

ارائه الگوریتمهای نو مبتنی بر اتوماتانهای یادگیر برای تعیین تعداد و وزنهای ورودی نرونهاي لایه مخفی برای شبکه‌های عصبی سه لایه

بهبود مشعوفی

استادیار

گروه مهندسی دانشکده فنی، دانشگاه ارومیه

محمد رضا میبیدی

دانشیار

دانشکده مهندسی کامپیوتر، دانشگاه صنعتی امیرکبیر

سید احمد معتمدی

دانشیار

دانشکده مهندسی برق، دانشگاه صنعتی امیر کبیر

محمد باقر منهج

دانشیار

دانشکده مهندسی برق، دانشگاه صنعتی امیر کبیر

چکیده

الگوریتم پس انتشار خطای استاندارد، فقط در فضای وزنهای شبکه، با توپولوژی ثابت، عمل جستجو را انجام میدهد. تعداد لایه‌ها، نرونها و وزنهای شبکه، تأثیر بسزایی بر روی کارایی شبکه دارد. بنا بر این نیاز به الگوریتمهایی داریم که بتوانند بطور اتوماتیک ساختار مناسب شبکه را تعیین کنند. برای تعیین اندازه مطلوب برای شبکه‌های عصبی الگوریتمهای گوناگونی توسط افراد مختلف ارائه شده است. توسط آقایان میبیدی و بیگی الگوریتم جدیدی تحت عنوان WSA با استفاده از اتوماتان یادگیری مهاجرت اشیا برای تعیین تعداد و وزنهای طراحی گردیده است. این الگوریتم با حذف وزنهایی که دارای تأثیر کمتری هستند نه تنها باعث کاهش پیچیدگی شبکه میشود. بلکه قدرت تعمیم شبکه را نیز افزایش میدهد. در الگوریتم تعیین تعداد وزن، در ابتدا همه وزنهای موجود در شبکه روشن بوده و به همه وزنهای مدتی اجازه داده میشود تا در آموزش شبکه شرکت نمایند. وزنهای روشنی که قدر مطلق مقدار آنها از یک مقدار آستانه‌ای کمتر باشد جریمه شده و وزنهای روشنی که قدر مطلق آنها از یک مقدار آستانه‌ای دیگر بیشتر باشد پاداش میگیرند. وزنهای روشنی که قدر مطلق مقدار آنها بین دو مقدار آستانه‌ای قرار گیرد تغییری در وضعیت آنها ایجاد نمیشود. با انتخاب مقادیر بهینه برای این بازه‌ها میتوانیم در کمترین زمان به شبکه‌هایی با حداقل تعداد وزن که بتواند با خطای قابل قبولی الگوهای آموزش را یاد گرفته و همچنین از قدرت تعمیم قابل قبولی برخوردار باشد برسیم. در این مقاله با استفاده از اتوماتان یادگیر، بازه‌های تصمیم‌گیری را تطبیق داده‌ایم. در الگوریتم پیشنهادی تحت عنوان AWSA برای پاسخ دادن به اتوماتان مربوط به تنظیم بازه‌های تصمیم‌گیری، از معیار خطا کمک گرفته‌ایم. همچنین در این مقاله الگوریتم جدید دیگری تحت عنوان MWSA ارائه گردیده است. در الگوریتم پیشنهادی معیارهای جدیدی برای ارزیابی عملکرد وزنهای روشن و خاموش ارائه گردیده است. نتایج شبیه‌سازیهای مختلف بر روی مسائل شناسایی اعداد انگلیسی، شناسایی اعداد چاپی فارسی، تقریب تابع غیر خطی زمان گسسته از مرتبه دوم و باز شناسی فونمهای فارسی نشان میدهد الگوریتمهای پیشنهادی AWSA و MWSA دارای عملکرد بهتری نسبت به الگوریتمهای دیگر یادگیری ساختار میباشد.

کلمات کلیدی

مهندسی شبکه‌های عصبی، شبکه‌های عصبی چندلایه، ساختار شبکه‌های عصبی، الگوریتم پس انتشار خطا، اتوماتانهای یادگیر

New Learning Automata Based Algorithms for Determining the Number of Input Weights of Hidden Layers in a Multi layer Neural Networks

B. Mashoufi
Assistance Professor
Electrical Engineering Department,
Uremia University

M. R. Meybodi
Professor
Computer Engineering Department,
Amirkabir University of Technology

Sayed A. Motamedi
Associate Professor
Electrical Engineering Department,
Amirkabir University of Technology

M. B. Menhaj
Associate Professor
Electrical Engineering Department,
Amirkabir University of Technology

Abstract

Backpropagation algorithm performs gradient descent only in the weight space of a network with fixed topology. The Number of layers, Neurons and network weights have important influence on network performance. So algorithms that can find appropriate network architecture automatically are thus highly desirable. Researchers have proposed different algorithms for determining optimum size of neural networks. Meybodi and Beigy introduced the first learning automata based algorithms, called WSA algorithm. This algorithm by turning off the unimportant weights, not only reduces network complexity but also increases network generalization ability. At the beginning, all weights of the network are on and contribute to learning. The on weights whose absolute values are less than a threshold value, are penalized and those whose absolute value are larger than another threshold value, are rewarded. The on weights, whose absolute values lie between these two threshold values, neither rewarded, nor penalized. By choosing optimum values for these values we can obtain, networks with minimum number of weights which can learn training patterns with acceptable error and generalization ability. In this paper we introduce a new learning automata based algorithm, called AWSA for adaptation of parameters of WSA algorithm. Also, a new algorithm called MWSA is introduced to determine important weights in the multi layer neural networks. These algorithms are applied to number of problems such as recognition of English number. Persian printed numbers recognition, second order discrete time nonlinear function approximation and Persian phoneme recognition. The results obtained show that the proposed algorithms have better performance than other existing algorithms.

Keywords

Neural Networks Engineering, Multi layer Neural Networks, Neural Networks Topology, Backpropagation algorithm, Learning Automata

مقدمه

در سالهای اخیر مدل‌های شبکه عصبی زیادی، برای مسائل طبقه‌بندی الگو، تقریب تابع، بازشناسی گفتار و ... ارائه گردیده است. در بین اینها، شبکه‌های عصبی جلو رونده چند لایه، مهمترین آنها میباشد. روشهایی که از الگوریتم پس انتشار خطای استاندارد استفاده میکنند فقط در فضای وزنه‌های شبکه با توپولوژی ثابت، عمل جستجو را انجام میدهند [۱]. این روشها در حالت کلی فقط موقعی مناسب میباشد که ساختار شبکه درست انتخاب شده باشد. تعداد لایه‌ها، نرونها و وزنه‌های شبکه، تأثیر بسزائی بر روی کارائی شبکه دارد. شبکه‌های خیلی کوچک، قادر به یادگیری مسئله نبوده و شبکه‌های با ابعاد بزرگ دچار Overfitting شده و قدرت تعمیم پائینی خواهد داشت. علاوه براین بسیار کند بوده و هزینه بالائی خواهد داشت. حالت مشابه را در مسئله برازش منحنی با استفاده از چند جمله ایها شاهد هستیم. داده‌هایی را در نظر میگیریم که توسط یک تابع که نویز جمع‌شونده نیز در خروجی آن قرار دارد ایجاد شده است. یک چند جمله‌ای با تعداد پائین ضرائب قادر نخواهد بود از روی داده‌ها تابع را مدل‌سازی کند از طرف دیگر یک چند جمله‌ای با تعداد بالای ضرائب، نویز موجود در داده‌ها را نیز مدل‌سازی

کرده در نتیجه مدل خوبی را برای تابع ارائه نخواهد کرد. ولی اگر تعداد ضرائب بطور مناسب انتخاب شود، چند جمله‌ای، نمایش خوبی را برای تابع، همچنین پیشگویی دقیقی را برای داده‌های جدید ارائه خواهد کرد. در شبکه‌های عصبی نیز همین مسئله اتفاق می‌افتد. در این حالت نیز پیچیدگی مدل بایستی با مسئله تطبیق داده شود. بنابر این نیاز به الگوریتم‌هایی داریم که بتوانند بطور اتوماتیک ساختار مناسب شبکه را تعیین کنند. الگوریتم‌هایی را که تاکنون توسط افراد مختلف بمنظور تعیین اندازه مطلوب برای شبکه‌های عصبی ارائه شده است میتوان به پنج گروه عمده زیر تقسیم کرد [۴۳].

الف - الگوریتم‌های هرس: در این دسته از الگوریتم‌ها، از یک شبکه بزرگ شروع کرده و آنرا آموزش میدهم سپس وزنها و نرونهای زائد را حذف میکنیم. اندازه اولیه بزرگ، این اجازه را میدهد که شبکه سریعاً، با حداقل حساسیت به شرایط اولیه، آموزش ببیند. و پیچیدگی پائین شبکه هرس شده نیز قدرت تعمیم آنرا افزایش میدهد. این الگوریتمها بدو گروه عمده تقسیم میشوند. گروه اول، حساسیت تابع خطا نسبت به حذف یک عنصر را تخمین زده، عناصر با حداقل تأثیر را حذف میکند [۵]-[۲]. گروه دوم، عباراتی را به تابع هدف اضافه میکند که به شبکه برای انتخاب جوا بهای کارآمد پاداش میدهد [۱۳]-[۶].

ب - الگوریتم‌های سازنده: این الگوریتمها از یک شبکه کوچک شروع کرده سپس تا حصول یک جواب رضایت بخش، واحدهای مخفی و وزنهایی را به شبکه اضافه میکنند [۱۸]-[۱۴]، [۱].

ج - الگوریتم‌های ترکیبی: این دسته از الگوریتمها در حقیقت ترکیبی از الگوریتم‌های هرس و الگوریتم‌های سازنده میباشد. این الگوریتمها در طول آموزش با اضافه و حذف کردن وزنها و نرونها سعی در رسیدن به یک شبکه با ساختار اپتیمال را دارند [۲۰]، [۱۹].

د - الگوریتم‌های تکاملی: این الگوریتمها با استفاده از یک معیار کارآیی مثل حداقل خطا و یا پیچیدگی آموزش، در فضای ساختارها، دنبال مناسبترین ساختار میشوند. در این روش هر نقطه از فضای جستجو، متناظر با یک ساختار شبکه میباشد [۲۵]-[۲۱].

ه - الگوریتم‌های بر اساس اتوماتانهای یادگیر: تعدادی از محققین از اتوماتانهای یادگیر برای تطبیق پارامترهای الگوریتم BP استفاده کرده‌اند [۴۹]، [۵۰]، [۵۲]. برخی دیگر، از اتوماتانهای یادگیر برای ارائه ساختارهای بهینه با پیچیدگی آموزش کم و قدرت تعمیم بالا استفاده کرده‌اند [۲۶]، [۴۳]، [۴۸]، [۵۱]. در اولین الگوریتم ارائه شده [۲۶] تحت عنوان الگوریتم بقا از یک اتوماتان یادگیر مهاجرت اشیا بعنوان یک ابزار جستجوی عمومی استفاده شده است. این الگوریتم حین آموزش، ساختار مناسبی برای شبکه عصبی سه لایه از حیث پائین بودن پیچیدگی آموزش و قدرت تعمیم بالا تعیین میکند. الگوریتم بقا آموزش را با یک شبکه عصبی سه لایه بزرگ شروع کرده و با افزودن و کاستن نرونهای مخفی، تعداد نرونهای لایه مخفی شبکه را تعیین میکند. در ادامه کار در مقاله [۴۳] الگوریتم جدید دیگری با استفاده از اتوماتان یادگیری مهاجرت اشیا تحت عنوان الگوریتم بقا وزن WSA برای تعیین تعداد وزنها طراحی گردیده است. این الگوریتم با حذف وزنهایی که دارای تأثیر کمتری هستند نه تنها باعث کاهش پیچیدگی شبکه می‌شود. بلکه افزایش قدرت تعمیم شبکه را نیز دنبال دارد. الگوریتم تعیین تعداد وزن با خاموش و روشن کردن وزنها سعی در پیدا کردن وزنهایی مناسب دارد. در الگوریتم ارائه شده در ابتدا همه وزنهایی موجود در شبکه روشن بوده و به همه وزنها مدتی اجازه داده میشود تا در آموزش شبکه شرکت نمایند. وزنهایی روشنی که قدر مطلق مقدار آنها از یک مقدار آستانه‌ای کمتر باشد جریمه شده و وزنهایی روشنی که قدر مطلق آنها از یک مقدار آستانه‌ای دیگر بیشتر باشد پاداش میگیرند. وزنهایی روشنی که قدر مطلق مقدار آنها بین دو مقدار آستانه‌ای قرار بگیرد تغییری در وضعیت آنها ایجاد نمیشود. در مورد وزنهایی خاموش نیز وضع بهمین منوال است وزنهایی خاموشی که قدر مطلق آنها کمتر از یک مقدار آستانه‌ای باشد پاداش میگیرند و وزنهایی خاموشی که قدر مطلق آنها بیشتر از یک مقدار آستانه‌ای باشد جریمه میشوند. وزنهایی که بین دو مقدار آستانه‌ای قرار میگیرند تغییری در وضعیت آنها صورت نمیگیرد. مقادیر آستانه‌ای که در این تصمیم‌گیرها مورد استفاده قرار میگیرد بسیار تعیین‌کننده بوده و تأثیر چشمگیری در عملکرد شبکه عصبی دارد. با انتخاب مقادیر بهینه برای این بازه‌ها میتوانیم در کمترین زمان به شبکه‌هایی با حداقل تعداد وزن که بتواند با خطای قابل قبولی آموزش را یاد گرفته و از قدرت تعمیم قابل قبولی برخوردار باشد برسیم. در این مقاله الگوریتم جدیدی تحت عنوان AWSA برای تنظیم بازه‌های تصمیم‌گیری ارائه گردیده

است. در الگوریتم پیشنهادی برای پاسخ دادن به اتوماتان مربوط به تنظیم بازه‌های تصمیم‌گیری، از معیار خطا کمک گرفته‌ایم. مدر ادامه الگوریتم جدید دیگری تحت عنوان MWSA ارائه گردیده است. در الگوریتم پیشنهادی معیارهای جدیدی برای ارزیابی عملکرد وزنه‌های روشن و خاموش ارائه گردیده است. نتایج شبیه‌سازیهای مختلف بر روی مسائل شناسایی اعداد انگلیسی، شناسایی اعداد چاپی فارسی، تقریب تابع غیر خطی زمان گسسته از مرتبه دوم و باز شناسی فونمهای فارسی نشان می‌دهد الگوریتمهای پیشنهادی AWSA و MWSA دارای عملکرد بهتری نسبت به الگوریتمهای دیگر یادگیری ساختار میباشند.

بخش‌های بعدی مقاله بصورت زیر سازماندهی شده است. در بخش ۲ الگوریتم یادگیری پس انتشار خطا که برای آموزش شبکه‌های عصبی چند لایه مورد استفاده قرار می‌گیرد توضیح داده شده است. موضوع Overtraining و قدرت تعمیم شبکه در بخش ۳ مورد بررسی قرار گرفته است. در بخش ۴ مقدمه‌ای بر اتوماتانهای یادگیر و انواع مهم آن آورده شده است. در بخش ۵ الگوریتم بقا وزن را توضیح داده‌ایم. در بخشهای ۶ و ۷ به ترتیب الگوریتمهای پیشنهادی AWSA و MWSA توضیح داده شده است. نتایج آزمایشها برای مسائل مختلف در بخش ۸ و در بخش پایانی نتیجه‌گیری شده است.

۱- الگوریتم پس انتشار خطا

الگوریتم BP یک روش سیستماتیک برای آموزش شبکه‌های عصبی چند لایه می‌باشد. الگوریتم BP بطور مبسوط در مرجع [۲] توضیح داده شده است. الگوریتم BP، دو مسیر محاسباتی دارد. مسیر اول مسیر پیشخور یا رفت و مسیر دوم مسیر پس خور یا مسیر برگشت نامیده می‌شود. مسیر رفت: این مسیر با معادلات زیر توصیف می‌شود:

$$\begin{aligned} \underline{a}^0 &= \underline{p}(k) \\ \underline{a}^{l+1}(k) &= \underline{F}^{l+1}(\underline{W}^{l+1}(k)\underline{a}^l + \underline{b}^{l+1}(k)), \quad l=0,1,\dots,L-1 \\ \underline{a} &= \underline{a}^L(k) \end{aligned} \quad (1)$$

که در آن داریم:

\underline{a}^l : بردار خروجی لایه l ام

$\underline{p}(k)$: بردار ورودی

\underline{F}^l : بردار تابع محرک لایه l ام

\underline{W}^l : ماتریس وزن لایه l ام

\underline{b}^l : بردار بایاس لایه l ام

\underline{a} : بردار خروجی شبکه عصبی

L : تعداد لایه‌های شبکه عصبی

در این مسیر وزنها و بایاسهای شبکه تغییر داده نمی‌شوند. توابع محرک، روی تمامی نرونها عمل می‌کند، یعنی:

$$\underline{F}^{l+1}(\underline{n}^{l+1}(k)) = [f^{l+1}(n_{1,l+1}^{l+1}(k)), \dots, f^{l+1}(n_{s_{l+1},l+1}^{l+1}(k))] \quad (2)$$

که در آن سملها بصورت زیر تعریف میشوند:

f^l : تابع محرک لایه l ام

$\underline{n}^l(k)$: بردار ورودی خالص لایه l ام

$n_i^l(k)$: ورودی خالص i ام لایه l

S_l : تعداد نرونهای لایه l ام

مسیر برگشت: در این مسیر بردارهای حساسیت از لایه آخر به لایه اول برگشت داده می‌شوند. معادلات زیر، دینامیک

مسیر برگشت را بیان می‌کند:

$$\begin{aligned}\underline{\delta}^l(k) &= -2F^l(\underline{n})\underline{e}(k) \\ \underline{\delta}^l(k) &= F^l(\underline{n}^l)(W^{l+1})^T \underline{\delta}^{l+1}, \quad l = L-1, \dots, 1 \\ \underline{e}(k) &= \underline{t}(k) - \underline{a}(k)\end{aligned}\quad (3)$$

که در آن داریم:

$\underline{\delta}^l(k)$: بردار گرادیان محلی لایه خروجی

$\underline{e}(k)$: بردار خطا

$\underline{\delta}^l(k)$: بردار گرادیان محلی لایه l ام

$\underline{t}(k)$: بردار هدف

در مسیر برگشت ابتدا با در دست بودن بردار هدف، بردار خطا محاسبه می‌شود. سپس بردار خطا از سمت راست به چپ و از لایه آخر به لایه اول توزیع شده، گرادیان محلی نرون به نرون با الگوریتم بازگشتی محاسبه می‌شود. تنظیم پارامترها: در این مرحله ماتریسهای وزن و بردارهای بایاس شبکه بصورت زیر تنظیم می‌شوند.

$$\begin{aligned}W^l(k+1) &= W^l(k) + \alpha \underline{\delta}^l(k) (\underline{a}^{l-1}(k))^T + \eta W^l(k-1) \\ b^l(k+1) &= b^l(k) + \alpha \underline{\delta}^l(k) + \eta b^l(k-1) \quad l = 1, 2, \dots, L\end{aligned}\quad (4)$$

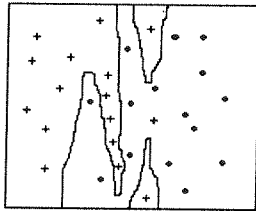
که در رابطه فوق α و η به ترتیب نرخ یادگیری و ضریب ممنت می‌باشد.

توقف: اگر میانگین مربعات خطا در هر epoch (جمع مربعات خطا برای تمامی الگوهای یادگیری) کمتر از مقدار آستانه کوچکی بوده و یا اینکه فرم تغییرات در پارامترهای شبکه پس از هر سیکل خیلی کوچک باشد الگوریتم BP متوقف می‌شود.

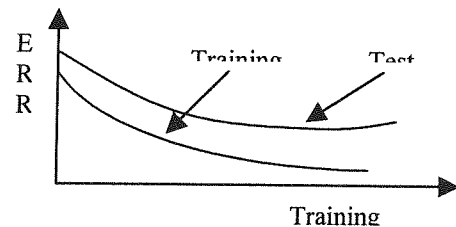
۲- Overtraining و قدرت تعمیم

موقعیکه یک شبکه عصبی آموزش می‌بیند، وزنها طوری تغییر داده میشوند که خطا کاهش یابد. اگر شبکه توسط الگوهایی که فقط اختلاف جزئی با الگوهای آموزش دارند تست شود خطا در الگوهای تست همچنانکه شبکه آموزش می‌بیند و قدرت تعمیم آن بیشتر میشود کاهش پیدا میکند. این مسئله در شکل (۱) نشان داده شده است. اگر الگوهای آموزش ناکافی باشد در اینصورت محتوی داده‌ها نادرست و همراه‌کننده خواهد بود. در بعضی نقاط، بویژه در مراحل بعدی یادگیری، داده‌های ناکافی تأثیر خود را نشان داده و با اینکه خطای آموزش کاهش می‌یابد خطای آزمایش شروع به افزایش میکند [۲۸]. یک روش برای پرهیز از Overfitting این است که توانائی تعمیم را بهنگام آموزش تخمین زده و زمانیکه شروع به کاهش میکند آموزش را قطع نماییم. ساده‌ترین روش این است که الگوها را به دو گروه الگوهای آموزش و الگوهای تصدیق تقسیم کنیم. الگوهای آموزش برای تغییر دادن وزنها استفاده میشود و الگوهای تصدیق برای تخمین توانائی تعمیم استفاده شده و آموزش زمانی متوقف می‌شود که خطا روی الگوهای تصدیق شروع به افزایش کند. این روش موقعیکه الگوهای کمی برای آموزش در اختیار می‌باشد بدلیل اینکه الگوهای تصدیق نمی‌تواند برای آموزش مورد استفاده قرار گیرد عملی نمی‌باشد [۳۱] - [۲۹]. یک راه دیگر برای دوری جستن از Overtraining این است که توانایی شبکه در بهره‌گیری از همبستگیهای نادرست در الگوها را محدود کنیم. Overtraining موقعی اتفاق می‌افتد که درجه آزادی شبکه (تعداد وزنها) بیشتر از الگوهای آموزش باشد. با اینکه شبکه با اعمال الگوهای آموزش جواب دقیقی را بدست میدهد ولی ممکن است در نقاط دیگر خیلی بد عمل کند. شبکه‌های کوچک در کنار قدرت تعمیم بالا مزیت‌های دیگری را نیز بهمراه دارند. معمولاً اینها را میتوان خیلی سریع و با هزینه کمتری پیاده‌سازی کرد. تئوریها و فرمولهایی برای تخمین اندازه مورد نیاز سیستم، مورد استفاده قرار گرفته است [۳۴] - [۳۲]. این روشها پیچیدگی سیستم یادگیری و تعداد الگوهای مورد نیاز برای یادگیری یک تابع را بهم ربط میدهند. اگر تعداد الگوها نسبت به پیچیدگی سیستم کوچک باشد. در اینصورت انتظار داریم خطای تعمیم زیاد باشد. شکلهای ۲ و ۳ تأثیر شاخه زنی را نشان میدهد. شکل ۲ مرزهای تشکیل شده توسط یک شبکه ۴ لایه با دو ورودی، ۲ لایه مخفی با ۵۰ نرون در لایه مخفی اول و ۱۰ نرون در لایه

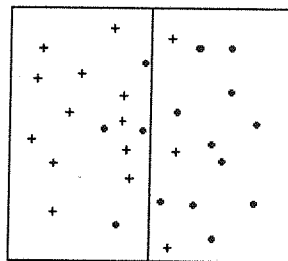
مخفی دوم و یک خروجی را نشان می‌دهد. شبکه دارای ۶۷۱ وزن بوده اما فقط ۳۱ الگو برای آموزش وجود دارد. بنابر این تعداد پارامترهای شبکه در مقایسه با تعداد الگوها خیلی زیاد می‌باشد. گرچه داده‌ها تقریباً بطور خطی جدا پذیرند (با کمی همپوشانی در نزدیکی مرز) با اینحال مرز طبقه‌بندی شبکه خیلی غیر خطی بوده و احتمالاً قدرت تعمیم خوبی روی داده‌های اضافه دیگر از همان تابع، نخواهد داشت. شکل ۳ همان شبکه را پس از شاخه زنی نشان می‌دهد. اندازه شبکه به ۲/۲/۲/۱ با ۱۵ وزن تقلیل پیدا کرده و مرز تصمیم‌گیری بسیار یکنواخت می‌باشد. روشهای پیچیده‌تر می‌تواند شبکه را به یک شبکه ساده فقط با ۲ وزن تقلیل دهد. مثال همچنان نشان می‌دهد که روش شاخه زنی می‌تواند بعنوان ابزاری برای انتخاب ویژگی مورد استفاده قرار گیرد. اگر ورودیهای خاصی برای مسئله مهم نباشد الگوریتم، اتصالات مربوط به آنرا قطع خواهد کرد.



شکل (۲) مرزهای تصمیم‌گیری مربوط به شبکه هرس نشده.



شکل (۱) تغییرات خطای مربوط به الگوهای آزمایشی و آموزشی.



شکل (۳) مرز تصمیم‌گیری مربوط به شبکه هرس شده.

۳- اتوماتان یادگیر

اتوماتان یادگیر^۱ (LA) را می‌توان به دو گروه اصلی اتوماتان یادگیر با ساختار ثابت^۲ (FSLA) و اتوماتان یادگیر با ساختار متغیر^۳ (VSLA) تقسیم کرد [۳۸] - [۳۵]. اگر احتمال انتقال از یک حالت به حالت دیگر و احتمالهای اقدام و حالت ثابت باشند، اتوماتان با ساختار ثابت، در غیر این صورت اتوماتان با ساختار متغیر نامیده می‌شود. بعضی از انواع FSLA عبارتند از اتوماتانهای ستلین^۴، کرینسکی^۵ و کرایلف^۶. اتوماتان یادگیر با ساختار ثابت یک پنج تایی بصورت $\langle \underline{\alpha}, \underline{\Phi}, \underline{\beta}, F, G \rangle$ می‌باشد که در آن داریم:

$\alpha = (\alpha_1, \dots, \alpha_R)$ مجموعه اقدامهاست که در هر مرحله یکی از این اقدامها انتخاب می‌شود.

$\Phi = (\Phi_1, \dots, \Phi_r)$ مجموعه حالتهاست.

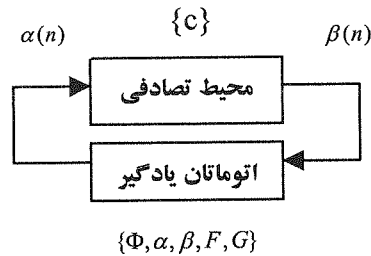
$\beta = \{0,1\}$ مجموعه ورودیهاست که در آن ۱ جریمه و ۰ پاداش را نشان می‌دهد.

$F: \Phi \times \beta \rightarrow \Phi$ نگاشت انتقال نامیده می‌شود. این نگاشت، انتقال حالتی اتوماتان پس از دریافت ورودی را تعریف می‌کند.

F می‌تواند استوکستیک^۷ باشد.

$G: \Phi \rightarrow \alpha$ نگاشت خروجی بوده و اقدام اتوماتان در حالت Φ را نشان می‌دهد.

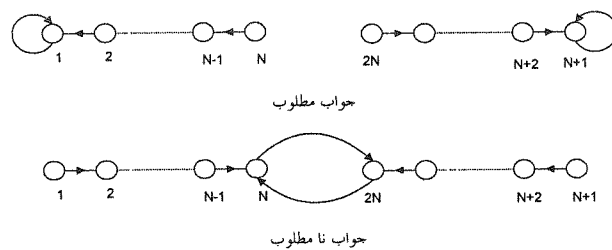
اقدام انتخاب شده بعنوان ورودی به محیط اعمال شده و محیط در واکنش به آن پاسخ تصادفی $\beta(n)$ را در لحظه n تولید می کند. $\beta(n)$ عنصری از $\beta = \{0,1\}$ بوده و پاسخ فیدبک محیط، به اتوماتان می باشد. محیط اتوماتان را با احتمال c_i جریمه کرده ($\beta(n)=1$) که این احتمال وابسته به اقدام می باشد. بر اساس پاسخ $\beta(n)$ ، حالت اتوماتان $\Phi(n)$ تغییر یافته و اقدام جدیدی در لحظه $n+1$ انتخاب می شود. نحوه اتصال اتوماتان و محیط در شکل (۴) نشان داده شده است.



شکل (۴) نحوه اتصال محیط با اتوماتان.

۱-۳- اتوماتان ستلین ($L_{2N,2}$)

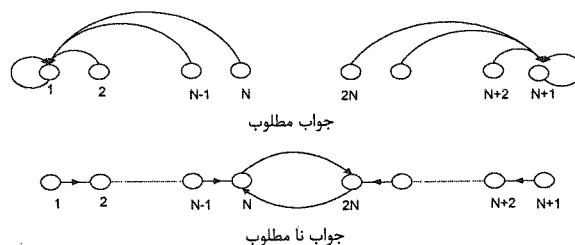
این اتوماتان دارای $2N$ حالت و دو اقدام بوده و رفتار گذشته سیستم را در قانون تصمیم گیری برای انتخاب رشته اقدام دخالت میدهد. اتوماتان $L_{2N,2}$ وقتی که در داخلی ترین حالت قرار گرفته باشد برای انتقال به اقدام دیگر نیاز به N پاسخ نا مطلوب متوالی دارد. N عمق حافظه اقدام نامیده شده و می گوئیم اتوماتان دارای تعداد $2N$ حافظه می باشد. برای هر پاسخ مطلوب، حالت اتوماتان به سمت حالت های داخلی تر حرکت می کند و به ازای پاسخ نامطلوب به سمت حالت های بیرونی تر میل میکند. گراف انتقال حالت اتوماتان $L_{2N,2}$ در شکل (۵) نشان داده شده است.



شکل (۵) گراف انتقال حالت برای $L_{2N,2}$.

۲-۳- اتوماتان کرینسکی

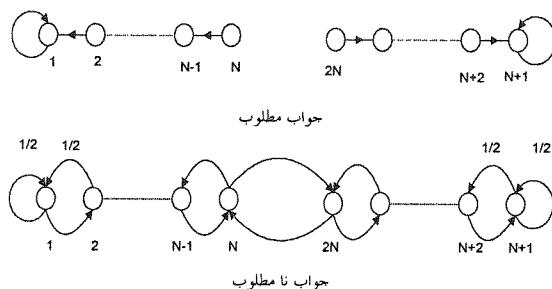
این اتوماتان موقعیکه پاسخ محیط نامطلوب باشد. دقیقاً مثل $L_{2N,2}$ رفتار می کند. اما برای پاسخ مطلوب، هر حالت Φ_i (برای $i=1, \dots, N$) به حالت Φ_1 می رود و هر حالت Φ_i (برای $i=N+1, \dots, 2N$) به حالت Φ_{N+1} می رود. لذا پس از هر پاسخ مطلوب تعداد N پاسخ نامطلوب لازم است تا اتوماتان از یک اقدام به اقدام دیگر سوئیچ کند. گراف انتقال حالت اتوماتان کرینسکی در شکل (۶) نشان داده شده است.



شکل (۶) گراف انتقال حالت برای اتوماتان کرینسکی.

۳-۳- اتوماتان کرایلف

این اتوماتان موقعیکه خروجی محیط مطلوب باشد انتقال حالت مشابه $L_{2N,2}$ دارد. ولی موقعیکه پاسخ محیط نامطلوب باشد حالت Φ_i با احتمال $0/5$ به حالت Φ_{i+1} و با احتمال $0/5$ به حالت Φ_{i-1} می‌رود. موقعیکه $i = 1$ یا $i = N+1$ می‌باشد، با احتمال $0/5$ در همان حالت باقی می‌ماند. و با همان احتمال به حالت Φ_{i+1} می‌رود. موقعیکه $i = N$ می‌باشد اتوماتان با احتمال $0/5$ به حالت $2N$ می‌رود و با همان احتمال به حالت $N-1$ می‌رود و بالاخره موقعیکه $i = 2N$ می‌باشد اتوماتان با احتمال $0/5$ به حالت N و با همان احتمال به حالت $2N-1$ می‌رود. گراف انتقال حالت این اتوماتان در شکل (۷) نشان داده شده است.



شکل (۷) گراف انتقال حالت برای اتوماتان کرایلف.

۳-۴- اتوماتان با ساختار متغیر

اتوماتان با ساختار متغیر، احتمالهای انتقال یا احتمالهای اقدام را بر اساس ورودی تغییر می‌دهد. این نوع اتوماتان توسط شش تایی $\langle \beta, \Phi, \alpha, p, G, T \rangle$ نمایش داده می‌شود. که در آن β مجموعه ورودیها، Φ مجموعه حالت‌های داخلی، α مجموعه خروجیها و p بردار احتمال اقدامهاست. که بر اساس آن در هر مرحله اقدامی انتخاب می‌شود. G نگاشت خروجی و T الگوریتم یادگیری را نشان می‌دهد. الگوریتم یادگیری یک رابطه برگشتی بوده و برای تغییر دادن بردار احتمال مورد استفاده قرار می‌گیرد. الگوریتمهای یادگیری مختلفی در مرجع [۳۵] گزارش شده است. الگوریتم یادگیری پاداش - جریمه خطی L_{R-P} جزء اولین روشها می‌باشد. فرض کنیم در لحظه k اقدام α_i از مجموعه اقدامها با توجه به توزیع $p(k)$ انتخاب شود. در الگوریتم L_{R-P} معادله برگشتی برای تغییر p به صورت زیر تعریف می‌شود.

$$\beta(n)=0 \quad \text{پاسخ مطلوب} \quad \begin{aligned} p_i(n+1) &= p_i(n) + a[1 - p_i(n)] \\ p_j(n+1) &= (1-a)p_j(n) \quad j \neq i \end{aligned} \quad (5)$$

$$\beta(n)=1 \quad \text{پاسخ نامطلوب} \quad \begin{aligned} p_i(n+1) &= (1-b)p_i(n) \\ p_j(n+1) &= b/(r-1) + (1-b)p_j(n) \quad j \neq i \end{aligned} \quad (6)$$

پارامترهای a و b طول گام را نشان داده، مقدار افزایش و کاهش احتمال اقدامها را تعیین می‌کنند. الگوریتم یادگیری دیگری که اکثراً مورد استفاده قرار می‌گیرد الگوریتم پاداش - بیحرکت خطی L_{R-1} می‌باشد. در L_{R-1} به ازای پاسخ مطلوب $\beta(n)=0$ احتمال متناظر با اقدام α_i افزایش یافته و بقیه احتمالات کاهش می‌یابند. ولی به ازای پاسخ نامطلوب $\beta(n)=1$ هیچگونه تغییری در احتمالات صورت نمی‌گیرد. معادله برگشتی برای تغییر P به صورت زیر می‌باشد.

$$\beta(n)=0 \quad \text{پاسخ مطلوب} \quad \begin{aligned} p_i(n+1) &= p_i(n) + a[1 - p_i(n)] \\ p_j(n+1) &= (1-a)p_j(n) \quad j \neq i \end{aligned} \quad (7)$$

$$\beta(n)=1 \quad \text{پاسخ نامطلوب} \quad p_j(n+1) = p_j(n) \quad 1 \leq j \leq r \quad (8)$$

۳-۵- اتوماتان مهاجرت اشیا

اتوماتان مهاجرت اشیا [۴۰] توسط پنج تایی $\langle \alpha, \Phi, \beta, F, G \rangle$ نشان داده میشود. که $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_r\}$ مجموعه اقدامهای مجاز، $\Phi = \{\Phi_1, \Phi_2, \dots, \Phi_r\}$ مجموعه وضعیتها، $\beta \in \{0, 1\}$ مجموعه ورودیها، $F: \Phi \times \beta \rightarrow \Phi$ تابع نگاشت وضعیتها و $G: \Phi \rightarrow \alpha$ تابع نگاشت خروجی اتوماتان میباشد. این نوع اتوماتان برای دسته‌بندی اشیا، و تعیین تعداد واحدهای مخفی شبکه‌های عصبی [۴۲]، [۴۱] مورد استفاده قرار گرفته است. در این اتوماتان هر اقدام یک دسته از الگوها را نشان میدهد. در اتوماتان مهاجرت اشیا، اشیا به وضعیت‌های اتوماتان انتساب داده میشوند و پاسخ محیط به اتوماتان سبب گردش اشیا در بین وضعیت‌های اتوماتان میگردد. از طریق این گردش طبقه‌بندی اشیا صورت میگیرد. اگر شیء W_i در اقدام شماره i اتوماتان مهاجرت اشیا قرار داشته باشد. این شیء متعلق به دسته شماره α_k مجموعه وضعیت $\{\Phi_{(k-1)N+1}, \dots, \Phi_{kN}\}$ در نظر گرفته میشود. که N عمق حافظه را نشان میدهد. $\Phi_{(k-1)N+1}$ داخلی ترین وضعیت و Φ_{kN} خارجی ترین وضعیت این اقدام میباشد. اگر دو شیء W_i و W_j بترتیب در وضعیت‌های $\Phi_{(k-1)N+m}$ و $\Phi_{(k-1)N+m}$ (برای $m > 1$) قرار داشته باشند در اینصورت احتمال تعلق شیء W_i به این دسته از احتمال تعلق شیء W_j بیشتر است.

الگوریتم تعیین تعداد وزن WSA

در این قسمت ابتدا بطور مختصر الگوریتم کاهش تعداد وزن ارائه شده در مرجع [۴۳] را شرح داده در ادامه، الگوریتم‌های جدیدی را ارائه میکنیم. در این الگوریتم از یک اتوماتان مهاجرت اشیا استفاده شده است. این اتوماتان برای تعیین تعداد وزنهاى لایه مخفی بوده و دارای دو اقدام و عمق حافظه N میباشد. روی وضعیت‌های آن مجموعه وزن $W = \{w_1, w_2, w_3, \dots\}$ قرار دارد. گراف تغییر حالت برای این اتوماتان در شکل ۸ نشان داده شده است. این الگوریتم نحوه اتصال وزنهاى بین لایه ورودی و لایه مخفی را مشخص میکند. در ابتدا همه وزنهاى موجود در شبکه در وضعیت Φ_1 قرار میگیرند. به همه وزنهاى روشن مدتی اجازه داده میشود تا در آموزش شبکه شرکت نمایند. وزنهاى که عملکرد آنها در این مدت مناسب نیست جریمه میشوند، وزنهاى که عملکرد آنها خوب است پاداش میگیرند و وزنهاى که در مورد آنها نمیتوان تصمیم‌گیری قطعی نمود بدون تغییر وضعیت باقی می‌مانند. برای تشخیص خوب یا بد بودن عملکرد یک وزن از قوانین زیر استفاده میشود.

عملکرد خوب یک وزن: وزنی دارای عملکرد خوب میباشد که سیگنال وارد شده به این وزن بمیزان زیادی تضعیف نگردد و یا به عبارتی دیگر قدر مطلق آن کوچک نباشد.

عملکرد بد یک وزن: وزنی دارای عملکرد بد میباشد اگر سیگنال وارد شده به این وزن بمیزان زیادی تضعیف گردد و یا به عبارتی دیگر قدر مطلق آن کوچک باشد.

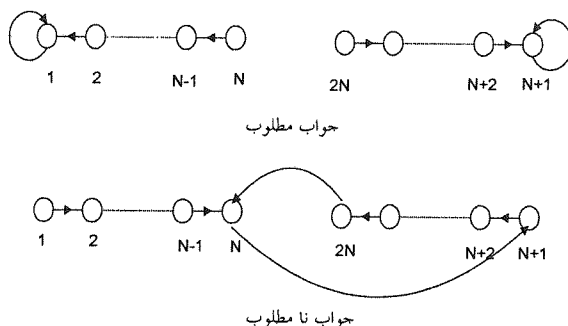
تشخیص نحوه عملکرد یک وزن روشن: الگوریتم پس انتشار خطا را به دفعات معینی تکرار میکنیم در انتها اگر قدر مطلق وزنی از یک مقدار آستانه‌ای بیشتر باشد آن وزن مناسب بوده پاداش میگیرد و اگر از یک مقدار آستانه‌ای دیگر کوچک باشد نامناسب بوده جریمه میشود. و در صورتیکه بین دو مقدار آستانه‌ای قرار گیرد تغییری در وضعیت آن ایجاد نمیشود (شکل ۹). برای تعیین مقادیر آستانه‌ای از واریانس قدر مطلق مقدار وزنها استفاده شده است. برای وزنهاى مربوط به هر کدام از نرونها، واریانس بصورت زیر محاسبه میگردد.

$$\sigma_h = \sqrt{\frac{\sum_{W_{hk} \in ON_W(h)} (W_{hk} - \mu_W(h))^2}{|ON_W(h)|}} \quad h \in ON \quad (9)$$

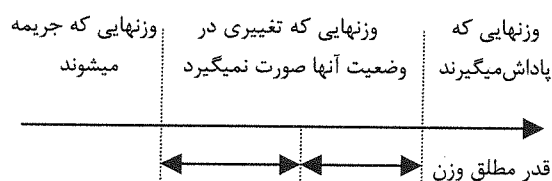
که در آن ON مجموعه واحدهای روشن، $ON_W(h)$ مجموعه وزنهاى ورودی روشن متصل به واحد مخفی h و $\mu_W(h)$ میانگین قدر مطلق مقدار وزنهاى روشن متصل به واحد مخفی h میباشد. $\mu_W(h)$ پهنای روشنی نامیده شده و بصورت زیر محاسبه میگردد.

$$X_{ON}(h) = \lambda_{ON} \frac{OFF_w(h)}{|\#INPUT + 1|} \sigma_h \quad (10)$$

ثابت $\lambda_{ON} > 0$ ضریب پهنای روشنی، $\#INPUT$ تعداد واحدهای لایه ورودی و $OFF_w(h)$ مجموعه وزنه‌های خاموش متصل به واحد مخفی روشن h است.



شکل (۸) اتوماتان مربوط به الگوریتم WSA.



شکل (۹) مقدار آستانه مربوط به وزنه‌های روشن.

تشخیص نحوه عملکرد یک وزن خاموش: یک وزن خاموش در آموزش شرکت نمیکند. بنابر این مقدار آنرا از آخرین زمان روشن بودن آن محاسبه میکنیم. اگر یک وزن برای مدت طولانی خاموش باشد از اهمیت و در نتیجه از مقدار آن کاسته میشود. بنابر این مقدار یک وزن خاموش بصورت زیر محاسبه میشود.

$$W_{hk}(n+1) = W_{hk}(n) \lambda_d \quad (11)$$

که ثابت $0 < \lambda_d < 1$ ضریب کاهش مقدار وزن نامیده میشود. به این ترتیب مقدار یک وزن خاموش بتدریج کم میشود. واریانس وزنه‌های خاموش بصورت زیر محاسبه میشود.

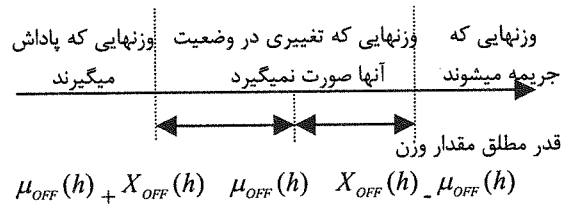
$$\sigma_h = \sqrt{\frac{\sum_{W_{hk} \in OFF_w(h)} (W_{hk} - \mu_{OFF}(h))^2}{|OFF_w(h)|}} \quad h \in ON \quad (12)$$

که در آن ON مجموعه واحدهای روشن و $\mu_{OFF}(h)$ میانگین قدر مطلق مقدار وزنه‌های خاموش متصل به واحد مخفی روشن h میباشد. با استفاده از واریانس و طبق فرمول زیر $X_{OFF}(h)$ که پهنای خاموشی نامیده میشود بصورت زیر محاسبه میگردد.

$$X_{OFF}(h) = \lambda_{OFF} \frac{ON_w(h)}{|\#INPUT + 1|} \sigma_h \quad (13)$$

که ثابت $0 < \lambda_{OFF} < 1$ ضریب پهنای خاموشی و $\#INPUT$ تعداد واحدهای لایه ورودی است. وزنه‌هایی که قدر مطلق مقدار آنها از $\lambda_{OFF} \times \sigma_h$ کمتر باشد پاداش میگیرند. وزنه‌هایی که قدر مطلق آنها از $\lambda_{OFF} \times \sigma_h$ بیشتر باشند جریمه

میشوند و وزنهایی که قدر مطلق آنها در فاصله $[\mu_{OFF}(h) + X_{OFF}(h), \mu_{OFF}(h) - X_{OFF}(h)]$ قرار دارند، وضعیت آنها تغییر نمیکنند (شکل ۱۰).



شکل (۱۰) مقدار آستانه مربوط به وزنهایی خاموش.

این الگوریتم را برای راحتی در مراجعات بعدی بطور اختصار با عبارت WSA نشان میدهم.

ارائه الگوریتم بقا وزن تطبیقی AWSA

مقدار $X_{ON}(h)$ که بازه تصمیم گیری را نشان میدهد تأثیر بسزایی در عملکرد الگوریتم کاهش تعداد وزن دارد. وزنهایی روشن که قدر مطلق آنها کمتر از $\mu_{ON}(h) - X_{ON}(h)$ می باشد جرمه شده و وزنهایی روشن که قدر مطلق آنها بیشتر از $\mu_{ON}(h) + X_{ON}(h)$ می باشد پاداش میگیرند. هر قدر مقدار بازه $X_{ON}(h)$ کم باشد وزنهایی بیشتری جرمه شده در نتیجه وزنها با سرعت بیشتری ریزش خواهند کرد. اگر این سرعت از یک حدی بیشتر باشد شبکه دیگر قادر به یادگیری الگوهای آموزشی نخواهد بود. اگر مقدار $X_{ON}(h)$ خیلی زیاد انتخاب شود شرط جرمه کردن وزنهایی روشن خیلی سنگین شده، در نتیجه وزنی خاموش نشده یا تعداد بسیار کمی وزن خاموش خواهد شد. در نتیجه شاهد هیچگونه کاهش در پیچیدگی شبکه نخواهیم بود. پس ملاحظه میکنیم مقدار $X_{ON}(h)$ خیلی مهم بوده و با انتخاب یک مقدار بهینه برای $X_{ON}(h)$ میتوانیم در کمترین زمان به شبکه‌هایی با حداقل تعداد وزن که بتواند با خطای قابل قبولی نیز الگوهای آموزش را یاد بگیرد برسیم. بحثهایی که در مورد بازه $X_{ON}(h)$ مطرح کردیم در مورد بازه $X_{OFF}(h)$ نیز قابل طرح میباشد. $X_{OFF}(h)$ هم که بازه تصمیم گیری برای وزنهایی خاموش را نشان میدهد در عملکرد الگوریتم کاهش تعداد وزن بسیار موثر میباشد. وزنهایی خاموشی که قدر مطلق آنها کمتر از $\mu_{OFF}(h) - X_{OFF}(h)$ می باشد پاداش میگیرند و وزنهایی خاموشی که قدر مطلق آنها بیشتر از $\mu_{OFF}(h) + X_{OFF}(h)$ می باشد جرمه میشوند. هر قدر مقدار بازه $X_{OFF}(h)$ کم باشد وزنهایی خاموش بیشتری جرمه شده در نتیجه وزنهایی خاموش با سرعت بیشتری شروع به روشن شدن میکنند و اگر مقدار $X_{ON}(h)$ خیلی زیاد انتخاب شود شرط جرمه کردن وزنهایی خاموش خیلی سنگین شده، در نتیجه وزنی روشن نشده یا تعداد بسیار کمی وزن روشن میشوند. در نتیجه آنجائیکه نیاز داریم وزنهایی به شبکه برگردانده شوند وزنی روشن نمیشود. پس ملاحظه میکنیم مقدار $X_{OFF}(h)$ نیز خیلی مهم بوده و با انتخاب یک مقدار بهینه برای $X_{OFF}(h)$ میتوانیم در کمترین زمان به شبکه‌هایی با حداقل تعداد وزن که بتواند با خطای قابل قبولی نیز الگوهای آموزش را یاد بگیرد برسیم. همچنانکه در مبحث مربوط به الگوریتم WSA دیدیم پارامتر $X_{ON}(h)$ براساس فرمول زیر تابع چندین متغیر می باشد.

$$X_{ON}(h) = \lambda_{ON} \frac{OFF_W(h)}{|\#INPUT + 1|} \sigma_{ON}(h) \quad (14)$$

#INPUT بعد فضای ورودی بوده و برای یک مسئله خاص مقدار ثابتی می باشد. $OFF_W(h)$ تعداد وزنهایی خاموش را نشان داده و بعنوان یک عامل بازدارنده در مقابل ریزش بیش از حد وزنها عمل میکند. با حذف تدریجی وزنها مقدار $OFF_W(h)$ افزایش یافته و باعث افزایش $X_{ON}(h)$ میگردد. با افزایش $X_{ON}(h)$ از سرعت ریزش وزنها کاسته میشود. λ_{ON} مقدار ثابتی بوده و ضریب پهنای روشنی نامیده میشود. واریانس وزنهایی روشن مربوط به نرون مخفی h میباشد. پارامتر $X_{OFF}(h)$ نیز براساس فرمول زیر تابع چندین متغیر می باشد.

$$X_{OFF}(h) = \lambda_{OFF} \frac{ON_W(h)}{|\#INPUT + 1|} \sigma_{OFF}(h) \quad (14)$$

#INPUT بعد فضای ورودی بوده و برای یک مسئله خاص مقدار ثابتی می‌باشد. $ON_W(h)$ تعداد وزنهای روشن را نشان داده و بعنوان یک عامل بازدارنده در مقابل برگشت بیش از حد وزنها عمل میکند. λ_{OFF} مقدار ثابتی بوده و ضریب پهنای خاموشی نامیده میشود. $\sigma_{OFF}(h)$ واریانس وزنهاى خاموشن مربوط به نرون مخفی h میباشد. پس ملاحظه میکنیم بر اساس فرمولهای ۱۴ و ۱۵ بازه های تصمیم‌گیری X_{OFF} و X_{ON} به ترتیب متناسب با ضرائب λ_{ON} و λ_{OFF} بوده و مقادیر این ضرائب میتواند در فرایند تعیین تعداد وزنها بسیار تعیین‌کننده باشد. برای بررسی تجربی نقش ضرائب λ_{ON} و λ_{OFF} بر روی عملکرد الگوریتم WSA آزمایشی را بصورت زیر ترتیب میدهم. برای انجام آزمایش مسئله شناسائی اعداد لاتین را در نظر میگیریم. تأثیر λ_{ON} و λ_{OFF} را بطور مجزا مورد بررسی قرا میدهم. ابتدا مقدار λ_{OFF} را ثابت در نظر گرفته مقدار λ_{ON} را تغییر میدهم.

مقدار λ_{OFF} را ۱۰ در نظر گرفته و به ازای مقادیر مختلف λ_{ON} آزمایش را تکرار میکنیم. در این آزمایش ضریب کاهش وزن ۰/۹ و نرخ یادگیری ۰/۰۱ انتخاب شده است. همچنین در هر گام الگوریتم پس انتشار خطا ۵۰ بار تکرار شده است. از تابع غیر خطی سیگموئید بعنوان توابع لایه مخفی و لایه خروجی استفاده شده است. نتایج آزمایشهای مختلف در جدول ۱ نشان داده شده است.

جدول (۱) بررسی تأثیر مقادیر مختلف λ_{ON} بر روی سرعت شاخه زنی و همچنین همگرایی شبکه به ازای $\lambda_{OFF} = 10$

Epoch \ λ_{ON}	-	۰	۱	۱۵	۲	۲۵	۳	۳۵	۴	۴۵	۵	خطای آموزش
۱۰	۳۸۴۰	۳۸۱۰	۳۱۰۰	۲۷۰۳	۲۰۰۱	۱۴۰۳	۱۰۳۰	۷۰۳	۵۵۳	۵۰۴	۴۹۹	۰/۰۱
۵	۳۸۴۰	۳۲۰۳	۲۸۴۰	۲۲۴۵	۱۹۴۰	۱۲۴۰	۸۰۳	۵۰۳	۴۳۰	۳۰۴	۲۵۰	۰/۰۱
۲	۳۸۴۰	۳۱۰۰	۲۵۰۳	۲۱۰۳	۱۸۳۹	۱۰۲۳	۴۷۶	۳۷۵	۲۷۴	۱۲۰	۹۵	۲/۰۵
۰/۱	۳۸۴۰	۳۰۳۰	۲۳۰۴	۲۰۰۴	۱۷۴۷	۹۹۵	۳۹۵	۲۵۸	۱۵۰	۹۴	۸۵	۲/۱۵

جدول (۲) بررسی تأثیر مقادیر مختلف λ_{OFF} بر روی سرعت شاخه زنی و همچنین همگرایی شبکه به ازای $\lambda_{ON} = 10$

Epoch \ λ_{OFF}	-	۰	۱	۱۵	۲	۲۵	۳	۳۵	۴	۴۵	۵	خطای آموزش
۱۰	۳۸۴۰	۳۸۱۰	۳۱۰۰	۲۷۰۳	۲۰۰۱	۱۴۰۳	۱۰۳۰	۷۰۳	۵۵۳	۵۰۴	۴۹۹	۰/۰۱
۲	۳۸۴۰	۳۸۲۳	۳۳۰۳	۲۹۶۴	۲۱۰۳	۱۷۳۰	۱۲۰۳	۷۸۰	۶۷۳	۶۰۱	۵۴۰	۰/۰۱
۰/۵	۳۸۴۰	۳۸۲۰	۳۶۴۳	۳۵۲۰	۲۳۹۴	۱۸۴۵	۱۳۸۴	۸۴۲	۷۳۵	۶۹۳	۶۷۰	۰/۰۱
۰/۱	۳۸۴۰	۳۸۲۵	۳۷۸۴	۳۶۴۸	۲۴۸۹	۱۹۳۴	۱۴۸۶	۱۳۷۵	۱۲۹۴	۱۰۲۳	۹۰۰	۰/۰۱

همچنانکه در این جدول نشان داده شده است با کاهش λ_{ON} ، پهنای بازه روشنی کاهش یافته در نتیجه وزنهاى زیادی جرمه شده و نهایتاً عمل شاخه زنی با سرعت بیشتری انجام میگردد. اگر فرایند حذف وزنها با نرخ بسیار بالایی صورت گرفته و وزنهاى بیشتری از شبکه حذف شود. در اینحالت شبکه عصبی بیش از حد کوچک شده و قادر به یادگیری الگوهای آموزش

نخواهد بود. همچنانکه در جدول ۱ نشان داده شده است به ازای $\lambda_{ON} = 2$ و $\lambda_{ON} = 0.1$ همین حالت اتفاق می‌افتد.

در مرحله بعد با هدف بررسی تأثیر λ_{OFF} بر روی فرایند حذف وزن و همچنین همگرایی شبکه، آزمایشی را بصورت زیر ترتیب میدهیم. در این آزمایش نیز مسئله شناسایی اعداد لاتین را در نظر میگیریم. برای مطالعه تأثیرات λ_{ON} ، λ_{OFF} را ثابت و مساوی ۱۰ انتخاب میکنیم. بقیه شرایط آزمایش مشابه آزمایش قبل میباشد. جدول ۲ نتایج آزمایشها را نشان میدهد. با دقت در جدول ۲ مشاهده میکنیم که با کاهش λ_{OFF} ، شرط جریمه شدن وزنهاى خاموش سبکتر شده در نتیجه وزنهاى بیشتری روشن شده و نهایتاً درصد حذف وزنها کم شده و کاهش مطلوبی در پیچیدگی شبکه خواهیم داشت. پس ملاحظه کردیم سرعت عمل شاخه زنی وابسته به پارامتر λ_{ON} و سرعت عمل برگشت وزن تابعی از پارامتر λ_{OFF} می‌باشد و با تطبیق آنها میتوان سرعت عمل شاخه زنی و سرعت روشن شدن وزنهاى خاموش را کنترل کرد. برای دستیابی به شبکه‌هایی با کمترین پیچیدگی و در کمترین زمان، نیاز به الگوریتمهایی داریم که بتواند بطور پویا سرعت عمل شاخه زنی و برگشت وزن را بر اساس معیاری تغییر دهد. در این مقاله با استفاده از اتوماتانهای یادگیر ضرائب λ_{ON} و λ_{OFF} را تطبیق داده‌ایم. الگوریتم پیشنهادی بصورت زیر عمل میکند. در ابتدا تمامی وزنهاى موجود در شبکه در وضعیت Φ_1 قرار میگیرند. به همه وزنهاى روشن مدتی اجازه داده میشود تا در آموزش شبکه شرکت نمایند. وزنهاى که عملکرد آنها در این مدت مناسب نیست جریمه میشوند، وزنهاى که عملکرد آنها خوب است پاداش میگیرند و وزنهاى که در مورد آنها نمیتوان تصمیم‌گیری قطعی نمود بدون تغییر وضعیت باقی میمانند.

عمل شاخه زنی در ابتدا با یک سرعت مشخصی شروع میشود اگر سرعت شاخه زنی مناسب بوده و منجر به افزایش خطای یادگیری نگردد اتوماتان مربوط به تنظیم مقادیر λ_{ON} و λ_{OFF} پاداش میبینند در صورتیکه سرعت شاخه زنی بیش از حد زیاد بوده و منجر به واگرایی الگوریتم پس انتشار خطا گردد در این صورت اتوماتان مربوط به تنظیم ضرائب λ_{ON} و λ_{OFF} جریمه میشود. برای پاسخ دادن به اتوماتان مربوط به تنظیم ضرائب λ_{ON} و λ_{OFF} از معیار خطا کمک میگیریم. مینیمم مقدار خطا در تکرار فعلی را با مینیمم مقدار خطا در تکرار قبلی مقایسه میکنیم اگر با یک ضریبی خطا افزایش پیدا کرده بود اتوماتان را جریمه میکنیم در غیر اینصورت به آن پاداش میدهیم. برای تطبیق هر کدام از پارامترهای λ_{ON} و λ_{OFF} از اتوماتانهای مختلف ستلین، کرینسکی، کرایلف، L_{R-P} و L_{R-I} استفاده کرده ایم. الگوریتم جدید را که تحت عنوان الگوریتم بقا وزن تطبیقی (AWSA) نامگذاری می‌کنیم در شکل ۱۱ نشان داده شده است.

ارائه الگوریتم بقا وزن بهبود یافته MWSA

همچنانکه قبلاً در مورد الگوریتم تعیین وزن توضیح دادیم. این الگوریتم در ابتدا با ارزیابی عملکرد وزنها به تدریج وزنهاى با اهمیت کم را خاموش میکند. با پیشرفت روند آموزش و با خاموش کردن وزنهاى غیرمهم به تدریج از تعداد وزنهاى روشن کاسته شده در نتیجه با توجه به فرمول ۱۳ مقدار X_{OFF} کم میشود. با کاهش مقدار X_{OFF} ، وزنهاى خاموش بیشتری جریمه و در نتیجه روشن میشوند. ملاحظه می‌کنیم که خاموش شدن وزنهاى روشن خود عاملی برای روشن شدن وزنهاى خاموش شده در نتیجه در نهایت به یک نقطه تعادل می‌رسیم. همین وضعیت را بهنگام روشن شدن وزنهاى خاموش نیز شاهد هستیم. موقعیکه تعدادی وزن روشن میشود از تعداد وزنهاى خاموش کاسته شده در نتیجه با توجه به فرمول ۱۰ مقدار X_{ON} کم میشود. با کم شدن مقدار X_{ON} شرط جریمه شدن وزنهاى روشن سهلتر شده در نتیجه وزنهاى روشن بیشتری خاموش میشود. در این حالت نیز شاهد یک فیدبک منفی هستیم. بدین معنا که روشن شدن وزنهاى خاموش خود عاملی میشود برای خاموش کردن وزنهاى روشن. بخاطر وجود همین فیدبک منفی در طی فرایند آموزش، ساختار شبکه به تدریج به یک نقطه تعادل میل میکند. نقطه تعادل و در حقیقت تعداد نهایی وزنهاى شبکه طبق فرمولهای ۱۰ و ۱۳ وابسته به مقادیر λ_{ON} و λ_{OFF} میباشد. در بخش ۶ ضمن تحلیل تأثیر مقادیر X_{ON} و X_{OFF} بر روی الگوریتم تعیین تعداد وزن، روشی را برای تنظیم λ_{ON} و λ_{OFF} ارائه کردیم.

```

Input:
Training Patterns (p,t), p is input and t is desired output
No. of Hidden Units H
Output:
Network Weight Vector:W
Network Topology
Initialize Weights and biases
Initialize Automata Parameters
Initialize Learning Parameters
for i=1:100
for i=1:K
Call BP
end
MinimumOfError_new=minimum(Error Vector);
Select an Action from on_width_coefficient set
Select an Action from off_width_coefficient set
for all Hidden Neurons
for all weights
if weight is OFF
W = W*WEIGHT_DEC_COEFF.;
end
end
Compute Xon
Compute Xoff
for all weights
if weight is OFF
if abs(W) < Average_off- Xoff
Reward OFFWeight
elseif abs(W) > Average_off + Xoff
Penalize OFFWeight
end
if weight is ON.
if abs(W) < Average_on - Xon
Penalize ON Weights
elseif abs(W) > Average_on + Xon
Reward ON Weights
end
end
end
end
if MinOfErr_new / MinOfErr > MAX_ERR_RATIO
EnvironmentResponce=1;
else
EnvironmentResponce=0;
end
Update_Tsetlin_State_λON (EnvironmentResponce)
Update_Tsetlin_State_λOFF (EnvironmentResponce)
end

```

شکل (۱۱) الگوریتم AWSA.

در این بخش قصد داریم روش دیگری را برای بهبود عملکرد الگوریتم تعیین تعداد وزن ارائه دهیم. همچنانکه قبلاً توضیح دادیم الگوریتم تعیین تعداد وزن طوری عمل میکند در نهایت به یک نقطه تعادل برسیم. تعداد نهایی وزنهاى شبکه یا عبارت دیگر نقطه تعادل را مقادیر λ_{ON} و λ_{OFF} تعیین میکنند. اگر این پارامترها بطور مناسب انتخاب شده باشند در اینصورت در انتهای آموزش ضمن اینکه به خطای قابل قبولی می رسیم، می توانیم در حد مطلوب باحذف وزنهاى غیر ضروری، پیچیدگی شبکه را کاهش دهیم. ولی اگر مقادیر این پارامترها بطور مناسبی انتخاب نشوند دو حالت ممکن است اتفاق بیفتد در حالت اول ممکن است تعداد وزنهاى حذف شده بقدری زیاد باشد که در نتیجه شبکه عصبی حاصل بیش از اندازه ساده شده و توانایی یادگیری الگوهای آموزش را نداشته باشد و درحالت دوم ممکن است تعداد بسیار کمی وزن حذف شده در نتیجه به هدف اصلی

که همان کاهش پیچیدگی شبکه عصبی میباشد دست پیدا نکنیم. طی یک آزمایش حالتهای فوق را برای مسئله شناسائی اعداد لاتین نشان میدهیم. در این آزمایش ضریب کاهش وزن را ۰/۹، تعداد تکرار الگوریتم پس انتشار خطا در هر گام را ۵۰، نرخ یادگیری را ۰/۰۱ و ضریب ممنتم را ۰/۹۸ در نظر می‌گیریم. از تابع غیر خطی سیگموئید که بین ۱- و ۱+ تغییر میکند بعنوان تابع لایه مخفی و لایه خروجی استفاده می‌کنیم. نتایج آزمایش در جدول ۳ آورده شده است. همچنانکه در جدول ۳ نشان داده شده است در سطر اول مقادیر مناسبی برای پارامترهای λ_{ON} و λ_{OFF} انتخاب شده در نتیجه ضمن اینکه شبکه همگرا شده است با درصد خوبی نیز شاهد کاهش پیچیدگی شبکه هستیم ولی در سطر دوم وسوم مقادیر انتخاب شده برای λ_{ON} و λ_{OFF} مناسب نبوده در نتیجه نقاط تعادل حاصل نقاط مناسبی نمیباشد. در سطر دوم درصد شاخه زنی بالا بوده در نتیجه شبکه حاصل نتوانسته الگوهای آموزش را یاد بگیرد و در سطر سوم با اینکه شبکه همگرا شده است ولی درصد شاخه زنی پائین بوده و در حد مطلوبی شبکه ساده نشده است.

جدول (۳) تعداد وزنها بعد از هرس و خطای آموزش به ازای مقادیر مختلف λ_{ON} و λ_{OFF} .

λ_{ON}	λ_{OFF}	تعداد وزنها قبل از هرس	تعداد وزنها بعد از هرس	خطای آموزش
۱۰	۱۰	۳۸۴۰	۴۹۹	۰/۰۱
۲	۱۰	۳۸۴۰	۹۵	۲/۰۵
۱۰	۰/۱	۳۸۴۰	۹۰۰	۰/۰۱

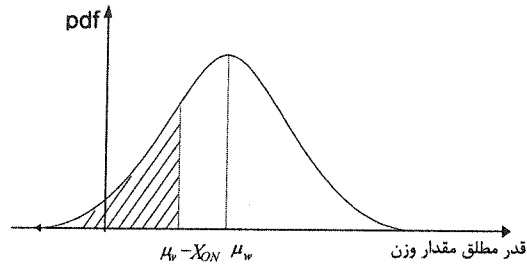
برای حل مشکل فوق روش زیر ارائه می‌گردد. در الگوریتم WSA همچنانکه قبلاً توضیح دادیم خاموش شدن وزنها روشن خود بصورت یک فیدبک منفی عمل کرده و شدت خاموش شدن وزنها را کاهش میدهد. در الگوریتم جدید که آنرا MWSA نامگذاری میکنیم. فرمول مربوط به بازه روشنی X_{ON} را طوری تغییر میدهیم. بطوریکه کاهش وزنها روشن، عبارت دیگر افزایش وزنها خاموش تأثیری بر روند حذف وزنها نداشته باشد. برای این منظور فرمول جدیدی بصورت زیر ارائه میشود.

$$X_{ON} = \lambda_{ON} \times \sigma_h \quad (16)$$

که در آن λ_{ON} ضریب پهنای روشنی و σ_h واریانس وزنها روشن میباشد که بصورت زیر محاسبه میشود.

$$\sigma_h = \sqrt{\frac{\sum_{w_{jk} \in ON_w(h)} (W_{jk} - \mu_w(h))^2}{|ON_w(h)|}} \quad h \in ON \quad (17)$$

که در آن W_{jk} وزن اتصال بین واحد ورودی K به واحد مخفی h، ON مجموعه واحدهای روشن، $ON_w(h)$ مجموعه وزنها ورودی روشن متصل به واحد مخفی h و $\mu_w(h)$ میانگین قدر مطلق وزنها روشن متصل به واحد مخفی h میباشد. هر قدر واریانس وزنها ورودی یک نرون زیاد باشد. مفهوم آن این است که تعداد زیادی وزن داریم که مقدار آنها کمتر از مقدار متوسط وزنها میباشد. لذا در این حالت بایستی مقدار X_{ON} نیز زیاد انتخاب شود تا شرط جریمه کردن وزنها روشن سنگین شده و وزنها زیادی خاموش نشوند. در مقابل هر قدر واریانس وزنها ورودی یک نرون کم باشد. این به منزله آن است که اکثریت وزنها حول و حوش مقدار متوسط وزنها بوده در نتیجه تعداد کمی وزن داریم که مقدار آنها خیلی کمتر از مقدار متوسط میباشد. در این حالت بایستی مقدار X_{ON} را متناسب با واریانس وزنها کاهش داد تا شرط جریمه کردن وزنها روشن سهلتر شده و بتوان تعدادی از وزنها را جریمه کرد. با توجه به مطالب مذکور مقدار X_{ON} بایستی متناسب با σ_h انتخاب شود. شکل ۱۲ تابع چگالی احتمال وزنها روشن را نشان میدهد. در اینجا فرض کردیم که وزنها روشن دارای توزیع نرمال میباشد. توزیع نرمال دارای تابعی بصورت زیر می‌باشد.



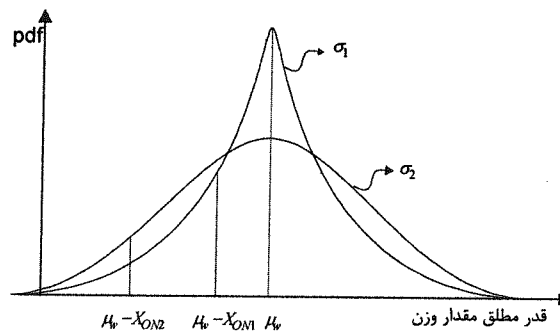
شکل (۱۲) منحنی توزیع قدر مطلق.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-(x-\eta)^2/2\sigma^2} = \frac{1}{\sigma} g\left(\frac{x-\eta}{\sigma}\right) \quad (18)$$

احتمال اینکه قدر مطلق وزنی از $\mu_w - X_{ON}$ کمتر باشد با محاسبه انتگرال زیر بدست می آید.

$$F(\mu_w - X_{ON}) = \frac{1}{\sigma\sqrt{2\pi}} \cdot \int_{-\infty}^{\mu_w - X_{ON}} e^{-(\xi-\eta)^2/2\sigma^2} \cdot d\xi = G\left(\frac{x-\eta}{\sigma}\right) \quad (19)$$

که در آن η مقدار متوسط قدر مطلق وزنه‌های روشن و σ واریانس قدر مطلق وزنه‌های روشن می‌باشد. انتگرال فوق در حقیقت سطح زیر منحنی توزیع نرمال در بازه $[-\infty, \mu_w - X_{ON}]$ را نشان می‌دهد. حال هر قدر مقدار واریانس وزنه‌ها کمتر باشد در این صورت منحنی تابع چگالی احتمال باریکتر شده و در نتیجه سطح زیر منحنی کاهش پیدا می‌کند. در این حالت نیز برای اینکه بتوان درصد قابل قبولی از وزنه‌ها را جریمه کرد بایستی متناسب با کاهش واریانس، نقطه $\mu_w - X_{ON}$ را به سمت راست انتقال داده و کاهش سطح زیر منحنی را جبران کنیم. که این کار را میتوان با کاهش X_{ON} انجام داد. لذا نتیجه می‌گیریم که مقدار X_{ON} بایستی متناسب با σ باشد. در شکل ۱۳ منحنی تابع چگالی احتمال برای دو واریانس مختلف نشان داده شده است. منحنی شماره یک دارای واریانس σ_1 بوده و منحنی شماره دو دارای واریانس σ_2 می‌باشد. $\sigma_2 > \sigma_1$ بوده در نتیجه منحنی شماره یک باریکتر و تیزتر از منحنی شماره دو می‌باشد. با کاهش σ و باریک شدن منحنی تابع چگالی احتمال بایستی مقدار X_{ON} متناسب با آن کاهش یابد تا کاهش سطح منحنی جبران شود. در الگوریتم جدید MWSA برای ارزیابی عملکرد وزنه‌های خاموش و تصمیم‌گیری در مورد نحوه روشن کردن وزنه‌های خاموش بصورت زیر عمل می‌کنیم.



شکل (۱۳) منحنی توزیع قدر مطلق وزنه‌های

روشن به ازای دو مقدار متفاوت σ

نحوه ارزیابی عملکرد وزنه‌های خاموش: برای ارزیابی عملکرد وزنه‌های خاموش از منحنی خطای آموزش بهره می‌گیریم. اگر نرخ کاهش خطا از یک مقدار آستانه کمتر بوده و همچنین تفاضل مقدار خطا از خطای هدف از یک مقدار آستانه دیگر بیشتر باشد. در این حالت شبکه قادر به یادگیری الگوهای آموزش نبوده و گوئیم وزن خاموش دارای عملکرد نامطلوب می‌باشد. اگر وزنی دارای عملکرد نامطلوب باشد در این صورت جریمه شده و به سمت حالت‌های بیرونی حرکت می‌کند.

اگر نرخ کاهش خطا از یک مقدار آستانه کمتر بوده و همچنین تفاضل مقدار خطا از خطای هدف از یک مقدار آستانه دیگر بیشتر باشد. در این حالت شبکه قادر به یادگیری الگوهای آموزش نبوده و گوئیم وزن خاموش دارای عملکرد نامطلوب میباشد. اگر وزنی دارای عملکرد نامطلوب باشد در این صورت جریمه شده و به سمت حالت‌های بیرونی حرکت میکند. نرخ کاهش خطا را بصورت زیر تعریف می‌کنیم.

$$\text{نرخ کاهش خطا} = \frac{\min E(k) - \min E(k-1)}{N} \quad (20)$$

کاهش خطا

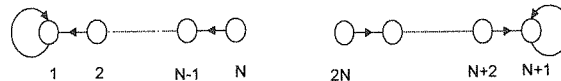
که در آن $\min E(k)$ حداقل مقدار خطا در مرحله k و $\min E(k-1)$ حداقل مقدار خطا در مرحله $k-1$ می‌باشد. N نیز تعداد تکرار الگوریتم پس انتشار خطا در هر مرحله می‌باشد. پس شرط جریمه کردن وزنهای خاموش را میتوان بصورت زیر نوشت. اگر:

$$\left[\frac{\min E(k) - \min E(k-1)}{N} < \varepsilon \right] \wedge [(\min E(k) - \text{errorgoal}) > \gamma]$$

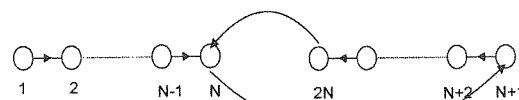
آنگاه: وزنهای خاموش را جریمه کن

اگر نرخ کاهش خطا از یک مقدار آستانه بیشتر باشد. این حالت نشاندهنده آنست که شبکه در حال یادگیری الگوهای آموزش بوده و نیازی به روشن کردن وزنهای خاموش نمی‌باشد که در این حالت گوئیم وزن خاموش دارای عملکرد مناسب بوده در نتیجه به آن پاداش داده میشود.

با دریافت پاداش حالت وزن خاموش به سمت حالت‌های درونی حرکت می‌کند. اتوماتان مربوط به الگوریتم جدید MWSA در شکل ۱۴ نشان داده شده است. در شکل ۱۵ حالتی را که شبکه قادر به یادگیری الگوهای آموزش نبوده نشان داده شده است. الگوریتم روش جدید MWSA در شکل ۱۶ نشان داده شده است.

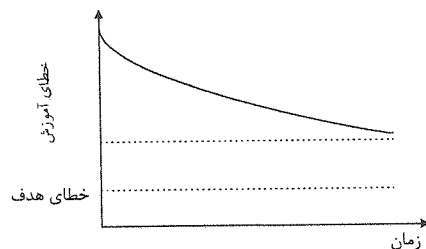


جواب مطلوب



جواب نا مطلوب

شکل (۱۴) اتوماتان مربوط به الگوریتم MWSA.



شکل (۱۵) شبکه عصبی قادر به یادگیری الگوهای

نتایج شبیه سازی

در این قسمت ۴ مسئله نمونه که الگوریتمهای پیشنهادی روی آنها پیاده سازی شده است معرفی میگردد. در ادامه نتایج آزمایشهای مختلف ارائه خواهد شد.

الف - مسئله شناسایی اعداد لاتین: در این مسئله می‌خواهیم شبکه‌ای را آموزش بدهیم که بتواند اعداد از ۰ تا ۹ را تشخیص دهد. برای این منظور اعداد را مطابق شکل ۱۷ نمایش می‌دهیم. برای هر عدد یک ماتریس 8×8 در نظر می‌گیریم. خانه‌های سیاه را با ۱ و خانه‌های سفید را با ۰ نشان می‌دهیم. اعداد ۰ تا ۹ را می‌توان مطابق جدول ۴ با چهار بیت کد کرد. لذا خروجی شبکه دارای ۴ نرون خواهد بود. از یک شبکه سه لایه با ۶۴ نرون در لایه ورودی و ۴ نرون در لایه خروجی استفاده کرده‌ایم. برای آزمایشهای مختلف تعداد نرونهای لایه مخفی را بین ۱۵ و ۲۳ تغییر می‌دهیم.

ب - مسئله تقریب تابع غیر خطی زمان گسسته درجه دوم: تابع غیر خطی زمان گسسته از مرتبه دوم با رابطه زیر مفروض می‌باشد.

$$y_{k+1} = \frac{1.5y_k y_{k-1}}{1+y_k^2+y_{k-1}^2} + 0.35(y_k + y_{k-1}) + 1.2u_k \quad (21)$$

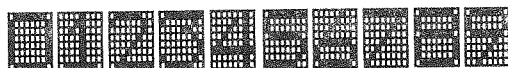
می‌خواهیم با استفاده از یک شبکه عصبی سه لایه تابع فوق را با تقریب قابل قبولی شبیه‌سازی کنیم. برای این منظور ورودیهایی بین ۰- و ۱+ بطور تصادفی ایجاد کرده و به تابع اعمال می‌کنیم شرایط اولیه را نیز بطور تصادفی بین ۰- و ۱+ انتخاب می‌کنیم بدین ترتیب یک سری الگوی آموزش ایجاد می‌شود که از این الگوها برای آموزش شبکه استفاده می‌کنیم. در رابطه فوق u_k و y_k به ترتیب ورودی و خروجی در لحظه k بوده و y_{k+1} و y_{k-1} به ترتیب خروجی در لحظات $k+1$ و $k-1$ می‌باشد. برای تقریب تابع از یک شبکه عصبی سه لایه با ۳ نرون در لایه ورودی و ۱ نرون در لایه خروجی استفاده کرده‌ایم. تعداد نرونهای لایه میانی را برای آزمایشهای مختلف بین ۸ و ۲۵ تغییر می‌دهیم.

```

Input:
Training Patterns (p,t), p is input and t is desired output
No. of Hidden Units H
Output:
Network Weight Vector: W
Network Topology
Initialize Weights and biases
Initialize Automata Parameters
Initialize Learning Parameters
for i=1:100
  for i=1:K
    Call BP
  end
  for all Hidden Neurons
    Compute Xon
    for all weights
      if weight is OFF
        if  $\left[ \frac{\min E(k) - \min E(k-1)}{N} < \varepsilon \right] \wedge [\min E(k) - errorgoal] > \gamma$ 
          Reward OFFWeight
        else
          Penalize OFFWeight
        end
      if weight is ON.
        if  $abs(W) < Average\_on - Xon$ 
          Penalize ON Weights
        elseif  $abs(W) > Average\_on + Xon$ 
          Reward ON Weights
        end
      end
    end
  end
end
end
end

```

شکل (۱۶) الگوریتم MWSA.



شکل (۱۷) نحوه نمایش اعداد از ۰ الی ۹.

جدول (۴) نحوه کد کردن اعداد از ۰ الی ۹.

عدد	بیت ۳	بیت ۲	بیت ۱	بیت ۰
۰	-۱	-۱	-۱	-۱
۱	-۱	-۱	-۱	۱
۲	-۱	-۱	۱	-۱
۳	-۱	-۱	۱	۱
۴	-۱	۱	-۱	-۱
۵	-۱	۱	-۱	۱
۶	-۱	۱	۱	-۱
۷	-۱	۱	۱	۱
۸	۱	-۱	-۱	-۱
۹	۱	-۱	-۱	۱

ج - مسئله تشخیص اعداد چاپی فارسی: در این مسئله هدف تشخیص اعداد چاپی فارسی میباشد (شکل ۱۸). این اعداد روی یک صفحه کاغذ چاپ شده‌اند. اعداد فارسی از طریق یک دستگاه پویسگر با قدرت تفکیک ۳۰۰ نقطه در اینچ نمونه برداری گردیده است. پس از جدا سازی حروف در تصویر، ثابت‌های گشتاور M_1 تا M_7 استخراج شده و ۷ ویژگی ورودیهای شبکه عصبی را تشکیل میدهند.



شکل (۱۸) اعداد چاپی فارسی.

د - مسئله بازشناسی فونمهای فارسی: در این مسئله هدف بازشناسی فونمهای مصوت فارسی میباشد. دادگان گفتار مورد استفاده در این آزمایشها دادگان فارسی فارس دات میباشد [۴۴]. ابتدا سیگنالهای گفتار با فرکانس ۴۴/۱ KHZ نمونه برداری شده سپس فرکانس آن تا ۱۶ KHZ کاهش داده میشود. ضرائب کپسترال LPC از مرتبه ۱۴ بعلاوه مشتق ضرائب کپسترال LPC و تغییرات لگاریتم انرژی بعنوان ویژگیهای گفتار از هر فریم با طول ۲۰ میلی ثانیه و شیفت زمانی ۱۰ میلی ثانیه استخراج شده است. نمونه‌های گفتار قبل از محاسبه ضرائب کپسترال LPC با یک فیلتر با ضریب ۰/۹۷ فیلتر میشوند. در مرحله آموزش تعداد ۹۳۸ الگو مورد استفاده قرار گرفته است. همچنین از یک شبکه عصبی سه لایه با تاخیر زمانی ۳ فریم استفاده شده است. در بخشهای بعدی نتایج شبیه‌سازی برای مسائل فوق ارائه گردیده است. در تمامی شبیه‌سازیها نرخ یادگیری ۰/۰۱ و تعداد تکرارهای الگوریتم پس انتشار خطا در هر مرحله ۵۰ میباشد.

۱-۸- مقایسه الگوریتم ارائه شده AWSA با الگوریتم WAS

به منظور ارزیابی عملکرد الگوریتم پیشنهادی AWSA آزمایشهای گوناگونی بر روی چهار مسئله مختلف صورت گرفته است. برای هر مسئله دو دسته آزمایش ترتیب داده شده است یک دسته آزمایش برای الگوریتم WSA و دسته دیگر برای الگوریتم AWSA.

نتایج شبیه‌سازی برای الگوریتم WSA: الگوریتم WSA برای هر کاربرد و برای ۶ شبکه مختلف با تعداد نرونهای لایه مخفی متفاوت پیاده‌سازی شده است. نتایج شبیه‌سازیها در جداول ۵ الی ۸ نشان داده شده است.

الف - مسئله شناسایی اعداد انگلیسی: ضرائب پهنای خاموشی و روشنی ۱۰، ضریب کاهش وزن ۰/۹ و ضریب ممنتم صفر می‌باشد. الگوهای آموزش ۵۰۰۰ مرتبه به شبکه اعمال شده است. نتایج شبیه‌سازی در جدول ۵ نشان داده شده است.

ب - مسئله تقریب تابع غیر خطی زمان گسسته از مرتبه دوم: ضرائب پهنای خاموشی و روشنی ۱۰، ضریب کاهش وزن ۰/۹ و ضریب ممنتم ۰/۹۸ میباشد. الگوهای آموزش ۵۰۰۰ مرتبه به شبکه اعمال شده است. نتایج شبیه سازی در جدول ۶ نشان داده شده است.

ج - مسئله شناسای اعداد فارسی: ضرائب پهنای روشنی و خاموشی ۵، ضریب کاهش وزن ۰/۹ و ضریب ممنتم ۰/۹۸ میباشد. نتایج شبیه سازی در جدول ۷ نشان داده شده است.

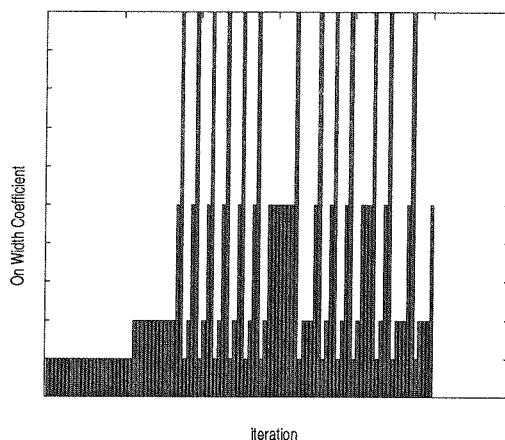
د - مسئله باز شناسی فونم: ضرائب پهنای خاموشی و روشنی به ترتیب ۲ و ۵۰، ضریب کاهش وزن ۰/۹ و ضریب ممنتم ۰/۹۸ می باشد. نتایج شبیه سازی در جدول ۸ نشان داده شده است.

نتایج شبیه سازی برای الگوریتم AWSA: برای هر مسئله، الگوریتم AWSA برای ۶ شبکه مختلف با تعداد نرونهای لایه مخفی متفاوت پیاده سازی شده است. برای تمامی مسائل اتو ماتانهای مختلف برای تطبیق ضرائب پهنای روشنی و خاموشی مورد استفاده قرار گرفته است. نتایج شبیه سازیها در جداول ۵ الی ۸ و شکلهای ۱۹ الی ۲۵ نشان داده شده است.

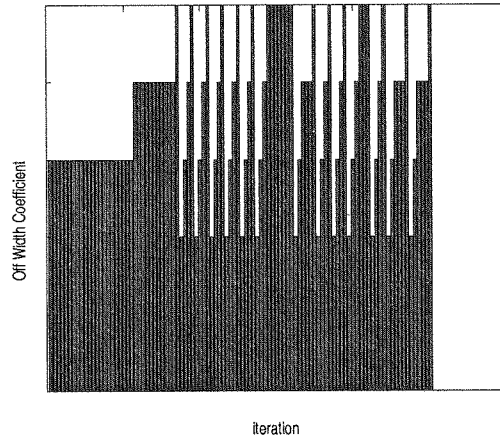
الف - مسئله شناسای اعداد انگلیسی: {۱،۲،۵،۱۰} مجموعه ضرائب پهنای روشنی بوده و {۱۰،۱۵،۲۰،۲۵} مجموعه ضرائب پهنای خاموشی می باشد. ضریب کاهش وزن و ضریب ممنتم به ترتیب ۰/۹ و ۰ در نظر گرفته شده است. الگوهای آموزش ۵۰۰۰ مرتبه به شبکه اعمال شده است. نتایج شبیه سازی در جدول ۵ نشان داده شده و شکلهای ۱۹ و ۲۰ تغییرات ضریب پهنای روشنی و خاموشی نسبت به epochها را نشان میدهد.

جدول (۵) مقایسه الگوریتمهای WSA و AWSA برای مسئله شناسای اعداد لاتین.

شماره شبکه	تعداد نرونهای مخفی	الگوریتم WSA		الگوریتم AWSA	
		در صد تشخیص		در صد تشخیص	
		تعداد وزنهای لایه مخفی	آموزش	تعداد وزنهای لایه مخفی	آموزش
۱	۱۵	۴۹۹	٪۱۰۰	۲۰۶	٪۱۰۰
۲	۱۵	۵۳۴	٪۱۰۰	۱۸۷	٪۹۰
۳	۱۷	۵۵۸	٪۱۰۰	۱۷۷	٪۹۰
۴	۱۹	۶۳۷	٪۱۰۰	۱۹۲	٪۱۰۰
۵	۲۱	۷۱۴	٪۱۰۰	۱۷۲	٪۱۰۰
۶	۲۲	۷۷۲	٪۱۰۰	۲۶۱	٪۹۰
متوسط	۱۸	۶۱۹	٪۱۰۰	۱۹۹	٪۹۵

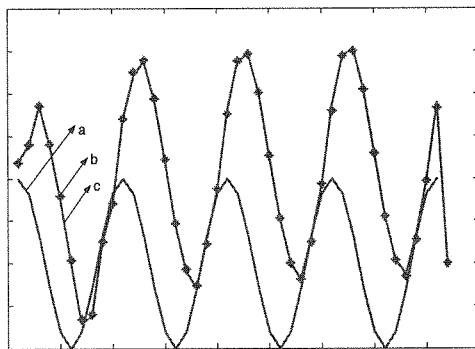


شکل (۱۹) منحنی تغییرات ضریب پهنای روشنی.

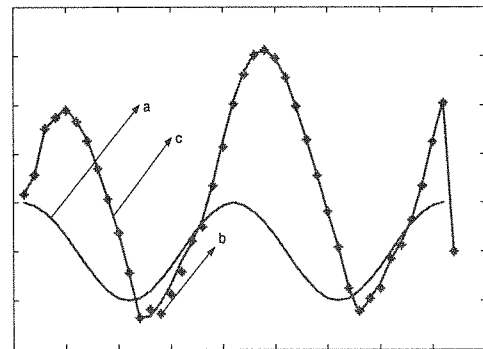


شکل (۲۰) منحنی تغییرات ضریب پهنای روشنی.

ب - مسئله تقریب تابع غیر خطی زمان گسسته مرتبه دوم: در الگوریتم AWSA حداکثر خطا مربوط به شبکه شماره ۳ با مقدار $2/2082$ می‌باشد. برای این مورد، موقعیکه ورودی سینوسی مورد استفاده قرار گرفته است شکل موجهای خروجی تابع و خروجی شبکه در شکلهای ۲۱ و ۲۲ برای دو فرکانس مختلف نشان داده شده است. همچنانکه در این شکلها نشان داده شده است خروجی شبکه عصبی به نحو مطلوبی خروجی تابع را دنبال میکند. مجموعه ضرائب پهنای روشنی و $\{10, 12, 14, 16\}$ مجموعه ضرائب پهنای خاموشی می‌باشد. ضریب کاهش وزن و ضریب ممنتم به ترتیب $0/9$ و $0/98$ انتخاب شده است. الگوهای آموزش 5000 مرتبه به شبکه اعمال شده است. نتایج شبیه سازی در جدول ۶ نشان داده شده است.



شکل (۲۲) (a) ورودی (b) خروجی تابع (c) خروجی شبکه.



شکل (۲۱) (a) ورودی (b) خروجی تابع (c) خروجی شبکه.

ج - مسئله شناسایی اعداد چاپی فارسی: $\{3/5, 4, 4/5, 5\}$ مجموعه ضرائب پهنای روشنی و $\{15, 16, 17, 18\}$ مجموعه ضرائب پهنای خاموشی می‌باشد. ضریب کاهش وزن و ضریب ممنتم به ترتیب $0/9$ و $0/98$ در نظر گرفته شده است. نتایج آزمایشها در جدول ۷ نشان داده شده است.

د - مسئله باز شناسی فونم: $\{1, 2, 5, 10\}$ مجموعه ضرائب پهنای روشنی و $\{10, 15, 20, 25\}$ مجموعه ضرائب پهنای خاموشی می‌باشد. ضریب کاهش وزن و ضریب ممنتم به ترتیب $0/9$ و $0/98$ میباشد. نتایج شبیه سازی در جدول ۸ نشان داده شده است.

برای این مسئله شبیه سازی دیگری انجام شده و منحنی تغییرات تعداد وزنها بر حسب epoch (شکلهای ۲۳ و ۲۴ را ببینید) رسم شده است. همچنانکه در این شکلها نشان داده شده است در الگوریتم تطبیقی AWSA در ابتدای الگوریتم یک کاهش شدید در تعداد وزنها اتفاق می افتد و سپس به تدریج کم می‌شود. بر خلاف الگوریتم AWSA، در الگوریتم WSA

ابتدای الگوریتم، تعداد وزنها با نرخ کم کاهش یافته و بعد از تعدادی epoch شروع به افزایش کرده و برای مدت زمان زیادی ثابت باقی می ماند.

جدول (۶) مقایسه الگوریتمهای WSA و AWSA برای مسئله تقریب تابع غیر خطی زمان گسسته درجه دوم.

شماره شبکه	تعداد نرونهای مخفی	الگوریتم WSA		الگوریتم AWSA	
		تعداد وزنها لایه مخفی	خطای شبکه عصبی به ازای ورودی سینوسی	تعداد وزنها لایه مخفی	خطای شبکه عصبی به ازای ورودی سینوسی
۱	۸	۲۳	۰/۳۸۲۲	۱۹	۱/۱۱۶۲
۲	۱۰	۲۵	۰/۹۷۱۷	۲۲	۰/۹۴۰۹
۳	۱۳	۳۲	۲/۷۰۰۵	۲۸	۲/۲۰۸۲
۴	۱۷	۴۱	۰/۶۵۲۶	۳۸	۰/۷۳۲۵
۵	۲۰	۴۷	۰/۶۶۳۱	۴۳	۰/۸۰۱۲
۶	۲۵	۶۳	۱/۳۰۴۷	۵۷	۱/۴۵۸۰
متوسط	۱۵/۵	۳۸/۵	۱/۱۱۲۵	۳۴/۵	۱/۲۰۹۵

تذکره: الگوریتم ارائه شده AWSA شبکه‌هایی با اتصالات محلی که دارای پیچیدگی پائین و قدرت تعمیم بالایی هستند تولید میکنند. نتایج مشابه‌سازیها نشان میدهد شبکه‌های تولید شده توسط الگوریتم جدید از پیچیدگی پائین تری نسبت به الگوریتم تعیین وزن WSA برخوردار میباشند. قدرت شاخه زنی الگوریتم پیشنهادی جدید برای مسئله شناسایی اعداد لاتین ۶۸٪، مسئله تشخیص اعداد چاپی فارسی ۳۲٪، مسئله تقریب تابع غیر خطی زمان گسسته درجه دوم ۱۰٪ و مسئله شناسایی فونمهای فارسی ۴۹٪ بیشتر از الگوریتم WSA میباشد.

جدول (۷) مقایسه الگوریتمهای WSA و AWSA برای مسئله تشخیص اعداد چاپی فارسی.

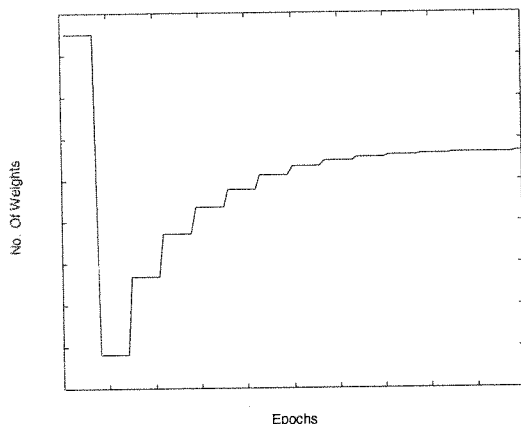
شماره شبکه	تعداد نرونهای مخفی	الگوریتم WSA		الگوریتم AWSA	
		در صد تشخیص		تعداد وزنها لایه مخفی	
		آموزش	آزمایش	آموزش	آزمایش
۱	۱۴	۵۹	٪۹۸	۳۵	٪۹۸
۲	۱۵	۶۰	٪۹۹	۴۰	٪۹۷
۳	۱۶	۶۱	٪۹۲	۴۳	٪۹۶
۴	۱۷	۷۰	٪۹۴	۵۰	٪۹۶
۵	۱۸	۸۰	٪۹۷	۵۵	٪۹۷
۶	۱۹	۸۹	٪۹۹	۶۰	٪۹۹
متوسط	۱۶/۵	۷۰	٪۹۶/۵	۴۷	٪۹۷/۲

جدول (۸) مقایسه الگوریتمهای WSA و AWSA برای مسئله شناسایی فونمهای فارسی.

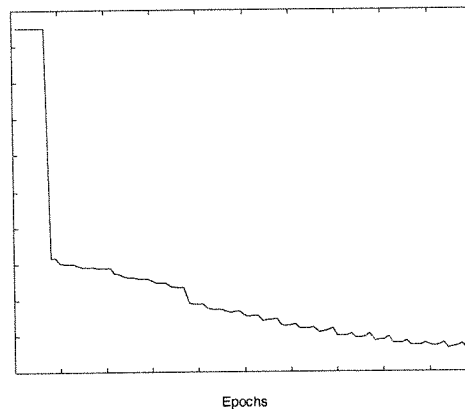
الگوریتم AWSA		الگوریتم WSA		تعداد نرونهای مخفی	شماره شبکه
در صد تشخیص		در صد تشخیص			
آزمایش	آموزش	آزمایش	آموزش	تعداد وزنه‌های لایه مخفی	
٪۸۱	٪۸۳	٪۸۷	٪۸۷	۹۴۵	۱۶
٪۸۹	٪۸۵	٪۸۴	٪۸۷	۱۱۷۲	۲۰
٪۸۴	٪۸۷	٪۸۲	٪۸۵	۱۴۱۸	۲۴
٪۸۸	٪۸۲	٪۸۲	٪۸۳	۱۶۴۴	۲۸
٪۸۱	٪۸۸	٪۸۱	٪۸۴	۱۸۵۵	۳۲
٪۸۴	٪۸۲	٪۸۴	٪۸۲	۲۱۰۷	۳۶
٪۸۴/۵	٪۸۴	٪۸۳	٪۸۴/۷	۱۵۲۳	۲۶

تذکر ۲: همچنانکه در شکل‌های ۲۳ و ۲۴ نشان داده است سطح زیر منحنی تغییرات تعداد وزن بر حسب epoch مربوط به الگوریتم AWSA کمتر از سطح زیر منحنی مربوط به الگوریتم WSA می‌باشد. این بدان معناست که در الگوریتم AWSA طی فرایند آموزش برای مدت زمان زیادی پیچیدگی شبکه پائین بوده که این مسئله می‌تواند باعث کاهش زمان آموزش شبکه گردد. بویژه این مسئله موقعی اهمیت پیدا میکند که از پردازشگرهای موازی برای آموزش شبکه استفاده کنیم که در این حالت با کاهش پیچیدگی شبکه نه تنها زمان محاسبات کاهش می‌یابد بلکه بدلیل کاهش ارتباطات بین نرونها کاهش چشمگیری هم در زمان ارتباطات حاصل میشود. لذا الگوریتم AWSA کاندید مناسبی برای آموزش شبکه‌های عصبی با استفاده از پردازشگرهای موازی می‌باشد [۴۵].

تذکر ۳: الگوریتم AWSA در ابتدا مقادیر ضرائب پهنای روشنی و خاموشی را طوری انتخاب میکند که فرایند حذف وزن با بیشترین سرعت شروع شود. عمل حذف وزن با حداکثر سرعت ادامه می‌یابد تا زمانیکه خطای آموزش شروع به افزایش کند که در این حالت الگوریتم AWSA با انتخاب مقادیر دیگری برای ضرائب پهنای روشنی و خاموشی سرعت شاخه زنی را کاهش داده و در صورت لزوم برخی از وزنه‌های حذف شده را دوباره روشن میکند. ملاحظه میکنیم الگوریتم AWSA با تنظیم مقادیر ضرائب پهنای روشنی و خاموشی میتواند با داشتن عملکرد تطبیقی درصد بیشتری از وزنه‌های شبکه را نسبت به الگوریتم WSA حذف کند.



شکل (۲۴) منحنی تعداد وزنها بر حسب epochs برای الگوریتم WSA



شکل (۲۳) منحنی تعداد وزنها بر حسب epochs برای الگوریتم AWSA



شکل (۲۵) نمایش اعداد ۰، ۱، ۲.

تذکره ۴: قدرت تعمیم شبکه‌های هرس شده در اکثر موارد از قدرت تعمیم شبکه‌های هرس نشده بیشتر می‌باشد. دلیل این امر اینست که با کاهش تعداد پارامترهای شبکه ابر صفحه‌های تولید شده توسط هر یک از نرونهای مخفی هموار می‌گردد. حقیقت دیگری که با دقت در نتایج آزمایش می‌توان به آن رسید آنست که هر چقدر بعد بردارهای ورودی بیشتر باشد به همان نسبت درصد حذف وزنهای ورودی نرونهای لایه مخفی بیشتر می‌شود. هر چقدر بعد فضای ورودی زیاد باشد عبارت دیگر اگر تعداد زیادی ویژگی برای شناسایی الگوهای ورودی مورد استفاده قرار گیرد. در این صورت افزونگی بردارهای ورودی افزایش یافته و فقط تعداد کمی از این ویژگیها برای طبقه‌بندی الگوها مورد نیاز می‌باشد. در این حالت الگوریتم یادگیری ساختار شبکه در طی آموزش ارتباط ویژگیهایی را که نقش مهمی در شناسایی الگوها ایفا نمی‌کنند را با نرونهای لایه مخفی قطع میکند. عبارت دیگر وزنهای مربوطه را صفر میکند. بعنوان مثال برای طبقه‌بندی اعداد لاتین 0,1,2 از بردارهای ورودی با ابعاد 64×1 استفاده می‌کنیم. در این حالت بعد فضای ورودی ۶۴ می‌باشد. با دقت در اشکال این اعداد (شکل ۲۵) ملاحظه می‌کنیم که قسمت پائین این اعداد در تمامی آنها مشترک بوده و اطلاعات مفیدی را جهت طبقه‌بندی این اعداد در اختیار شبکه عصبی قرار نمیدهد. لذا این بخش از بردار ورودی غیر ضروری بوده و الگوریتم یادگیری ساختار شبکه در حین آموزش ارتباط این بخش از بردار ورودی با نرونهای لایه مخفی را قطع می‌کند.

۲-۸ - مقایسه اتوماتانهای مختلف

در این آزمایش انواع مختلف اتوماتانهای با ساختار ثابت که عبارتند از اتوماتان ستلین، کریسکی و کرایلف و همچنین اتوماتانهای L_{R-I} و L_{R-P} را که جزو اتوماتانهای با ساختار متغییر محسوب می‌شوند برای تنظیم پارامترهای λ_{ON} و λ_{OFF} مورد استفاده قرار گرفته است. این آزمایش را برای کاربردهای شناسایی اعداد انگلیسی، تقریب تابع غیر خطی زمان گسسته از مرتبه دوم، شناسایی اعداد فارسی و باز شناسی فونمهای فارسی انجام داده ایم. شرایط آزمایش مطابق آزمایشهای مربوط به الگوریتم AWSA میباشد که پیش از این توضیح داده شده است. برای اتوماتانهای ستلین، کریسکی و کرایلف عمق حافظه را ۵ در نظر گرفته ایم. پارامترهای a, b مربوط به اتوماتان L_{R-P} را که در حقیقت طول گامهای اتوماتان را مشخص می‌کند به ترتیب ۰/۰۱ و ۰/۰۱ انتخاب کرده ایم. پارامتر ϵ مورد استفاده در اتوماتان را نیز ۰/۰۱ در نظر گرفته ایم. نتایج آزمایشهای مختلف در جدول ۹ الی ۱۲ آورده شده است.

جدول (۹) مقایسه اتوماتانهای مختلف برای تنظیم پارامترهای λ_{ON} و λ_{OFF} برای کاربرد شناسایی اعداد انگلیسی.

شماره شبکه	تعداد نرونهای مخفی	تعداد وزنهای لایه اول			
		ستلین	کریسکی و کرایلف	L_{R-P}	L_{R-I}
۱	۱۵	۲۰۶	۲۲۴	۲۳۰	۲۳۵
۲	۱۵	۱۸۷	۱۹۶	۱۹۵	۲۱۰
۳	۱۷	۱۷۷	۱۹۳	۱۹۸	۲۰۹
۴	۱۹	۱۹۲	۲۰۵	۲۱۰	۲۱۴
۵	۲۱	۱۷۳	۱۹۵	۲۰۳	۲۱۹
۶	۲۳	۲۶۱	۲۵۹	۲۶۹	۲۷۷
متوسط	۱۸/۳	۱۹۹/۳	۲۱۲	۲۱۷/۵	۲۲۵/۳

جدول (۱۰) مقایسه اتوماتانهای مختلف برای تنظیم پارامترهای λ_{ON} و λ_{OFF} برای مسئله تقریب تابع غیر خطی زمان گسسته درجه دوم.

شبکه	تعداد نرونهاي مخفي	تعداد وزنهاي لايه اول				
		L_{R-I}	L_{R-P}	کرایلف	کرنيسکي	ستلين
۱	۸	۲۴	۲۵	۲۳	۲۳	۱۹
۲	۱۰	۲۷	۲۹	۲۸	۲۶	۲۲
۳	۱۳	۳۰	۳۳	۳۲	۳۲	۲۸
۴	۱۷	۴۵	۴۴	۴۲	۴۲	۳۸
۵	۲۰	۴۸	۴۷	۴۴	۴۲	۴۳
۶	۲۵	۶۳	۶۱	۶۰	۵۹	۵۷
متوسط	۱۵/۵	۳۹/۵	۳۹/۸	۳۸/۲	۳۷/۳	۳۴/۵

جدول (۱۱) مقایسه اتوماتانهای مختلف برای تنظیم پارامترهای λ_{ON} و λ_{OFF} برای مسئله تشخیص اعداد چاپی فارسی.

شبکه	تعداد نرونهاي مخفي	تعداد وزنهاي لايه اول				
		L_{R-I}	L_{R-P}	کرایلف	کرنيسکي	ستلين
۱	۱۴	۳۴	۳۷	۳۸	۳۷	۳۵
۲	۱۵	۴۵	۴۸	۴۵	۴۷	۴۰
۳	۱۶	۴۹	۴۹	۴۸	۵۰	۴۳
۴	۱۷	۵۵	۵۵	۵۹	۵۸	۵۰
۵	۱۸	۵۹	۶۰	۶۵	۶۳	۵۵
۶	۱۹	۷۰	۷۱	۷۱	۷۲	۶۰
متوسط	۱۶/۵	۵۲	۵۳/۳	۵۴/۳	۵۴/۵	۴۷/۲

تذکره ۵: با توجه به آزمایشهای انجام یافته و با دقت در جدول ۹ الی ۱۲ میتوان به نتایج زیر رسید. در اکثر موارد در بین اتوماتانهای با ساختار ثابت، اتوماتان ستلین دارای عملکرد بهتری می باشد. در مورد اتوماتان کرنيسکي فرض کنیم که اتوماتان دارای عملکرد بدی بوده و محیط پاسخ نامطلوب داده و اتومان به سمت حالت‌های بیرونی حرکت میکند با دریافت هر پاسخ نامطلوب اتوماتان به حرکت خود به سمت حالت‌های بیرونی ادامه می دهد حال اگر در بین راه یک پاسخ مطلوب از محیط در یافت کند به داخلی ترین حالت منتقل می شود.

حال اگر اتومان دارای عملکرد بد بود برای تغییر اقدام بایستی حرکت خود را دو باره از داخلی ترین اقدام شروع کرده و با دریافت N پاسخ نامطلوب متوالی موفق به تغییر اقدام شود ملاحظه می کنیم که در این نوع اتوماتان تغییر اقدام با کندی صورت می گیرد لذا در مقایسه با اتوماتان ستلین دارای عملکرد ضعیفی می باشد. با توجه به نتایج ملاحظه می کنیم که اتوماتان کرایلف در بعضی موارد دارای عملکرد خوبی بوده و در برخی از موارد دارای عملکرد بد میباشد. که این رفتار ناشی از ساختار این نوع اتوماتان میباشد. در این اتوماتان با دریافت پاسخ نامطلوب از محیط با احتمال ۱/۲ یا به سمت حالت‌های داخلی و یا به سمت حالت‌های خارجی حرکت میکند. عملکرد اتوماتانهای L_{R-P} و τ_{o} تقریباً یکسان میباشد.

۸-۳- مقایسه عملکرد الگوریتم MWSA با الگوریتم WSA

با هدف بررسی عملکرد الگوریتم معرفی شده MWSA آزمایشهای متفاوتی بر روی چهار مسئله مختلف صورت است. برای هر کاربرد دو دسته آزمایش انجام داده شده است یک دسته برای الگوریتم WSA و دسته دیگر برای الگوریتم MWSA. نتایج شبیه سازیها برای الگوریتم WSA: این نتایج در حقیقت همان نتایجی است که قبلاً برای مقایسه این الگوریتم با الگوریتم AWSA انجام دادیم. نتایج این آزمایش ها مجدداً در جداول ۱۳ الی ۱۶ آورده شده است.

جدول (۱۲) مقایسه اتو ما تا نهایی مختلف برای تنظیم پارامترهای λ_{ON} و λ_{OFF} برای مسئله شناسایی فونمهای فارسی.

شبکه	تعداد نرونهای لایه مخفی	تعداد وزنه‌های لایه اول بعد از هرس				
		ستلین	کرینسکی	کرایلر	L-R-P	L-R-H
۱	۱۶	۵۲۳	۵۵۷	۵۵۵	۵۵۳	۵۵۱
۲	۲۰	۶۱۹	۶۳۰	۶۴۹	۶۳۴	۶۳۷
۳	۲۴	۷۴۹	۷۷۹	۷۶۰	۷۵۹	۷۶۹
۴	۲۸	۸۱۰	۸۴۰	۸۵۴	۸۳۰	۸۲۰
۵	۳۲	۹۰۲	۹۵۰	۹۳۲	۹۶۰	۹۵۴
۶	۳۶	۱۰۲۳	۱۲۶۰	۱۱۳۶	۱۲۵۰	۱۲۵۷
متوسط	۲۶	۷۷۱	۸۳۶	۸۱۴/۳	۸۳۱	۸۳۱/۳

جدول (۱۳) مقایسه الگوریتمهای WSA و MWSA برای مسئله شناسایی اعداد لاتین.

شبکه	تعداد نرونهای مخفی	الگوریتم WSA		الگوریتم MWSA		
		در صد تشخیص		تعداد وزنه‌های لایه مخفی	در صد تشخیص	
		آموزش	آزمایش		آموزش	آزمایش
۱	۱۵	۱۰۰٪	۱۰۰٪	۳۴۰	۱۰۰٪	
۲	۱۵	۱۰۰٪	۹۰٪	۳۶۷	۹۰٪	
۳	۱۷	۱۰۰٪	۱۰۰٪	۳۷۶	۹۰٪	
۴	۱۹	۱۰۰٪	۱۰۰٪	۴۲۳	۱۰۰٪	
۵	۲۱	۱۰۰٪	۱۰۰٪	۴۶۴	۱۰۰٪	
۶	۲۳	۱۰۰٪	۹۰٪	۴۸۴	۹۰٪	
متوسط	۱۸	۱۰۰٪	۹۶/۷٪	۴۰۹	۹۵٪	

نتایج شبیه‌سازی برای الگوریتم MWSA: برای هر مسئله الگوریتم MWSA برای شبکه‌های مختلف با تعداد مختلف نرونهای لایه مخفی پیاده‌سازی شده است. در این آزمایشها، مقدار $\lambda_{ON} = 1/5$ ، تعداد تکرار الگوریتم پس انتشار خطا در هر گام ۵۰ ($N=50$)، نرخ یادگیری ۰/۰۱ و ضریب ممنتم صفر در نظر گرفته شده است. تابع غیر خطی مورد استفاده در لایه مخفی و لایه خروجی، تابع سیگموئید میباشد. الگوهای آموزش ۵۰۰۰ بار به شبکه اعمال شده است. همچنین در این آزمایشها مقدار ۱/

$\varepsilon = 0$ و مقدار $\gamma = 0/5$ انتخاب شده است.

جدول (۱۴) مقایسه الگوریتمهای WSA و MWSA برای مسئله تقریب تابع غیر خطی زمان گسسته درجه دوم.

الگوریتم MWSA		الگوریتم WSA		تعداد نرونهای مخفی	شبکه
خطای شبکه عصبی به ازای ورودی سینوسی	تعداد وزنه‌های لایه مخفی	خطای شبکه عصبی به ازای ورودی سینوسی	تعداد وزنه‌های لایه مخفی		
۱/۱۲۲۶	۱۶	۰/۳۸۲۲	۲۳	۸	۱
۰/۹۸۰۹	۱۷	۰/۹۷۱۷	۲۵	۱۰	۲
۲/۲۹۸۲	۲۵	۲/۷۰۰۵	۳۲	۱۳	۳
۰/۹۳۲۵	۳۴	۰/۶۵۲۶	۴۱	۱۷	۴
۰/۹۰۱۲	۴۱	۰/۶۶۳۱	۴۷	۲۰	۵
۱/۵۵۸۰	۵۶	۱/۳۰۴۷	۶۳	۲۵	۶
۱/۴۴۹۵	۳۱/۵	۱/۱۱۲۵	۳۸/۵	۱۵/۵	متوسط

جدول (۱۵) مقایسه الگوریتمهای WSA و MWSA برای مسئله تشخیص اعداد چاپی فارسی.

الگوریتم MWSA			الگوریتم WSA			تعداد نرونهای مخفی	شبکه
در صد تشخیص		تعداد وزنه‌های لایه مخفی	در صد تشخیص		تعداد وزنه‌های لایه مخفی		
آزمایش	آموزش		آزمایش	آموزش			
%۹۹	%۹۸	۴۰	%۹۹	%۹۸	۵۹	۱۴	۱
%۹۳	%۹۷	۴۵	%۹۴	%۹۹	۶۰	۱۵	۲
%۹۲	%۹۶	۴۸	%۹۶	%۹۲	۶۱	۱۶	۳
%۹۸	%۹۱	۵۵	%۹۴	%۹۴	۷۰	۱۷	۴
%۹۳	%۹۷	۶۴	%۸۷	%۹۷	۸۰	۱۸	۵
%۹۱	%۹۹	۷۳	%۹۶	%۹۹	۸۹	۱۹	۶
%۹۴	%۹۶	۵۴	%۹۴	%۹۶/۵	۷۰	۱۶/۵	متوسط

تذکره ۶: نتایج شبیه‌سازیهای مختلف نشان می‌دهد الگوریتم جدید MWSA دارای عملکرد بهتری نسبت به الگوریتم WSA میباشد. الگوریتم پیشنهادی MWSA بدلیل اینکه در آن فرایند شاخه زنی تا زمانیکه منجر به افزایش خطای آموزش نگردیده است همچنان ادامه می‌یابد لذا در مقایسه با الگوریتم WSA که در آن عمل شاخه‌زنی تا زمانی ادامه می‌یابد که تعداد وزنها از یک حدی کمتر نشود دارای عملکرد بهتری میباشد. با اعمال این الگوریتم توانستیم برای مسئله شناسایی اعداد لاتین %۳۴، مسئله تشخیص اعداد چاپی فارسی %۲۳، مسئله تقریب تابع غیر خطی زمان گسسته درجه دوم %۱۸ و مسئله شناسایی

است که افزایش در صد شاخه زنی برای شبکه‌های پیچیده بیشتر می‌باشد. در بین مسائل مختلفی که برای ارزیابی الگوریتم MWSA مورد استفاده قرار گرفته است شبکه‌های مورد استفاده برای شناسایی فونمهای فارسی از بقیه شبکه‌ها پیچیده‌تر بوده و در این مورد این مسئله شاهد بیشترین افزایش در صد شاخه زنی ۴۵٪ می‌باشیم.

جدول (۱۶) مقایسه الگوریتمهای WSA و MWSA برای مسئله شناسایی فونمهای فارسی.

الگوریتم MWSA		الگوریتم WSA		تعداد وزنهای لایه مخفی	تعداد نرونهای مخفی	شبکه
در صد تشخیص		در صد تشخیص				
آموزش	آزمایش	آموزش	آزمایش	تعداد وزنهای لایه مخفی	تعداد نرونهای مخفی	
۸۳٪	۸۵٪	۸۷٪	۸۷٪	۹۴۵	۱۶	۱
۸۴٪	۸۸٪	۸۴٪	۸۷٪	۱۱۷ ۲	۲۰	۲
۸۶٪	۸۵٪	۸۲٪	۸۵٪	۱۴۱ ۸	۲۴	۳
۸۲٪	۸۷٪	۸۲٪	۸۳٪	۱۶۴ ۴	۲۸	۴
۸۷٪	۸۵٪	۸۱٪	۸۴٪	۱۸۵ ۵	۳۲	۵
۸۲٪	۸۶٪	۸۴٪	۸۲٪	۲۱۰ ۷	۳۶	۶
۸۴٪	۸۶٪	۸۳٪	۸۵٪	۱۵۲ ۳	۲۶	۷

۴-۸- مقایسه روشهای یادگیری ساختار پیشنهادی با روشهای موجود

برخی از روشهای مهم شاخه‌زنی که توسط محققین ارائه شده است بشرح زیر می‌باشد. کارنین [۴۶] روشی را ارائه کرده است که در آن روش حساسیت تابع خطا را نسبت به حذف هر اتصال حساب کرده سپس وزنهای باحساسیت پائین را حذف میکند. حساسیت وزن W_{ij} بصورت زیر بدست می‌آید.

$$S_{ij} = -\frac{E(w^f) - E(0)}{w^f - 0} \times w^f \quad (22)$$

که در آن w^f مقدار نهایی وزن پس از آموزش، 0 مقدار آن پس از حذف، $E(w^f)$ مقدار نهایی خطا پس از آموزش و $E(0)$ مقدار خطا موقعیست که وزن حذف میشود. در عمل بجای اینکه پس از آموزش شبکه هر کدام از وزنها را حذف کرده و حساسیت تابع خطا را نسبت به آن حساب کرد در طول آموزش یک تخمینی از S بصورت زیر محاسبه می‌گردد.

$$\hat{S}_{ij} = -\sum_1^N \frac{\partial E}{\partial w_{ij}} \cdot \Delta w_{ij}(n) \cdot \frac{w_{ij}^f}{w_{ij}^f - w_{ij}^i} \quad (23)$$

که در فرمول بالا، N تعداد epoch های آموزش و w^f مقدار اولیه وزن میباشد. تمامی این عبارتها در طول آموزش در

دسترس بوده و نیاز به محاسبه جداگانه حساسیت پس از پایان آموزش نمیباشد. پس از آموزش هر وزن دارای یک حساسیت تخمینی بوده و می‌توان وزنه‌های با حساسیت پائین را حذف کرد. دسته‌ای دیگر از روشها تابع خطا را طوری تغییر می‌دهد که در نتیجه الگوریتم پس انتشار خطا در حین آموزش تعدادی از وزنها را به سمت صفر میل میدهد. وزنه‌هایی که نهایتاً مقدار آنها از یک مقدار آستانه کمتر باشد حذف میشود. ایشی کاوا [۴۷] یک الگوریتم یادگیری ساختار به این صورت پیشنهاد کرد. تابع هدف این الگوریتم شامل قسمت خطا و قسمت جریمه میباشد. قسمت جریمه مجموع مقادیر قدر مطلق وزنها میباشد. یادگیری شامل دو مرحله میباشد. الگوریتم یادگیری با الگوریتم فراموشی در مرحله اول و الگوریتم یادگیری با الگوریتم انتخابی در مرحله دوم. تابع هدف E_{pf} در مرحله اول بصورت زیر تعریف میشود.

$$E_{pf} = \sum_j (t_{pj} - o_{pj})^2 + \varepsilon \sum_{i,j} |w_{ji}| \quad (24)$$

که در آن t_{pj} و o_{pj} به ترتیب الگوی هدف و الگوی خروجی واحد زام برای الگوی ورودی P ، w_{ji} وزن بین واحد زام در لایه پائینی با واحد زام در لایه بالایی و ε یک ثابت میباشد. روش گرادبان نزولی روی تابع هدف اعمال شده و قانون تغییر وزن بصورت زیر میباشد.

$$\Delta_p w_{ji}(t+1) = \eta \delta_{pj} o_{pi} + \alpha \Delta_p w_{ji}(t) - \varepsilon \operatorname{sgn}(w_{ji}) \quad (25)$$

که $\Delta_p w_{ji}(t)$ تغییراتی است که بایستی در w_{ji} ایجاد شود. و δ_{pj} خطای پس انتشار واحد زام برای الگوهای ورودی P ، η و $\alpha = \eta \varepsilon$ ثابت هستند. $\operatorname{sgn}(x)$ بصورت زیر تعریف میشود.

$$\operatorname{sgn}(x) = \begin{cases} 1 & \text{for } x > 0 \\ 0 & \text{for } x = 0 \\ -1 & \text{for } x < 0 \end{cases} \quad (26)$$

تابع هدف، E_{ps} در مرحله دوم بصورت زیر تعریف میشود.

$$E_{ps} = \sum_j (t_{pj} - o_{pj})^2 + \varepsilon \sum_{|w_{ji}| < \sigma} |w_{ji}| \quad (27)$$

که σ یک ثابت مثبت میباشد. بدین ترتیب یادگیری در مرحله دوم یادگیری فراموشی را فقط در مورد وزنه‌هایی که در یک محدوده مشخص قرار دارد اعمال میکند.

مراحل مختلف الگوریتم بصورت زیر میباشد.

یک ساختار مناسب برای شبکه انتخاب کن

شبکه را با استفاده از الگوریتم فراموشی آموزش بده.

شبکه را با استفاده از الگوریتم انتخابی آموزش بده

روش دیگری توسط لی کان [۵] تحت عنوان OBD ارائه گردیده است. این الگوریتم دسته‌ای از وزنها را پیدا میکند که

حذف این وزنها افزایش خیلی کمی را در تابع هدف ایجاد میکند.

الگوریتم OBD بصورت زیر میباشد.

یک ساختار مناسب یک شبکه را انتخاب کن

شبکه را با الگوریتم پس انتشار خطا آموزش بده

اهمیت هر وزن را که با S_{ji} نشان داده میشود طبق فرمول زیر برای تمامی وزنها حساب کن

$$S_{ji} = \frac{1}{2} \frac{\partial^2 E}{\partial w_{ji}^2} \cdot w_{ji}^2 \quad (28)$$

که در آن E تابع هدف میباشد.

وزنهای کم اهمیت را حذف کن

تا زمانی که حذف وزنها باعث افزایش خطا نشده به مرحله ۲ برو.

در این بخش الگوریتمهای جدید AWSA و MWSA را با روشهای موجود مقایسه میکنیم. بمنظور مقایسه عملکرد الگوریتمهای پیشنهادی AWSA و MWSA با روشهای مهم حذف وزن آزمایشهایی را ترتیب داده ایم. آزمایشها بر روی مسائل شناسایی اعداد لاتین، بازنسای اعداد چایی فارسی، تقریب تابع غیر خطی زمان گسسته از مرتبه دوم و بازنسای فونمهای فارسی صورت گرفته است. نتایج آزمایشهای مختلف در جداول ۱۷ الی ۲۰ نشان داده شده است.

الگوریتم کارنین: برای هر کاربردی الگوریتم هرس کارنین برای تعداد مختلف نرونهای لایه مخفی پیاده سازی شده است. در این آزمایشها نرخ یادگیری ۰/۰۱ و ضریب گشتاور صفر در نظر گرفته شده است. از تابع سیگموئید بعنوان توابع غیر خطی لایه های مخفی و خروجی استفاده شده است. همچنین در این الگوریتم مقدار آستانه برای حساسیت وزنها ۰/۰۱ انتخاب شده است. وزنهایی که حساسیت تابع خطا نسبت به آن کمتر از ۰/۰۱ باشد وزنهای غیر مهم تلقی شده و حذف شده اند.

الگوریتم یادگیری ساختار با فراموشی: برای هر مسئله، الگوریتم فراموشی برای تعداد مختلف نرونهای لایه مخفی پیاده سازی شده است. در تمامی آزمایشها نرخ یادگیری ۰/۰۱ و ضریب گشتاور صفر انتخاب شده است. از تابع سیگموئید بعنوان توابع غیر خطی لایه های مخفی و خروجی استفاده شده است. مقدار $\epsilon = 1 \times 10^{-5}$ و مقدار $\sigma = 0.01$ انتخاب شده است.

الگوریتم OBD: برای کاربردهای مختلف الگوریتم OBD برای تعداد مختلف نرونهای لایه مخفی اجرا شده است. در تمامی آزمایشها نرخ یادگیری ۰/۰۱ و ضریب گشتاور صفر انتخاب شده است. از تابع سیگموئید برای تابع غیر خطی لایه های خروجی استفاده شده است. مقدار آستانه برای پارامتر S_{ji} که بیان کننده اهمیت وزن w_{ji} میباشد $\gamma = 0/01$ انتخاب شده است.

جداول ۱۷ الی ۲۰ نتایج شبیه سازیها را برای هر کدام از الگوریتمها و برای کاربردهای مختلف را نشان میدهند.

تذکره ۷: با توجه به نتایج آزمایشات ملاحظه می کنیم که الگوریتم AWSA دارای بیشترین قدرت شاخه زنی میباشد. در این روش با توجه به اینکه با تنظیم پارامترهای ضریب پهنای روشنی و ضریب پهنای خاموشی در طول فرایند آموزش نرخ شاخه زنی و نرخ اضافه کردن وزنها را بطور تطبیقی تغییر میدهیم لذا میتوانیم به بیشترین درصد شاخه زنی بدون واگرایی شبکه دست پیدا کنیم.

جدول (۱۷) مقایسه الگوریتمهای مختلف شاخه زنی برای مسئله شناسایی اعداد لاتین.

شبکه	تعداد نرونهای مخفی	تعداد وزنهای لایه مخفی				
		AWSA	MWSA	WSA	فراموشی ساختار	OBD
۱	۱۵	۲۰۶	۳۴۰	۴۹۹	۵۰۴	۵۶۰
۲	۱۵	۱۸۷	۳۶۷	۵۳۴	۵۱۰	۵۹۰
۳	۱۷	۱۷۷	۳۷۶	۵۵۸	۵۳۰	۶۲۰
۴	۱۹	۱۹۲	۴۲۳	۶۲۷	۶۴۵	۶۲۵
۵	۲۱	۱۷۳	۴۶۴	۷۱۴	۷۱۰	۶۸۰
۶	۲۳	۲۶۱	۴۸۴	۷۷۲	۷۳۰	۶۹۵
میانگین	۱۸	۱۹۹	۴۰۹	۶۱۹	۶۰۵	۶۲۸

جدول (۱۸) مقایسه الگوریتمهای مختلف شاخه‌زنی برای مسئله تقریب تابع غیرخطی زمان گسسته درجه دوم.

شماره شبکه	تعداد نرونهای مخفی	تعداد وزنهای لایه مخفی					
		کارترین	OBD	یادگیری ساختاربا فراموشی	WSA	MWSA	AWSA
۱	۸	۲۵	۲۵	۲۲	۲۳	۱۶	۱۹
۲	۱۰	۲۷	۲۷	۲۶	۲۵	۱۷	۲۲
۳	۱۳	۲۹	۳۷	۳۴	۳۲	۲۵	۲۸
۴	۱۷	۵۰	۴۸	۴۳	۴۱	۳۴	۳۸
۵	۲۰	۶۳	۵۷	۴۹	۴۷	۴۱	۴۳
۶	۲۵	۸۹	۷۷	۶۵	۶۳	۵۶	۵۷
متوسط	۱۵/۵	۴۹	۴۵	۷۰	۲۸/۵	۳۱/۵	۳۴/۵

جدول (۱۹) مقایسه الگوریتمهای مختلف شاخه‌زنی برای مسئله تشخیص اعداد چاپی فارسی.

شماره شبکه	تعداد نرونهای مخفی	تعداد وزنهای لایه مخفی					
		کارترین	OBD	یادگیری ساختاربا فراموشی	WSA	MWSA	AWSA
۱	۱۴	۹۰	۸۵	۶۴	۵۹	۴۰	۲۵
۲	۱۵	۹۴	۸۱	۶۰	۶۰	۴۵	۴۰
۳	۱۶	۹۹	۸۵	۶۵	۶۱	۴۸	۴۳
۴	۱۷	۱۰۹	۹۰	۷۹	۷۰	۵۵	۵۰
۵	۱۸	۱۲۱	۹۵	۸۵	۸۰	۶۴	۵۵
۶	۱۹	۱۲۳	۱۰۱	۸۷	۸۹	۷۳	۶۰
متوسط	۱۶/۵	۱۰۶	۸۹/۵	۷۳	۶۹/۸	۵۴	۴۷

جدول (۲۰) مقایسه الگوریتمهای مختلف شاخه‌زنی برای مسئله شناسایی فونمهای فارسی.

شماره شبکه	تعداد نرونهای مخفی	تعداد وزنهای لایه مخفی					
		کارترین	OBD	یادگیری ساختاربا فراموشی	WSA	MWSA	AWSA
۱	۱۶	۱۰۲۵	۱۰۲۰	۹۶۰	۹۴۵	۵۴۰	۵۲۳
۲	۲۰	۱۲۷۰	۱۲۵۰	۱۲۰۱	۱۱۷۲	۶۳۴	۶۱۹
۳	۲۴	۱۵۲۰	۱۴۹۰	۱۴۰۳	۱۴۱۸	۷۵۸	۷۴۹
۴	۲۸	۱۷۴۳	۱۷۳۰	۱۷۲۰	۱۶۴۴	۸۸۹	۸۱۰
۵	۳۲	۲۱۰۰	۲۰۱۳	۱۹۳۰	۱۸۵۵	۱۰۴۰	۹۰۲
۶	۳۶	۲۴۹۴	۲۳۸۴	۲۳۰۲	۲۱۰۷	۱۱۳۴	۱۰۲۳
متوسط	۲۶	۱۶۹۲	۱۶۴۸	۱۵۸۶	۱۵۲۳	۸۳۳	۷۷۱

تذکر ۸: الگوریتم‌های کارنین و OBD شبکه را با استفاده از الگوریتم پس انتشار خطا آموزش داده و سپس وزنه‌های بی‌اهمیت را شناسایی کرده اقدام به حذف آنها می‌نمایند. لذا این الگوریتم‌ها در حین آموزش پیچیدگی محاسبات را کاهش نمیدهند. یکی از معضلاتی که گریبانگیر محققین فعال در زمینه شبکه‌های عصبی میباشد حجم بسیار بالای محاسبات و در نتیجه طولانی شدن زمان یادگیری میباشد. لذا الگوریتم‌های یادگیری ساختاری که در حین آموزش پیچیدگی شبکه را کاهش دهند از این نظر دارای برتری میباشند. چرا که با کاهش پیچیدگی شبکه نه تنها زمان محاسبات تقلیل پیدا میکند بلکه در پیاده‌سازی موازی شبکه‌های عصبی بر روی کامپیوترهای موازی با کاهش وزنها و نهایتاً حذف نرونهای بی‌اهمیت حجم ارتباطات بین پروسورها نیز کاهش یافته در نتیجه میتوان به Speed up های بالاتری دست یافت.

الگوریتم‌های MWSA، AWSA، WSA و الگوریتم یادگیری با فراموشی در حین آموزش، وزنه‌های بی‌اهمیت را حذف و باعث کاهش پیچیدگی شبکه میگردند که از این لحاظ برای پیاده‌سازی موازی شبکه‌های عصبی مناسب میباشد.

تذکر ۹: در الگوریتم‌های MWSA، AWSA و WSA که مبتنی بر اتوماتانهای یادگیر میباشند. پس از آنکه برای N بار متوالی که N عمق حافظه اتوماتان میباشد. قدر مطلق وزنی از یک مقدار آستانه کوچکتر بود آنگاه آن وزن حذف میشود. این نحوه حذف وزن با احتمال زیاد وزنهایی را حذف میکند که مطمئناً در epoch های متوالی قدر مطلق آنها دائماً در حال کاهش بوده و نقش آنها در طبقه‌بندی الگوها ناچیز بوده و میتوان آنها را براحتی حذف کرد. در طی فرایند آموزش ممکن است وزنی در مقطعی از آموزش دارای قدر مطلق کم بوده ولی بعداً افزایش یابد لذا حذف چنین وزنهایی میتواند باعث افزایش خطا شود.

نتیجه گیری

در این مقاله دو الگوریتم جدید تحت عنوان AWSA و MWSA ارائه گردیده است. الگوریتم AWSA با استفاده از اتوماتانهای یادگیر بازه‌های تصمیم‌گیری مورد استفاده در الگوریتم تعیین تعداد وزن WSA را تنظیم میکند. در این الگوریتم برای پاسخ دادن به اتوماتان مربوط به تنظیم بازه‌های تصمیم‌گیری از معیار خطا کمک گرفته‌ایم. در الگوریتم MWSA معیارهای جدیدی برای ارزیابی عملکرد وزنهایی روشن و خاموش و شرایط روشن کردن وزنهایی خاموش ارائه گردیده است. این الگوریتمها شبکه‌هایی با اتصالات محلی که دارای پیچیدگی پائین و قدرت تعمیم بالایی هستند تولید میکنند. در بخش دیگری از مقاله اتوماتانهای مختلف برای تنظیم بازه‌های تصمیم‌گیری با هم مقایسه گردیده‌اند. از طریق شبیه‌سازی نشان داده شده است در اکثر موارد در بین اتوماتانهای با ساختار ثابت، اتوماتان ستیلین دارای عملکرد بهتری بوده اتوماتان کرنیسکی با توجه به اینکه در آن تغییر اقدام با تاخیر صورت می‌گیرد دارای عملکرد ضعیفی بوده و اتوماتان کرایلف بدلیل تصادفی بودن عملکرد این نوع اتوماتان در بعضی مواقع دارای عملکرد خوب و در برخی از مواقع دارای عملکرد بد میباشد. الگوریتم‌های پیشنهادی AWSA و MWSA بر روی مسائل شناسایی اعداد انگلیسی، شناسایی اعداد چایی فارسی، تقریب تابع غیر خطی زمان گسسته از مرتبه دوم و باز شناسی فونمنهای فارسی پیاده‌سازی شده است. نتایج شبیه‌سازیها نشان میدهد الگوریتم‌های پیشنهادی AWSA و MWSA دارای عملکرد بهتری نسبت به الگوریتم‌های WSA، کارنین، OBD و الگوریتم یادگیری با فراموشی میباشد.

زیر نویس‌ها

- | | |
|---|--|
| 1-Weight Survival Algorithm | 12- Linear Reward-Inaction |
| 2-Adaptive Weight | 13-Weight Survival Algorithm |
| 3-Survival Algorithm | 14-Adaptive Weight Survival Algorithm. |
| 4- Modified Weight Survival Algorithm | 15- Modified Weight Survival Algorithm |
| 5-Net Input | 16- Weight survival Algorithm |
| 6- Learning Automata | 17-Modified weicht survial algorithm |
| 7- Fixed Structure Learning Automata | 18- FarsDat |
| 8- Variable Structure Learning Automata | 19-Time Delay |
| 9- Tsetlin | 20- forgetting algrithm |
| 10- Stochastic | 21-selective algorithm |
| 11-Linear Reward-Penalty | |

- [1] Tin-Yau Kwok and Dit-Yan Yeung, "Constructive Algorithms for Structure Learning in Feedforward Neural Networks for Regression Problems," *IEEE Trans. Neural Networks*, vol. 8, NO. 3, pp. 630-645, MAY 1997.
- [2] M. C. Mozer and P. Smolensky, "Skeletonization: A Technique for Trimming the Fat from a network via Relevance Assessment," in *Advances in Neural Information Processing (1)*, D.S. Touretzky, Ed. (Denver 1988), 1989, pp. 107-115.
- [3] B. E. Segee and M. J. Carter, "Fault Tolerance of Pruned Multilayer Networks," in *Proc. Int. Joint Conf. Neural Networks*, vol. II (Seattle), pp. 447- 452, 1991.
- [4] E. D. Karnin, "A Simple Procedure for Pruning Backpropagation Trained Neural Networks," *IEEE Trans. Neural Networks*, vol. 1, no. 2, pp. 239-242, 1990.
- [5] Y. Le Cun, J. S. Denker, and S. A. Solla, "Optimal Brain Damage," in *Advances in Neural information Processing (2)*, D. S. Touretzky, Ed. (Denver 1989), 1990, pp. 598-605.
- [6] A. S. Weigend, D. E. Rumelhart, and B. A. Hube-rman, "Generalization by Weight-Elimination with Application to Forecasting," in *Advances in Neural Information Processing (3)*, R. Lippmann, J. Moody, and D. Touretzky, Eds., 1991, pp. 875-882.
- [7] A. S. Weigend, D. E. Rumelhart, and B. A. Hube-rman, "Backpropagation, Weight-Elimination and Time Series Prediction," in *Proc. 990 Connectionist Models Summer School*, D. Touretzky, J. Elman, T. Sejnowski, and G. Hinton, Eds., 1990, pp. 105-116.
- [8] Y. Chauvin, "A Backpropagation Algorithm with Optimal Use of Hidden Units," in *Advances In Neural Information (1)*, D. S. Touretzky, Ed. (Denver 1988), 1989, pp. 519-526.
- [9] S. J. Hanson and L. Y. Pratt, "Comparing Biases for Minimal Network Construction with backpropagation," in *Advances in Neural Information Processing (1)*, D.S. Touretzky, Ed. (D-enver 1988), 1989, pp. 177-185.
- [10] Y. Chauvin, "Dynamic Behavior of Constrained Backpropagation Networks," in *Advances in Neural Information Processing (2)*, D. S. Touretzky, Ed. (Denver 1989), 1990, pp. 642- 649.
- [11] C. Ji, R. R. Snapp, and D. Psaltis, "Generalizing Smothness Constrains from Discrete Samples," *Neural Computation*, vol. 2, no. 2, pp. 188-197, -1990.
- [12] D. C. Plaut, S. J. Nowlan, and G. E. Hinton, "E-xperiments on Learning by Backpropagation," T-ech. Rep. CMU-CS-86-126, Carnegie-Mellon U-niv., 1986.
- [13] S. J. Nowlan and G. E. Hinton, "Simplifying Neural Networks by Soft Weight-Sharing," *Neural Computation*, vol. 4, pp. 473-493, 1992.
- [14] Marco Muselli, "On Sequential Construction of-Binary Neural Networks," *IEEE Trans. Neural-Networks*, vol. 6, no. 3, pp. 678-690, MAY. 1995.
- [15] Steven Young and Tom Downs, "CARVE-A Co-nstructive Algorithms for Real-Valued Example-s," *IEEE Trans. Neural Networks*, vol. 9, no. 6, pp. 1180-1190, NOV. 1998.
- [16] John O. Moody and Panos J. Antsaklis, "The De-pendence Identification Neural Network Constr-uction Algorithm," *IEEE Trans. Neural Networ-ks*, vol. 7, no. 1, pp. 3-15, Jan. 1996.
- [17] F. M. Frattale Mascioli and G. Martinelli, "A Constructive Algorithm for Binary Neural Net-works: The Oil-Sport Algorithm," *IEEE Trans. Neural Networks*, vol. 6, no. 3, pp. 794-797, -MAY. 1995.
- [18] Rudy Setiono and Lucas Chi Kwong Hui, "Use of a Quasi-Newton Method in a Feedforward Neur-al Network Construction Algorithm," *IEEE Tra-ns. Neural Networks*, vol. 6, no. 1, pp. 273-277, JAN. 1995.
- [19] Nabhan T. M. and Zomaya A. Y., "Toward Neu-ral Networks Structure for Function Approximat-ion," *Neural Networks*, vol. 7, no.1, pp. 89-99, 1993.
- [20] Hirose Y., Yamashita K. and Hijya S., "Backpro-pagation Algorithm which Varies The Number of Hidden Units," *Neural Networks*, vol. 4, no. 1, pp. 61-66, 1991.
- [21] J. D. Schaffer, D. Whitely and L. J. Eshelman, "Combination of Genetic Algorithms and Neural Networks: A Survey of the State of the Art," *IEEE Proc. COGANN-92*, pp. 1-37, 1992.
- [22] V. Maniezzo, "Genetic Evolution of the Topolo-gy and Weight Distribution of Neural Networks-," *IEEE Trans. On Neural Networks*, vol. 5, no. 1, pp. 39-53, 1994.
- [23] P. J. Angeline, G. M. Saunders and J. B. Pollack, "Evolutionary Algorithms that Construct Recur-rent Neural Networks," *IEEE Trans. on Neural Networks*, vol. 5, no. 1, pp. 54-65, 1994.
- [24] X. Yao and Y. Liu, "A New Evolutionary Syst-em for Evolving Artificial Neural Networks," *IEEE Trans. On Neural Networks*, vol. 8, no. 3, pp. 694-713, 1997.
- [25] D. Whitely and C. Bogart, "The Evolution of C-onnectivity: Pruning Neural Networks Using Genetic Lagorithms," *Proc. Of Int. Joint Conf. on Neural Networks*, vol. 1, pp. 134, 1990.
- [26] Beigy. H. and Meybodi, M. R., "Optimization of Topology of Neural Networks Using Learning Automata," *Proc. Of 4th Annual Int. Computer Society of Iran Computer Conf. CSICC-98*, Teh-ran, Iran, pp. 417-428 (In Persian) 1999. (Also appear in *Journal of Amirkabir*)
- [27] M. B. Menhaj, "Computational Intelligence (vol. 1) Fundamentals of Neural Networks", Professor Hessabi

Publication, IRAN, Tehran, 1998.

- [28] Y. Chauvin, "Generalization Performance of Overtrained Backpropagation Networks," in Neural Networks, Proc. EUROSIP Workshop, L. B. Al-Media and C. J. Wellekens, Eds., Feb. 1990, pp. 46-55.
- [29] E. Levin, N. Tishby, and S. A. Solla, "A Statistical Approach to Learning and Generalization in Layered Neural Networks," Proc. IEEE, vol. 78, no. 10, pp. 1568-1574, Oct. 1990.
- [30] D. B. Schwartz, V. K. Samalan, S. A. Solla, and J. S. Denker, "Exhaustive learning," Neural Computation, vol. 2, no. pp. 374-385, 1990.
- [31] N. Tishby, E. Levin, and S. A. Solla, "Consistent Inference of Probabilities in Layered Networks: Predictions and Generalization," in Proc. Int. Joint Conf. neural Networks, 1989, p. 403.
- [32] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth, "Learnability and the Vapnik-Chervon-Nenkis Dimension," J. Ass. Comput. Mach., vol. 36, no. 4, pp. 929-965, 1989.
- [33] A. Ehrenfeucht, D. Haussler, M. Kearns, and L. Valiant, "A General Lower Bound on the Number of Examples Needed for Learning." In Proc. 1988 Workshop Computational Learning theory, 1988.
- [34] L. G. Valiant, "A Theory of the Learnable," Commun. Ass. Comput. Mach., vol. 27, no. 11, pp. 1134-1142, 1984.
- [35] K. S. Narendra and M. A. L. Thathachar, Learning Automata: An Introduction, Prentice Hall, Englewood cliffs, 1989.
- [36] M. R. Meybodi and S. Lakshmivarahan, "Optimality of a General Class of Learning Algorithm", Information science, Vol. 28, pp. 1-20, 1982.
- [37] M. R. Meybodi and S. Lakshmivarahan, "On a Class of Learning Algorithm which have a Symmetric Behavior under Success and Failure", Springer Verlag Lecture Notes in Statistics, pp. 145-155, 1984.
- [38] M. R. Meybodi, "Results on a Strongly Absolutely Expedient Learning Automata", Proc. Of OU Inference Conf. 86, ed. D. R. Mootes and R. Butrick, Athens, Ohio: Ohio University Press, pp. 197-204.
- [39] R. Reed, "Pruning algorithms-A survey," IEEE Trans. Neural Networks, vol. 4, no. 5, pp. 740-747, Sept. 1993.
- [40] Oommen, B. J. and Croix, E. V. de St. (1996). Graph Partitioning Using Learning Automata, IEEE Trans. On Computers, No. 45, No.2, pp. 195-208.
- [41] Oommen, B. J. and Ma, D. C. Y. (1988). Deterministic Learning Automata Solutions to the Equipartitioning Problem, IEEE Trans. On Computers, No. 37, No. 1, pp. 2-13.
- [42] Oommen, B. J., Valiveti, R.S. and Zgierski, J. R. An Adaptive Learning Solution to the Keyboard Optimization Problem, IEEE Trans. On Systems, Man, and Cybernetics, Vol. 21, No. 6, pp. 1609-1618, (1991).
- [43] Meybodi, M. R. and Beigy. H. (2000), "Neural Networks Engineering Using Learning Automata: Determining Optimal size for three layer Neural Networks" Proc. Of 5th Annual Int. Computer Society of Iran Computer Conf. CSICC-99, Tehran, Iran, pp. 431-450, Iran (In Persian). Also will appear in journal of Daneshkadeh Fanni, 2001.
- [44] S. M. Ahadi, "Reduced Context Sensitivity in Persian Speech Recognition Via Syllable Modeling", Proceedings of The Eighth Australian International Conference on Speech Science and Technology, Cabberra, pp. 492-497, December 2000.
- [45] Urs A. Muller, Anton Gunzinger and Walter Guggenbuhl, "Fast Neural Net Simulation with a DSP Processor Array", IEEE Trans. On Neural Networks, Vol. 6, No. 1, pp. 203-213, January 1995.
- [46] E. D. Karnin, "A simple procedure for pruning back-propagation trained neural networks," IEEE Trans. Neural Networks, Vol. 1, no. 2, pp. 239-242, 1990.
- [47] M. Ishikawa, "A structural learning algorithm with forgetting of link weights," Tech. Rep. TR-90-7, Electrotechnical Lab., Tsukuba-City, Japan, 1990.
- [48] ____, "Neural Networks Engineering Using Learning Automata: Introducing an Adaptive Algorithm for Determining number of hidden Layer neurons for three layer neural networks", Proc. Of ICEE-2001, the 9th Iranian Conference on Electrical Engineering, pp. 27: 1-27: 14, Power and Water Institute of Technology, Tehran, Iran, 2001 (In Persian).
- [49] ____, "Adaptive VLRBP Using Learning Automata for Neural Network", Proc. Of WSES International Conference on: Neural Networks and Applications, pp. 280-286, Puerto De La Cruz, Tenerife, Canary Islands SPAIN February 11-15, 2001.
- [50] ____, "Introducing an Adaptive VLRBP Algorithm Using Learning Automata for Multilayer Neural Network", accepted in IEICE (The Institute of Electronics, Information and Communication Engineers) Transactions on Information and Systems.
- [51] ____, "Adaptive Survival Algorithm", submitted to engineering sciences journal, Iran University of Science & Technology, Narmak, TEHRAN.
- [52] N. Baba and K. Sato, "A consideration on the Learning Algorithm of Neural Network", Proc. Of IEEE, 1998.