انجام شده، در حالت $a = 0.007, b = 0.0007$ سیستم به ناپایداری رسید (بعد از $20,000,000$ مرحله، سیستم همچنان در حال نوسان بود). به همین علت برای $a < 0.007$ ، $b < 0.0007$ ، $\frac{a}{b} > 10$ روی زوجهای (a, b) در نظر گرفته شده است.

نمودارهای (۲-الف) و (۲-ب) دقت دست‌یابی و تعداد مراحل مورد نیاز جهت همگرایی را برای الگوریتمهای فوق به ازاء مقادیر مختلف a, b با شرط گفته شده را نشان می‌دهند. با توجه به نمودارهای (۲-الف) و (۲-ب) ملاحظه می‌شود که با کاهش مقادیر a, b دقت افزایش می‌یابد ولی در عوض تعداد مراحل مورد نیاز بیشتر می‌شود. در نمودارهای فوق به ازاء $b = 0.0001$ ، $a = 0.0001$ دقت و تعداد مراحل برای الگوریتم L_{R-I} به‌دست می‌آید. برای الگوریتم L_{R-I} به ازاء $a = 0.002, b = 0.0002$ دقت دست‌یابی $97/416$ با تعداد متوسط مراحل 26621 و برای الگوریتم L_{R-EP} با ازاء $a = 0.002, b = 0.0001$ دقت $97/750$ با تعداد مراحل متوسط 6012 حاصل شده است.

۷- آتاماتونهای یادگیر با ساختار ثابت

در این قسمت به بررسی کارایی آتاماتونهای یادگیر با ساختار ثابت استاندارد در یادگیری تولید تمایز در حل مسئله نمونه می‌پردازیم. آتاماتونهای بررسی شده در این قسمت عبارتند از: $TSetline_{KN,K}$ و $TSetline_{LKN,K}$.

Krinsky و Krylov [۹]. در آتاماتونهای یادگیر با ساختار ثابت، انتخاب عمل توسط حالت آتاماتون مشخص می‌شود به این ترتیب که ابتدا آتاماتون از یک حالت اولیه $\Phi(0)$ شروع و عمل $\Phi(0)$ را انتخاب انتخاب می‌کند. با دریافت پاسخ عمل از طرف محیط و با توجه به ساختار آتاماتون و نحوه تغییر حالت آن، آتاماتون به حالت $\Phi(1)$ می‌رود و این چرخه ادامه پیدا می‌کند. شکلهای (۲-الف) تا (۲-د) گراف تغییر حالت آتاماتونهای فوق‌الذکر را نشان می‌دهند.

مدلهای ارائه شده بر اساس مدل محیط P می‌باشند. در گرافهای مذکور برای آتاماتون، دو عمل α_1, α_2 در نظر گرفته شده است. هر ماشین $2N$ حالت دارد که N عضو حافظه آتاماتون نامیده می‌شود. اگر $\Phi(k)$ ، حالت ماشین در مرحله k ام، بین Φ_1 تا Φ_N باشد، عمل α_1 و اگر $\Phi(k)$ بین Φ_{N+1} تا Φ_{2N} باشد،

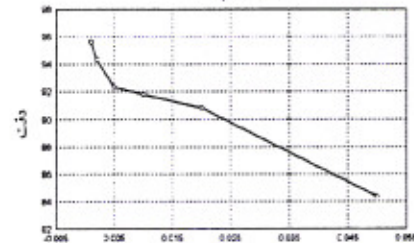
```
g := random; s := 0; i := 1;
while (i <= r) and not ((g >= s) and (g <= s + p[i])) do s := s + p[i];
SelectedAction := i;
```

الگوریتم ۲

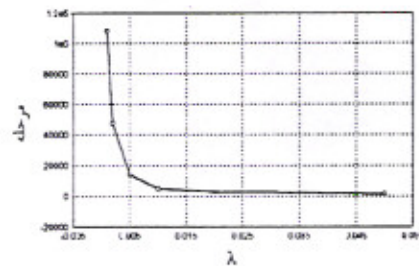
Function EndOfGame : Boolean;

Result := True if for All A^i if $p(\text{OptimalAction}) \geq 0.99$;
End EndOfGame;

الگوریتم ۳



نمودار ۱-الف



نمودار ۱-ب

۶- آتاماتونهای یادگیر با ساختار متغیر

در این بخش به بررسی کارایی آتاماتونهای یادگیر با ساختار متغیر که دارای شیوه به‌هنگام‌سازی خطی احتمالات می‌باشند، می‌پردازیم. اگر در مرحله k ام، عمل $\alpha_i(k)$ انتخاب شده باشد، بردار احتمال عمل بصورت زیر به‌هنگام می‌شود:

الف- با دریافت پاسخ مطلوب:

$$\begin{aligned} P_i(k+1) &= P_i(k) + a[1 - P_i(k)] \\ P_j(k+1) &= (1-a)P_j(k), \quad j \neq i \end{aligned} \quad (2)$$

ب- با دریافت پاسخ نامطلوب:

$$\begin{aligned} P_i(k+1) &= (1-b)P_i(k) \\ P_j(k+1) &= \frac{b}{r-1} + (1-b)P_j(k), \quad j \neq i \end{aligned} \quad (3)$$

a و b پارامترهای یادگیری هستند [۹].

با توجه به روابط (۲) و (۳) با دریافت پاسخ مطلوب از محیط احتمال عمل انتخاب شده افزایش یافته و احتمال سایر عملها کاهش می‌یابد و با دریافت پاسخ نامطلوب، احتمال عمل انتخاب شده کاهش یافته و احتمال سایر عملها افزایش می‌یابد. باید توجه داشت که به‌هنگام سازی به گونه‌ای انجام می‌شود که در هر مرحله k دقت به‌دست آید.

$$\forall i \text{ in } [1 \dots r] : P_i(k) \geq 0, \sum_{i=1}^r P_i(k) = 1$$

برحسب اینکه مقادیر a و b رابطه آنها نسبت به هم چگونه باشند سه نوع الگوریتم به‌هنگام سازی حاصل می‌شود که عبارتند از:

- $L_{R-P} : 1 > a = b = 0$
- $L_{R-EP} : 1 > a > b = 0$
- $L_{R-I} : 1 > a > 0, b = 0$

در آتامتونهای با ساختار متغیر از بردار احتمال عمل برای مشخص کردن خاتمه الگوریتم استفاده گردید. در آتامتونهای با ساختار ثابت به دلیل اینکه چنین برداری به طور صریح وجود ندارد، باید ملاک دیگری برای خاتمه الگوریتم تعریف شود. بدین منظور برای هر عمل α_i^n از آتامتون A^n که دارای Γ_n عمل است، وزن w_i^n را با مقدار اولیه $\frac{1}{\Gamma_n}$ تعریف می‌کنیم. شرط خاتمه بازی، رسیدن وزن عمل بهینه به حد معینی خواهد بود $(w_{i_n}^n \geq w_0)$. این وزن‌ها در پایان هر مرحله توسط روابط ۶ به‌هم‌گام می‌شوند. در روابط ۶ فرض بر این است که در مرحله k ام عمل α_i^n انتخاب شده باشد.

$$P_i^n(k+1) = \frac{P_i^n(k) (W-1) + 1}{W} \quad (6)$$

$$P_j^n(k+1) = \frac{P_j^n(k) (W-1)}{W}, \quad j \neq i$$

W عددی صحیح است و پارامتر یادگیری نامیده می‌شود. اگر عمل انتخاب

شده توسط آتامتون A^n در مرحله k ام عمل $\alpha_{i_n}^n$ باشد، با توجه به رابطه (۶) بسادگی می‌توان نشان داد:

$$\Delta w_{i_n}^n(k) = w_{i_n}^n(k+1) - w_{i_n}^n(k) = \frac{1 - w_{i_n}^n(k)}{W} (W-1)$$

$$\sum_{i=1}^{\Gamma_n} w_{i_n}^n(k) = 1 \quad (7-2)$$

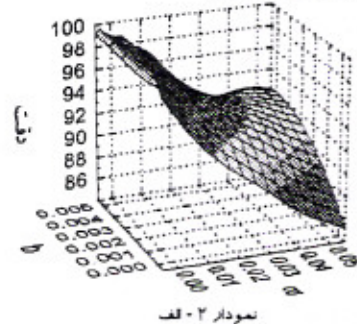
رابطه (۷-۱) نشان می‌دهد که با بزرگتر شدن W سرعت همگرایی کاهش می‌یابد.

پارامتر دیگری که در یادگیری موثر است، تعداد نمونه‌های آموزشی است که در هر مرحله، تست تابع تمایز روی آنها انجام می‌شود. دقت تابع بدست آمده در هر مرحله می‌تواند با آزمایش روی S نمونه از مجموعه نمونه‌های آموزشی سنجیده شود. این دقت به عنوان پاسخ ارسالی محیط به تیم آتامتونها و احتمال مناسب بودن عملهای انتخاب شده در هر مرحله در نظر گرفته می‌شود. اگر فرض شود در هر مرحله S نمونه دستبندی شوند و تعداد S_1 نمونه صحیح و

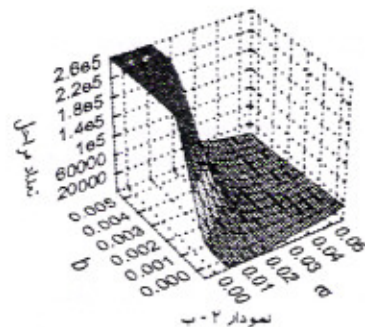
$S_1 = S - S_0$ نمونه غلط دستبندی شده باشند، مقدار $\Gamma_e = \frac{S_1}{S}$ به عنوان پاسخ محیط در نظر گرفته می‌شود. به این ترتیب از دید محیط، احتمال درست بودن و $q_e = 1 - \Gamma_e$ احتمال غلط بودن مجموعه عملهای انتخاب شده می‌باشد با توجه به اینکه S_1 مقادیر صحیح بین $[0, S]$ را اختیار می‌کند، مجموعه خروجی محیط (مجموعه مقادیر ممکن برای Γ_e) $\left\{0, \frac{1}{S}, \frac{2}{S}, \dots, \frac{S-1}{S}, 1\right\}$ خواهد بود. اگر $S = 1$ باشد، مجموعه خروجی به $\{0, 1\}$ تبدیل می‌شود که بیانگر مدل P برای محیط است و اگر $S > 1$ باشد، مجموعه خروجی شامل $S+1$ مقدار گسسته خواهد بود که بیانگر مدل Q برای محیط است. در حالت مدل Q ، مقدار بزرگتر برای پاسخ محیط بیانگر پاسخ مطلوبتر خواهد بود.

در ادامه این بخش به بررسی پارامترهای مختلف این آتامتونها در حل مسئله نمونه می‌پردازیم. ابتدا تأثیر مقادیر مختلف S را بررسی می‌کنیم و سپس به بررسی تأثیر عمق حافظه و پارامتر W می‌پردازیم. شرط پایان الگوریتم در پیاده‌سازی کلیه الگوریتمهای آتامتونهای یادگیر با ساختار ثابت، رسیدن وزن عمل بهینه به بیش از ۰/۹۹ در نظر گرفته شده است.

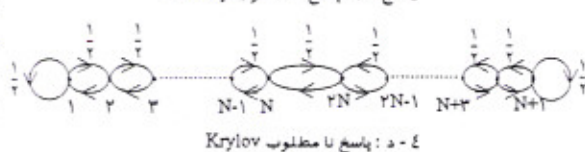
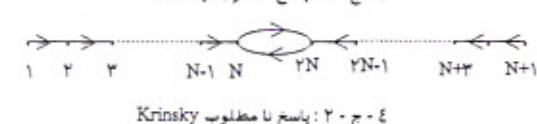
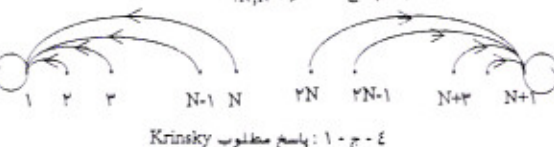
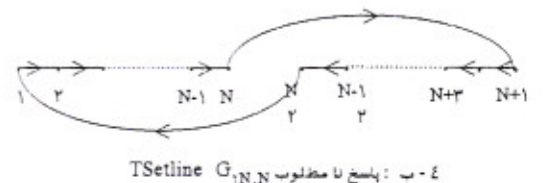
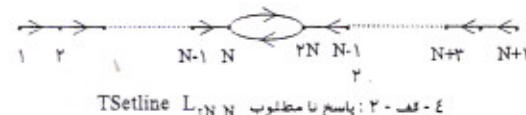
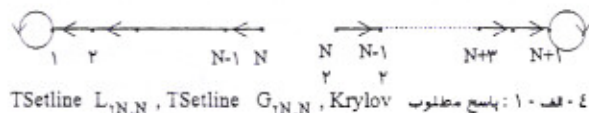
عمل α_i^n انتخاب می‌شود. عمق حافظه یکی از پارامترهایی است که در کارایی آتامتونهای یادگیر با ساختار ثابت بسیار موثر است. با توجه به گرافهای فوق در صورتیکه عمق حافظه ۱ باشد ($N=1$) کلیه آتامتونهای فوق (به غیر از مدل Krylov) فرم یکسان خواهند داشت.



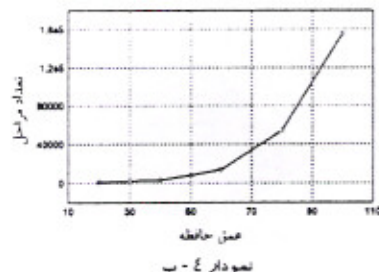
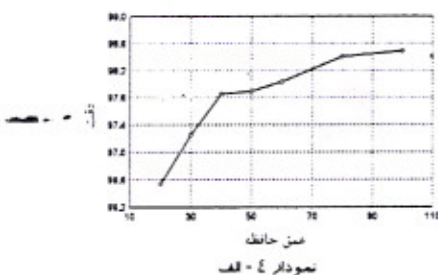
نمودار ۲-الف



نمودار ۲-ب



حافظه ۱ و W های مختلف کاربری را تحقیق می‌کنیم. نمودارهای (۲-الف) و (۲-ب) نمودارهای دقت دستبندی و تعداد مراحل مورد نیاز به ازای مقادیر مختلف ۲۰، ۳۰، ۴۰، ۵۰، ۶۰، ۸۰، ۱۰۰ می‌باشد.



همانطور که انتظار می‌رفت با افزایش W دقت افزایش می‌یابد و سرعت همگرایی کاهش می‌یابد. ملاحظه می‌شود که به ازاء $W = 100$ دقت ۹۸/۲۸۶ با تعداد مراحل ۱۵۵۹۷۰ و به ازاء $W = 80$ دقت ۹۸/۴۰۹ با تعداد مراحل ۵۴۰۱۱ و به ازاء $W = 60$ دقت ۹۸/۰۳۷ با تعداد مراحل ۱۴۰۰۱ بدست می‌آید. به ازاء $W > 60$ سرعت همگرایی به سرعت کم می‌شود (تعداد مراحل مورد نیاز به سرعت افزایش می‌یابد) و دقت به کندی افزایش می‌یابد.

جهت بررسی تاثیر عمق حافظه (N) و W در کاربری، مقادیر ۲، ۳، ۴، ۵، ۶، ۷، ۸، ۹، ۱۰، ۱۲، ۱۵ برای N و مقادیر ۱۰۰، ۲۰۰، ۳۰۰، ۴۰۰، ۵۰۰، ۶۰۰، ۸۰۰، ۱۰۰۰، ۱۲۰۰، ۱۵۰۰ را در نظر گرفته و به ازای زوجهای مختلف (N, W) الگوریتم را اجرا نموده‌ایم. نمودارهای (۵-الف) تا (۵-ج) نتایج این بررسی را نشان می‌دهند. نمودارهای (۵-الف) و (۵-ب)، نمودارهای دقت و سرعت همگرایی مربوط به آتاماتون $Tsetline L_{KN,K}$ ، نمودارهای (۵-ج) و (۵-د) مربوط به آتاماتون $Tsetline G_{KN,K}$ ، نمودارهای (۵-ه) و (۵-و) مربوط به آتاماتون Krinsky و نمودارهای (۵-ز) و (۵-ح) مربوط به آتاماتون Krylov می‌باشند. مشاهده می‌شود که آتاماتونها دارای رفتاری تقریباً یکسان می‌باشند. با افزایش W همانطور که انتظار می‌رفت دقت افزایش و سرعت همگرایی کاهش می‌یابد و با افزایش عمق حافظه، دقت دستبندی پایین می‌آید. به ازای $W = 1500$ و $N = 2$ بهترین دقتا بدست آمده که برای آتاماتونهای مختلف عبارتند از: آتاماتون $Tsetline L_{KN,K}$ دقت ۹۷/۵۵۱ با متوسط تعداد مراحل ۲۸۶۳۶، آتاماتون $Tsetline G_{KN,K}$ دقت ۹۷/۵۲۹ با تعداد مراحل ۲۳۱۴۶، آتاماتون Krinsky دقت ۹۷/۵۸۰ با تعداد مراحل ۲۶۶۳۳، آتاماتون Krylov دقت ۹۷/۰۹۹ با تعداد مراحل ۲۳۲۳۴. با مقایسه این نمودارها و نتایج بدست آمده برای آتاماتونهای با عمق حافظه ۱، می‌توان ادعا کرد که آتاماتونهای با عمق حافظه ۱، هم از نظر سرعت و هم از نظر دقت، دارای کاربری بالاتری هستند.

با انجام تغییراتی در آتاماتون با عمق حافظه ۱ میتوان دقت و سرعت همگرایی را بهبود بخشید که چند نمونه از این تغییرات در دو بخش بعدی پیشنهاد شده است.

۱-۷- تاثیر حجم نمونه آزمایشی

در این بخش به بررسی تاثیر پارامتر S بر روی آتاماتون $Tsetline L_{KN,K}$ با عمق حافظه ۲ و $W = 200$ می‌پردازیم. اسکلت اصلی برنامه در الگوریتم ۱ آمده است. الگوریتم ۵ نحوه تغییر حالت آتاماتون را بیان می‌کند. در الگوریتم ۵، متغیر State حالت ماشین را حفظ می‌کند که شامل عمل انتخاب شده و عمق حافظه عمل می‌باشد و T_e پاسخ محیط است. در صورت تنبیه شدن و رسیدن عمق عمل به ۱، عمل دیگری به غیر از عمل فعلی انتخاب می‌شود که این انتخاب می‌تواند بصورت تصادفی انجام گیرد [۹].

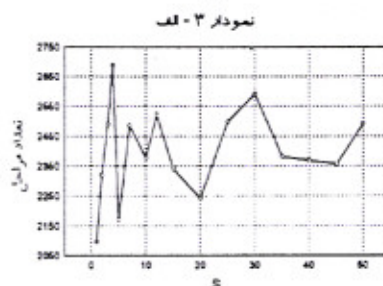
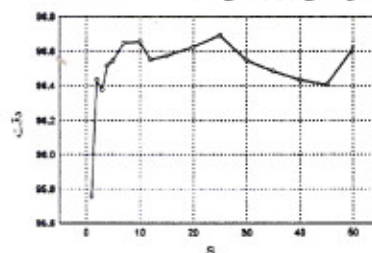
الگوریتم فوق به ازای مقادیر مختلف ۱، ۲، ۳، ۴، ۵، ۷، ۱۰، ۱۲، ۱۵، ۲۰، ۲۵، ۳۰، ۳۵، ۴۰، ۴۵، ۵۰ اجرا شده و دقت دستبندی و تعداد مراحل مورد نیاز برای همگرایی در نمودارهای (۳-الف) و (۳-ب) آمده است.

Procedure UpdateGame (r_e : Real)

```
Reward := Random <=  $r_e$ ;   Penalty := not Reward;
if Reward then
    if State.Depth <= MemoryDepth then inc (State.Depth);
End if
if Penalty then
    if State.Depth > 1 then dec (State.Depth)
    else State.Action := AnotherAction
End if
End UpdateGame
```

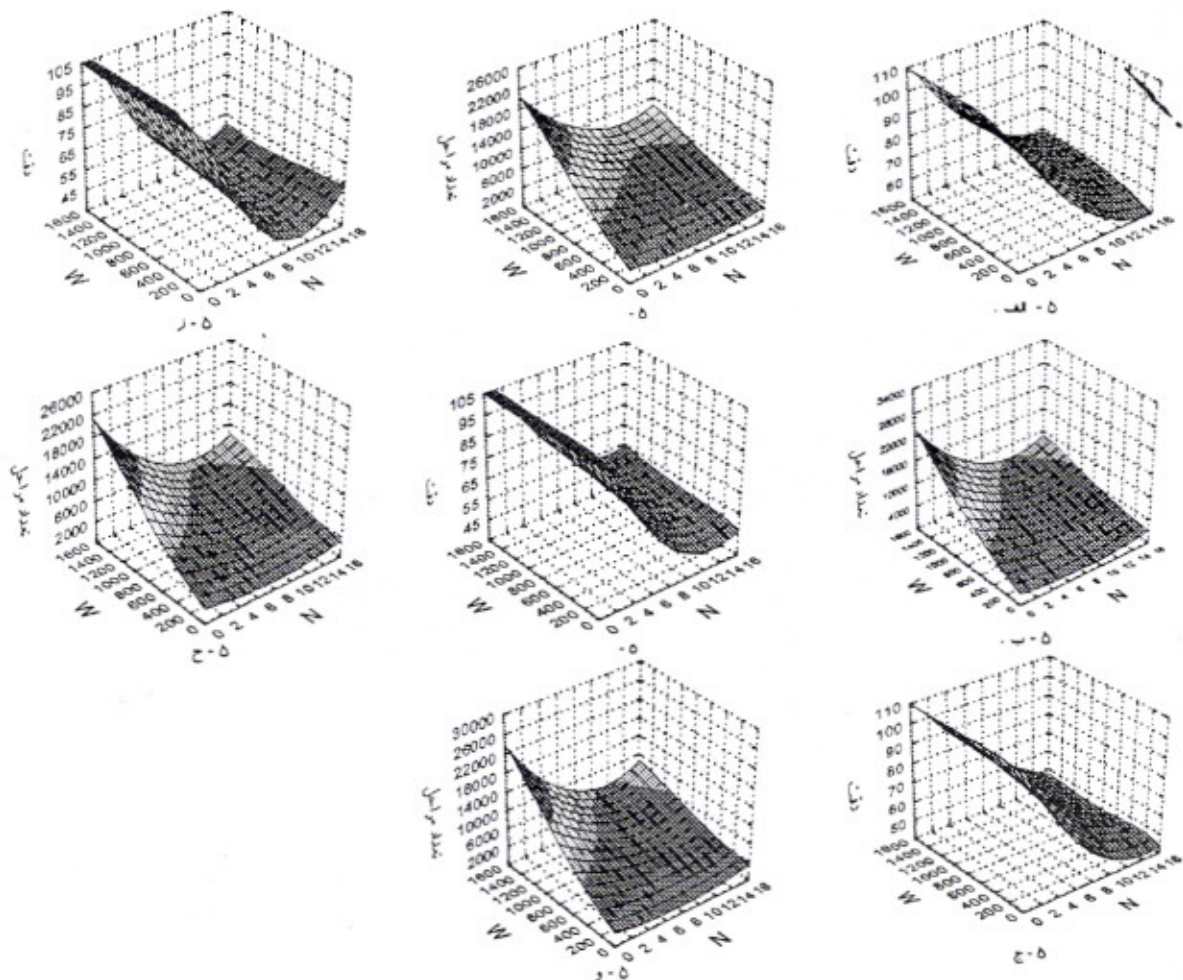
الگوریتم ۵

ملاحظه می‌شود که به ازای $S = 2$ بهبود زیادی (حدود ۱٪) در دقت ایجاد شده در حالی که تعداد مراحل حدود ۳۵۰ مرحله افزایش یافته است. بهترین نتیجه به ازای $S = 25$ بدست آمده که در آن دقت دستبندی صحیح ۹۶/۶۹۵ با تعداد مراحل ۲۲۹۷ بوده است. باید توجه داشت که تعداد مراحل الزاماً به معنی تعداد مقایسه‌ها نیست بلکه اگر تعداد مراحل مورد نیاز در هر آزمایش k باشد و تعداد نمونه‌های مورد آزمایش هر مرحله S باشد، تعداد مقایسه‌ها $S.k$ خواهد بود که در تحلیل سرعت عمل آتاماتونها نقش اصلی را بازی می‌کند.



۲-۷- تاثیر عمق و پارامتر W

برای عمق حافظه ۱ رفتار بیشتر آتاماتونهای فوق یکسان می‌باشد و در ضمن محدوده پارامتر W برای عمق ۱ با سایر عمقها متفاوت است، پس ابتدا به ازای عمق



۸- آتاماتونهای پیشنهادی گروه اول
در این قسمت آتاماتونی پیشنهاد می‌دهیم که از دقت و سرعت همگرایی بهتری در حل مسئله شناسایی تابع تمایز برخوردار است. نتایج بدست آمده در بخش قبل نشان داد که عمق حافظه ۱ می‌تواند بهترین پاسخ را در حل مسئله فراهم کند. این نکته بدین معنی است که ایجاد سهولت در تغییر عمل می‌تواند باعث هدایت آتاماتون به سمت انتخاب عمل بهینه شود. اما اگر بدست آوردن دقت بالا مد نظر باشد، این ایجاد سهولت در تغییر عمل می‌تواند باعث نوسانات اضافی آتاماتون بین عملهای مختلف و در نتیجه کند شدن سرعت همگرایی آتاماتون شود. در آتاماتونهای با عمق حافظه ۱، تغییر عمل با دریافت پاسخ نامطلوب از جانب محیط انجام می‌گیرد یعنی پاداش و تنیه با توجه به پاسخ محیط تعیین می‌شود. با تمایز قائل شدن بین پاسخ محیط و تنیه و تشویق آتاماتون، می‌توان به نتایج بهتری رسید. این تمایز از طریق قرار دادن فیلترهای مناسب بر روی پاسخ محیط بمنظور ضعیفتر و قویتر کردن آن حاصل می‌شود.

$$P = \begin{cases} r_e & \text{if } w_{i_n}^n(k) > w_{i_n, \max}^n(k) \\ r_e \times \frac{w_{i_n}^n(k)}{w_{i_n, \max}^n(k)} & \text{otherwise} \end{cases} \quad (8)$$

$$q = \begin{cases} q_e & \text{if } w_{i_n}^n(k) < w_{i_n, \max}^n(k) \\ q_e \times \frac{w_{i_n}^n(k)}{w_{i_n, \max}^n(k)} & \text{otherwise} \end{cases} \quad (9)$$

به طوریکه:

$$w_{i_n, \max}^n(k) = \max \left\{ w_j^n(k) \mid j = 1, \dots, r_n, j \neq i_n \right\} \quad (10)$$

در رولت فوق w_j^n وزن عمل α_j^n است که با استفاده از رولت (۶) به‌دست‌آمده است.

فرض کنیم آتاماتون A^n که دارای r_n عمل است و در مرحله k

عمل $\alpha_{i_n}^n(k)$ را انتخاب و به محیط اعلام کند. با فرض اینکه r_e احتمال مناسب بودن عمل و $q_e = 1 - r_e$ احتمال نامناسب بودن عمل در محیط

احتمال پاداش

$$p = \begin{cases} r_e & \text{if } PI \geq P_{Max} \\ r_e \times \frac{PI}{P_{Max}} & \text{otherwise} \end{cases} \quad (11)$$

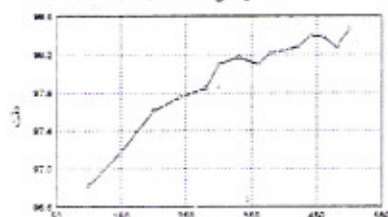
احتمال تنبیه

$$q = \begin{cases} q_e & \text{if } PI < P_{Max} \\ q_e \times \frac{P_{Max}}{PI} & \text{otherwise} \end{cases} \quad (12)$$

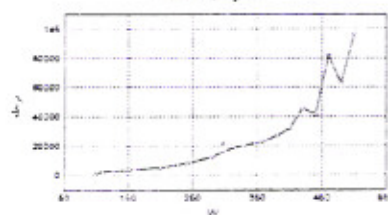
بطوریکه

$$PI = \sqrt[N]{\prod_{k=1}^N w_{i_n}^n(k)}, P_{Max} = \sqrt[N]{\prod_{k=1}^N w_{i_n, max}^n(k)} \quad (13)$$

طبق رابطه (۱۳)، میانگین هندسی وزنهای عملی انتخاب شده توسط آتاماتونهای شرکت کننده در بازی است و P_{Max} میانگین هندسی حداکثر وزن عملی انتخاب شده است. آتاماتونی که طبق روابط (۱۱) تا (۱۳) تنبیه و تشویق می‌شود را آتاماتون نوع ۲ می‌نامیم. نمودارهای (۷-الف) و (۷-ب) نتایج پلست آمده از پیاده‌سازی آتاماتون نوع ۲ به ازاء مقادیر مختلف W می‌باشد. با انجام تغییر فوق خطی بودن تعداد مراحل از بین رفته و رفتار آتاماتون آشفته‌تر و همگرایی مجموعه کندتر شده است بطوریکه حداکثر دقت $98.8/281$ به ازاء $W = 480$ یا تعداد مراحل 62735 پلست آمده است.



نمودار ۷-الف

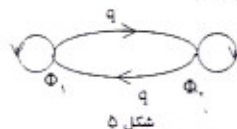


نمودار ۷-ب

در شیوه تشویق و تنبیه دیگر را نیز در این رستا آزمایش می‌کنیم و آتاماتونهای مربوطه را آتاماتونهای نوع ۳ و نوع ۴ می‌نامیم. در آتاماتون نوع ۳، برای تنبیه و تشویق مشابه روابط (۱۱) و (۱۲) عمل می‌کنیم با این تفاوت که به جای PI از $w_{i_n}^n$ ، یعنی وزن عمل انتخاب شده استفاده گردیده است. نتایج این روش در نمودارهای (۸-الف) و (۸-ب) آمده است. شیوه دیگر، استفاده از میانگین حسابی به جای میانگین هندسی و استفاده از $w_{i_n}^n$ در روابط (۱۱) و (۱۲) به جای PI می‌باشد. به این ترتیب P_{Max} با استفاده از رابطه (۱۲) محاسبه می‌شود. نتایج روش دوم در نمودارهای (۸-ج) و (۸-د) آمده است. آتاماتون نوع ۳ به ازای مقادیر $100, 150, 200, 250, 300, 350, 400, 450, 500, 550, 600, 650, 700, 750$ و آتاماتونهای نوع ۴ به ازای مقادیر $100, 150, 200, 250, 300, 350, 400, 450, 500, 550, 600, 650, 700, 750$ برای W آزمایش شده‌اند. حداکثر دقت پلست آمده در روش اول به ازای $W = 700$ برابر $98.4/223$ با تعداد مراحل 8833 و در روش دوم به ازای $W = 550$

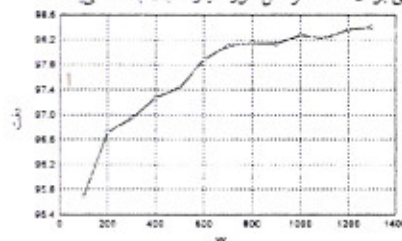
می‌شود و $w_{i_n, max}^n$ حداکثر وزن عملی انتخاب شده می‌باشد. طبق رابطه (۸) اگر آتاماتون در حالتی باشد که وزن عمل انتخاب شده در آن حالت، کمتر و حداکثر وزن سایر عملها باشد و پاسخ مطلوب دریافت کرده باشد، با احتمال کمتری نسبت به پاسخ محیط تشویق می‌شود. این موضوع بدین معنی است که احتمالاً عمل انتخاب شده فعلی، عمل بهینه نمی‌باشد و بنابراین پاسخ محیط تضعیف می‌شود. طبق رابطه (۹) اگر آتاماتون در حالتی باشد که وزن عمل انتخاب شده در آن حالت، بیشترین وزن باشد و پاسخ نامطلوب دریافت کرده باشد، احتمال تنبیه شدن آن کمتر از پاسخ محیط در نظر گرفته می‌شود به این معنی که احتمالاً عمل انتخاب شده، عمل بهینه بوده اما در مرحله‌ای باعث بروز پاسخ نامطلوب از جانب محیط شده است، بنابراین احتمال تنبیه آتاماتون تضعیف می‌شود. به این ترتیب روابط (۸) تا (۱۰) با ایجاد تمایز بین پاسخ محیط و تنبیه و تشویق آتاماتون، می‌توانند از نوسانات اضافی آتاماتون جلوگیری کنند. آتاماتونی که طبق روابط (۸) تا (۱۰) تغییر حالت می‌دهد را آتاماتون نوع ۱ می‌نامیم.

در آزمایش انجام شده از آتاماتونهای با عمق حافظه ۱ و پاداش و تنبیه طبق روابط (۸) تا (۱۰) استفاده گردیده است. شکل ۵ گراف تغییر حالت آتاماتون مورد نظر را برای دو عمل نشان می‌دهد.

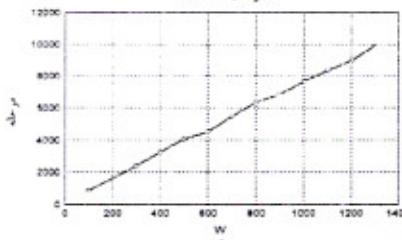


شکل ۵

نمودارهای (۶-الف) و (۶-ب) نتایج دقت و تعداد مراحل مورد نیاز را برای آتاماتون نوع ۱ به ازاء مقادیر مختلف W نشان می‌دهند. مشاهده می‌شود که نتایج پلست آمده از سرعت و دقت بهتری برخوردار می‌باشند. به ازای $W = 700$ دقت دستمندی $98.4/102$ با متوسط تعداد مراحل 5353 و به ازای $W = 1300$ دقت $98.4/395$ با تعداد مراحل پلست آمده است. نکته جالب دیگری که در نمودار (۶-ب) به چشم می‌خورد خطی بودن تعداد مراحل مورد نیاز نسبت به W می‌باشد.



نمودار ۶-الف



نمودار ۶-ب

در قدم بعدی سعی در ایجاد ارتباطی محکمتر بین آتاماتونهای شرکت کننده در بازی می‌کنیم به این امید که نتایج بهتری پلست آید. اگر فرض کنیم N آتاماتون A^1, A^2, \dots, A^N در یک بازی با پاسخ پکشان شرکت کرده‌اند و عمل انتخاب شده در مرحله k توسط آتاماتون $A_{i_n}^n$ ، $A_{i_n}^n$ باشد احتمالهای پاداش و تنبیه آتاماتون را به صورت زیر تعریف می‌کنیم:

$$p = \begin{cases} r_e & \text{if } PI \geq P_{Max} \\ r_e \times \frac{PI}{P_{Max}} & \text{otherwise} \end{cases} \quad (20)$$

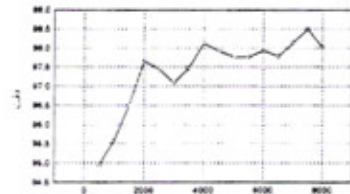
$$q' = \begin{cases} q_e & \text{if } PI < P_{Max} \\ q_e \times \frac{P_{Max}}{PI} & \text{otherwise} \end{cases} \quad (21)$$

بطوریکه:

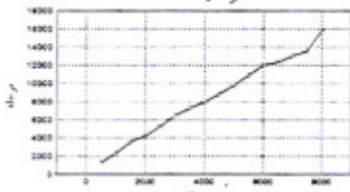
$$PI = \sqrt[N]{\prod_{n=1}^N w_{i_n}^n(k)}, P_{Max} = \sqrt[N]{\prod_{n=1}^N w_{i_n, max}^n(k)} \quad (22)$$

احتمال q طبق رابطه (۱۶) و $w_{i_n}^n$ ، $w_{i_n, max}^n$ طبق رابطه (۱۸) و

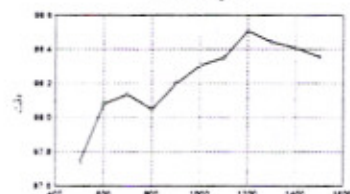
(۱۹) محاسبه می‌شوند. در این نوع آناماتون که آناماتون نوع ۶ نامیده می‌شود سعی شده است که ارتباط محکمتری بین آناماتونهای شرکت کننده در بازی برقرار شود. نتایج آزمایش برای این روش برای عمقهای ۵۰۰، ۶۰۰، ۷۰۰، ۸۰۰، ۹۰۰، ۱۰۰۰، ۱۱۰۰، ۱۲۰۰، ۱۳۰۰، ۱۴۰۰، ۱۵۰۰ نمودارهای (۱۰ - الف) و (۱۰ - ب) آمده است. با توجه به نمودارهای فوق مشاهده می‌شود که دقت و سرعت تا حدودی افزایش یافته است. حداکثر دقت بدست آمده به ازای عمق حافظه ۱۲۰۰ برابر ۹۸/۵۰۸ با تعداد مراحل ۱۰۹۹۷ می‌باشد.



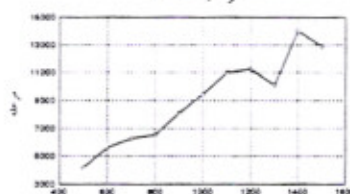
نمودار ۹ - الف



نمودار ۹ - ب



نمودار ۱۰ - الف



نمودار ۱۰ - ب

۱- نتیجه گیری

در بخشهای قبل روش ارائه شده توسط Thatachar و Sastry در شناسایی تابع تمایز و همچنین آناماتونهای پادگیر با ساختار متغیر و ثابت استاندارد پیاده سازی شد و کارایی هر یک از روشهای فوق در شناسایی تابع

رابطه (۱۸) برای هر عمل وزنی برابر نسبت عمق عمل به عمق حافظه آناماتون تعریف می‌گردد. رابطه (۱۹) حداکثر وزن سایر عملهای انتخاب نشده برای آناماتون را تعریف می‌کند و روابط (۱۵) تا (۱۷) احتمالاتی تنبیه و تشویق آناماتون را محاسبه می‌کنند. آناماتونی که به صورت فوق تغییر حالت می‌دهد را آناماتون نوع ۵ می‌نامیم.

سکلت اصلی برنامه شبیه ساز در الگوریتم ۱ آمده است. در الگوریتم ۶ رویه Initialize که مقدار دهی اولیه عمق را انجام می‌دهد، رویه UpdateGame که نحوه تغییر حالت آناماتون را پیاده سازی می‌کند و رویه EndOfGame که پایان بازی را مشخص می‌کند، برای یک بازی مشکل از N آناماتون، آورده شده است.

Procedure Initialize;

for $i := 1$ to r_n do $m(i) := N_n / r_n$

End Initialize

Procedure UpdateGame (r_e)

$q_e := 1 - r_e$; $w_i := m(\text{State.Action}) / N_n$;

$w_{i, max} := \text{Max}\{m(j) / N_n \mid j = i\}$

if $w_i > w_{i, max}$ then Reward := Random $\leq r_e$

else Reward := Random $\leq r_e * w_i / w_{i, max}$

Penalty := not Reward;

if Penalty then

if $w_i \leq w_{i, max}$ then Penalty := Random $\leq q_e$

else Penalty := Random $\leq q_e * w_{i, max} / w_i$

End if (Penalty)

if Reward then

if State.Depth $< N_n$ then inc (State.Depth);

for $i := 1$ to r_n and $i \neq \text{State.Action}$ do

if $m(i) > 1$ and (Random $\leq m(i) / N_n$) then dec ($m(i)$);

End for

End if; (Reward)

if Penalty then

$m(\text{State.Action}) := \text{State.Depth}$;

State.Action := Guess Another Action;

State.Depth := $m(\text{State.Action})$;

End if; (Penalty)

End UpdateGame;

Function EndOfGame : Boolean;

Result := True if for All n in $[1..N]$ $m(\text{Action.State}_n) / N_n \geq 0.99$;

End EndOfGame;

الگوریتم ۶

الگوریتم ۶ برای عملهای حافظه (N_n) مختلف

۵۰۰، ۱۰۰۰، ۱۵۰۰، ۲۰۰۰، ۲۵۰۰، ۳۰۰۰، ۳۵۰۰، ۴۰۰۰، ۴۵۰۰، ۵۰۰۰، ۵۵۰۰،

۶۰۰۰، ۶۵۰۰، ۷۰۰۰، ۷۵۰۰ اجرا گردید. نتایج مربوط به دقت دستمندی و

تعداد مراحل مورد نیاز در نمودارهای (۹ - الف) و (۹ - ب) آورده شده است.

همانطور که در نمودارهای (۹ - الف) و (۹ - ب) ملاحظه می‌شود تعداد

مراحل تقریباً به طور خطی با عمق حافظه زیاد می‌شود و با زیاد شدن عمق

حافظه آناماتون، دقت دسته بندی افزایش می‌یابد. حداکثر دقت به ازای عمق

۷۵۰۰ بدست آمده که برابر ۹۸/۵۰۳ با تعداد مراحل ۱۳۷۵۰ بوده است.

روش دیگری برای محاسبه احتمالاتی تشویق و تنبیه بر اساس میثکین

همنسوزی و زنها استفاده شده است. مقادیر p و q' (که قبلاً تعریف شده‌اند) طبق

روابط (۲۰) تا (۲۲) محاسبه می‌شوند:

- of Teams of Learning Automata", in Symp. Intell. Syst., Bangalore, Dec. 1991.
- [8] M. A. L. Thatachar and V. V. Phanaskar, "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. Lakshminavarhan, 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. Lakshminavarhan, "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. Mostes and R. Butrick (Athens, Ohio: Ohio University Press, 1987), pp. 197-209.

جدول ۱

دوش	پارامترها	دقت	تعداد مراحل
Sastry و Thatachar	$\lambda_i = 0.001$	۹۵/۶۵۰	۱۰۸۸۸۲
$L_R - I$	$a = 0.001$	۹۷/۳۱۶	۲۶۶۲۱
$L_R - \epsilon P$	$a = 0.002$ $b = 0.0001$	۹۷/۷۵۰	۶۰۰۱۲
TSetline $L_{KN,K}$ و Tsetline $G_{KN,K}$	$N = 1$ $W = 100$	۹۸/۲۸۶	۱۵۵۹۷۰
Krinsky برای عمق حافظه ۱	$N = 1$ $W = ۶۰$	۹۸/۰۳۷	۱۲۰۰۱
TSetline $L_{KN,K}$	$N = 2$ $W = 1500$	۹۷/۵۵۱	۲۸۶۳۶
Tsetline $G_{KN,K}$	$N = 2$ $W = 1500$	۹۷/۵۲۹	۲۳۱۳۶
Krinsky	$N = 2$ $W = 1500$	۹۷/۵۸۰	۲۶۶۳۳
Krylov	$N = 2$ $W = 1500$	۹۷/۰۹۹	۲۳۲۲۳
آتماتون نوع ۱	$W = 1300$	۹۸/۳۹۵	۹۸۹۹
آتماتون نوع ۲	$W = 280$	۹۸/۲۸۱	۶۲۷۳۵
آتماتون نوع ۳	$W = 700$	۹۸/۲۲۳	۸۸۳۳
آتماتون نوع ۴	$W = 550$	۹۸/۲۰۶	۱۸۱۶۷
آتماتون نوع ۵	$N_n = 7500$	۹۸/۵۰۳	۱۳۷۵۰
آتماتون نوع ۶	$N_n = 1200$	۹۸/۵۰۸	۱۰۹۹۷
قانون بیز	$P(w_1) = \frac{1}{3}$ $P(w_2) = \frac{2}{3}$	۹۸/۹۷۲	

نمونه مسئله نمونه بررسی شد. نتایج بدست آمده نشان دادند که آتماتونهای یادگیر با ساختار ثابت با عمق حافظه ۱، از سرعت همگرایی و دقت بهتری برخوردار می‌باشند. در بخش ۸ با اصلاح آتماتون ساختار ثابت با عمق حافظه ۱، آتماتونهای با ساختار ثابت جدیدی با کارایی بالاتر ارائه شدند. در بخش ۹ آتماتونهای ساختار ثابت دیگری پیشنهاد گردید که دارای عمق حافظه بزرگی بوده و شرط حاشیه از ساختار آتماتون استخراج می‌شود. در این آتماتونها کارایی بالاتر از طریق تمایز قائل شدن بین پاسخ محیط و تنبیه و تشویق آتماتون و همچنین ایجاد سهولت در تغییر عمل توسط آتماتون زمانی که پاسخ نامطلوب از جانب محیط بدست آمده است، حاصل شده است.

با توجه به اینکه مسئله حل شده از سه توزیع نرمال تشکیل شده است، می‌توان با استفاده از قانون بیز، دقت تابع تمایز بهینه را که دستمندی غلط را به‌نیم می‌کند اندازه‌گیری کرد. طبق قانون بیز برای جدا کردن کلاس w_1 از w_2 ، تابع $h_1(x)$ به صورت زیر می‌باشد:

$$h_1(x) = p(w_1)f(x|w_1) - p(w_2)f(x|w_2) \quad (23)$$

که در آن $f(x|w_i)$ تابع چگالی شرطی و $p(w_i)$ احتمال انتخاب کلاس w_i است. در استفاده از رابطه (۲۳) برای جداسازی کلاس w_1 از دو کلاس دیگر، کلاسهای w_1 ، w_2 را به صورت زیر در نظر می‌گیریم:

$$w_1 = w_2 + w_3, w_2 = w_1$$

به این ترتیب $p(w_1)$ برابر احتمال انتخاب کلاس w_1 و $p(w_2)$ احتمال انتخاب هر یک از دو کلاس w_2 یا w_3 می‌باشد. از آنجا که در پیاده‌سازی الگوریتمهای یادگیری در بخشهای گذشته در هر مرحله تابع تمایز روی یک نمونه تصادفی از مجموعه کل نمونهها آزمایش شده است، پس احتمال تعلق داشتن نمونه به کلاس w_1 برابر $\frac{1}{3}$ و احتمال تعلق

داشتن نمونه به کلاسهای w_2 یا w_3 برابر $\frac{2}{3}$ خواهد بود و در این صورت $p(w_1) = \frac{1}{3}$ ، $p(w_2) = \frac{2}{3}$ ، تابع $h_1(x)$ و $h_2(x)$ برای جداسازی کلاسهای w_1 و w_2 از سایر کلاسها به طور مشابه پیاده‌سازی می‌شوند. نتیجه ۱۰۰ بار آزمایش قانون بیز روی ۱۰۰ مجموعه نمونه مختلف، دقت متوسط ۹۸/۶۷۲ را بدست می‌دهد (برای $p(w_1) = p(w_2) = \frac{1}{3}$). دقت ۹۹/۱۰۷ حاصل می‌شود. جدول شماره ۱ خلاصه نتایج بدست آمده از کارایی روشهای مختلف ارائه شده در این مقاله در حل مسئله نمونه را نشان می‌دهد.

۱۱- مراجع

- [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] Mandsyam 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. Phanaskar, "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. Phanaskar, "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. Phanaskar and M. A. L. Thatachar, "Global Convergence