

الگوریتم بقا تعمیم یافته: الگوریتمی جهت تعیین ساختار شبکه‌های عصبی چندلایه

⁺ مجید انجیدنی ^{++*} محمد رضا میبدی

⁺ آزمایشگاه محاسبات نرم، دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه صنعتی امیرکبیر، تهران، ایران
^{*} مرکز تحقیقات فیزیک نظری، پژوهشکده علوم کامپیوتر، تهران، ایران

(anjidani, meybodi)@ce.aut.ac.ir

لست [۱۲][۱۳][۱۴]. دو نسخه متفاوت از الگوریتم بقا موجود است که یکی برای تعیین حداقل تعداد نورونها (الگوریتم بقا نورو) و دیگری برای تعیین حداقل تعداد وزنها (الگوریتم بقا وزن) به کار برده می‌شود. الگوریتم بقا از یک اتوماتای یادگیر مهاجرت اشیاء به عنوان ابزار جستجوی عمومی و الگوریتم یادگیری انتشار خطا به عقب استفاده می‌کند و در حین آموزش، یک ساختار مناسب برای شبکه عصبی سه لایه (ساختاری که دارای اندازه کوچک، پیچیدگی آموزش کم و قدرت تعمیم بالا باشد) تعیین می‌نماید. در الگوریتم بقا آموزش از یک شبکه عصبی سه لایه بزرگ شروع شده و اتوماتای یادگیر با افزودن و کاستن نورونهای مخفی، تعداد نورونهای لایه مخفی و یا وزنهای این شبکه را تعیین می‌کند. به دلیل استفاده از روشهای جستجوی عمومی (اتوماتاهای یادگیر)، امکان گرفتاری در مینیممهای محلی کاهش می‌یابد. قبلا اتوماتاهای یادگیر برای برای تطبیق پارامترهای شبکه‌های عصبی مورد استفاده قرار گرفته است [۱۶][۱۷][۱۸][۱۹][۲۰].

الگوریتم بقا اصلاح شده نیز برای شبکه‌های عصبی سه لایه توسط انجیدنی و میبدی ارائه شده است [21]. در این مقاله یک نسخه تعمیم یافته از الگوریتم بقا اصلاح شده برای شبکه‌های عصبی با تعداد لایه میانی دلخواه (ساختار لایه‌ای دلخواه) ارائه می‌گردد. نسخه تعمیم یافته الگوریتم بقا علاوه بر امکان استفاده از اتوماتاهای دلخواه (در این مقاله از اتوماتای یادگیر مهاجرت اشیاء و اتوماتای یادگیر کرایلو استفاده می‌شود) و کاهش فعالیت نورو به روش اصلاح شده، قادر به تعیین ساختار شبکه‌های با ساختار لایه‌ای دلخواه می‌باشد. همچنین در نسخه تعمیم یافته، استراتژیهای نیز جهت فعال سازی اتوماتاها در لایه‌ها مطرح می‌شود که بسته به شرایط، ممکن است یکی از آنها مناسب باشد. الگوریتم بقا تعمیم یافته برای شبکه‌های عصبی شامل دو و سه لایه میانی با دو مورد از الگوریتمهای هرس (S&D و تکراری) مقایسه می‌گردد. الگوریتمها بر روی مسائل ارقام انگلیسی، ارقام دست نویس فارسی آزمایش گردیده‌اند. نتایج آزمایشها برتری الگوریتم بقا تعمیم یافته در شبکه‌های با دو و سه لایه میانی را در مقایسه با دو

چکیده: در این مقاله یک نسخه تعمیم یافته از الگوریتم بقا نورو^۱ اصلاح شده برای تعیین ساختار شبکه‌های عصبی با تعداد لایه دلخواه ارائه و با دو الگوریتم به نامهای S&D^۲ و تکراری^۳ مقایسه می‌گردد. نتایج آزمایشها برتری الگوریتم بقا تعمیم یافته را در شبکه‌های با دو و سه لایه میانی در مقایسه با دو الگوریتم هرس ذکر شده نشان می‌دهد. **واژه‌های کلیدی:** شبکه‌های عصبی چند لایه، انتشار خطا به عقب، الگوریتم تنظیم ساختار شبکه، اتوماتای یادگیر

۱- مقدمه

تعداد لایه‌های مخفی، تعداد نورونها در هر لایه مخفی و وزنهای آن در شبکه‌های عصبی چند لایه تاثیر بالایی بر روی کارایی آنها دارد. یک شبکه با اندازه کوچک، خروجی دقیق تولید نمی‌کند و شبکه‌ای با اندازه بزرگ، بسیار کند و پرهزینه خواهد بود و برای تعمیم مجموعه آموزشی، نیاز به مجموعه آموزشی بزرگی خواهد داشت. طراحی یک شبکه یا ساختار بهینه یک مسئله NP-Hard است [۱]. بهمین جهت بیشتر الگوریتمهای ارائه شده برای تعیین ساختار شبکه‌های عصبی، الگوریتمهای تقریبی هستند. این الگوریتمها قبل، در حین یا بعد از یادگیری، ساختار مناسبی برای شبکه تعیین می‌نمایند. بعضی از این الگوریتمها از اطلاعات محلی و بعضی دیگر از اطلاعات عمومی برای یافتن ساختار مناسب شبکه استفاده می‌کنند. این الگوریتمها را می‌توان به پنج گروه عمده الگوریتمهای هرس^۴ [۲][۳][۴][۵]، الگوریتمهای سازنده^۵ [۶][۷]، الگوریتمهای ترکیبی^۶ [۸][۹]، الگوریتمهای تکاملی^۷ [۱۰][۱۱] و الگوریتمهای مبتنی بر اتوماتاهای یادگیر گروه بندی کرد. تنها الگوریتم گزارش شده بر اساس اتوماتاهای یادگیر الگوریتم بقا نام دارد که توسط بیگی و میبدی ارائه گردیده

^۱Neuron Survival Algorithm (NSA)

^۲Sietma and Dow

^۳Iterative

^۴Pruning Algorithms

^۵Constructive Algorithms

^۶Hybrid Algorithms

^۷Evolutionary Algorithms

الگوریتم هرس ذکر شده نشان می‌دهد. همچنین اتوماتای کرایلو از میان اتوماتاهای ساختار ثابت به عنوان بهترین گزینه برای تعیین ساختار شبکه‌های با دو و سه لایه میانی معرفی می‌شود. ادامه مقاله بصورت زیر سازماندهی شده است. در بخش ۲ اتوماتای یادگیر و انواع آن معرفی می‌گردد. سپس در بخش ۳ الگوریتم بقاء با ذکر اصلاحات انجام گرفته در آن شرح داده می‌شود. نتایج آزمایشها و نتیجه گیری نیز در بخشهای بعدی ارائه خواهند شد.

۲- اتوماتاهای یادگیر و الگوریتم انتشار خطا به عقب

۲-۱- اتوماتاهای یادگیر^۹

اتوماتای یادگیر یک ماشین با حالات محدود^۹ است که میتواند تعدادی محدود عمل را انجام دهد. هر عمل انتخاب شده توسط محیطی احتمالی ارزیابی می‌گردد و پاسخی به اتوماتای یادگیر داده می‌شود. اتوماتای یادگیر از این پلنک استفاده مینماید و عمل خود را برای مرحله بعد انتخاب می‌کند. در طی این فرایند، اتوماتای یادگیر یاد می‌گیرد که چگونه بهترین عمل را انتخاب نماید. اتوماتاهای یادگیر به دو گروه با ساختار ثابت و با ساختار متغیر تقسیم بندی می‌گردند. برای اطلاعات بیشتر در باره اتوماتاهای یادگیر با ساختار متغیر و اتوماتاهای یادگیر با ساختار ثابت مانند $G_{2N,2}$ ، $L_{2N,2}$ ، کرینسکی^{۱۰} و کرایلو^{۱۱} که در این مقاله از آنها استفاده شده است می‌توان به مراجع [۲۳] و [۱۵] مراجعه نمود.

۲-۳- الگوریتم انتشار خطا به عقب^{۱۲}

این الگوریتم یک الگوریتم بازگشتی کاهش گرادیان است که برای آموزش شبکه‌های عصبی پیشخور مورد استفاده قرار می‌گیرد. قانون کاهش گرادیان که این الگوریتم بر اساس آن کار می‌کند به صورت زیر است:

$$W(n+1) = W(n) + \eta G(n) + \alpha[W(n) - W(n-1)]$$

که W بردار وزن، n تعداد تکرار، η نرخ یادگیری، α منتم و G گرادیان تابع خطا می‌باشد که به صورت زیر محاسبه می‌شود

$$G(n) = -\nabla E_p(n)$$

E_p برابر مجموع مربعات خطا میباشد و به ورت زیر محاسبه میشود.

$$E_p(n) = \frac{1}{2} \sum_{j=1}^{\# \text{ outputs}} [T_{p,j} - O_{p,j}]^2 \quad \text{for } p = 1, 2, \dots, \# \text{ patterns}$$

بطوریکه $T_{p,j}$ و $O_{p,j}$ به ترتیب خروجی خواسته شده^{۱۳} و واقعی^{۱۴} برای الگوی p در نورون خروجی j هستند. کارایی این الگوریتم برای یک کاربرد خاص به میزان زیادی به توپولوژی شبکه (تعداد لایه‌ها، تعداد نورونها در هر لایه و اتصالات میان لایه ای) وابسته است.

۳- الگوریتم بقاء نورون

در الگوریتم بقاء نورون از یک اتوماتای یادگیر مهاجرت اشیاء برای تعیین تعداد نورونهای لایه مخفی یک شبکه سه لایه استفاده شده است. وظیفه این اتوماتای یادگیر تقسیم بندی نورونهای لایه مخفی به دو گروه روشن و خاموش می‌باشد. این اتوماتای یادگیر به صورت شش تایی $\langle \alpha, H, \Phi, \beta, F, G \rangle$ نشان داده می‌شود که در آن $\alpha = \{\alpha_1, \alpha_2\}$ اقدامهای اتوماتا یادگیر می‌باشد. اتوماتای یادگیر دارای دو اقدام است: اقدام شماره یک، اقدام متناسب یا واحدهای روشن نامیده می‌شود. نورونهایی که در وضعیتهای این اقدام واقع شوند برای آموزش شبکه مورد استفاده قرار می‌گیرند. اقدام شماره دو، اقدام نامناسب یا واحدهای خاموش نام دارد. $H = \{H_1, H_2, \dots, H_n\}$ نورونهای مخفی هستند که روشن و خاموش کردن آنها به عهده اتوماتای یادگیر می‌باشد. اگر نورون H_i در اقدام شماره یک ظاهر شود به معنای روشن بودن آن و در غیر این صورت خاموش خواهد بود. $\Phi = \{\Phi_1, \Phi_2, \dots, \Phi_{2N}\}$ حالت‌های اتوماتای یادگیر بوده و N عمق حافظه می‌باشد. حالت‌های اتوماتای یادگیر به دو گروه تقسیم می‌شوند: $\{\Phi_1, \Phi_2, \dots, \Phi_N\}$ و $\{\Phi_{N+1}, \Phi_{N+2}, \dots, \Phi_{2N}\}$. بر این اساس نورونهای روشن با مجموعه $ON = \{H_i \mid 1 \leq \text{State}(H_i) \leq N\}$ و نورونهای خاموش با مجموعه $OFF = \{H_i \mid N+1 \leq \text{State}(H_i) \leq 2N\}$ نشان داده می‌شوند. نورونهایی که در وضعیتهای مربوط به این اقدام واقع شوند برای آموزش مورد استفاده قرار نمی‌گیرند. $\beta = \{0, 1\}$ ورودیهای اتوماتای یادگیر می‌باشد. در این مجموعه ۱ جریمه و ۰ پاداش را نشان می‌دهد. نحوه عملکرد الگوریتم به این صورت است که در ابتدا تمامی نورونها روشن بوده هستند و در آموزش شرکت می‌کنند. نورونهایی که دارای عملکرد متناسب نیستند جریمه شده و نورونهای با عملکرد متناسب پاداش داده می‌شوند. برای ارزیابی عملکرد یک نورون، از متوسط انرژی نورونها استفاده می‌کنیم (شرح در بخشهای بعد). برای نحوه تعبیر چگونگی عملکرد یک نورون، دو قانون موجود می‌باشد: اگر برای تمامی الگوهای ورودی، مقدار فعالیت نورون تغییرات زیادی داشته باشد در این صورت نورون دارای عملکرد خوبی است و اگر برای تمامی الگوهای ورودی مقدار فعالیت دارای تغییرات کمی باشد نورون دارای عملکرد خوبی نیست.

^۹ Learning Automata

^۹ Finite State Machine

^{۱۰} Krinsky

^{۱۱} Krylov

^{۱۲} Back-propagation Algorithm

^{۱۳} Desired

^{۱۴} Actual

۳-۱- تشخیص نحوه عملکرد نورون روشن

اگر فعالیت نورونی برای تمامی الگوها از یک مقدار آستانه کمتر باشد، نورون بد و اگر از یک مقدار آستانه بیشتر باشد، نورون خوب نامیده می‌شود. برای تعیین مقادیر آستانه، ابتدا واریانس مقدار فعالیت نورون برای تمامی الگوهای آموزش بصورت زیر محاسبه می‌شود:

$$\delta_l = \sqrt{\frac{\sum_{k=1}^P (|U_{lk}| - \mu_l)^2}{P}} \quad l \in ON$$

که در آن، U_{lk} ، فعالیت نورون شماره l برای الگوی شماره K و P تعداد الگوهای آموزش می‌باشد. μ_l مقدار متوسط فعالیت نورون شماره l بوده که به صورت زیر تعریف می‌شود:

$$\mu_l = \frac{\sum_{k=1}^P |U_{lk}|}{P} \quad l \in ON$$

پس از محاسبه واریانس نورونهای روشن، نورونهای روشنی که واریانس فعالیت‌های آنها کمتر از یک مقدار آستانه باشد جریمه شده و نورونهایی که مقدار فعالیت آنها بزرگتر از یک مقدار آستانه دیگر باشد پاداش می‌بینند. نورونهای روشنی که واریانس فعالیت آنها بین دو مقدار آستانه قرار می‌گیرد جریمه یا پاداش داده نمی‌شوند (شکل ۱).



کل ۱: نحوه پاداش و جریمه نورونهای روشن

مقدار M_{ON} که مقدار متوسط واریانسهای نورونهای روشن می‌باشد بصورت زیر محاسبه می‌شود:

$$M_{ON} = \frac{\sum_{k \in ON} \delta_k}{|ON|}$$

پهنای X_{ON} بصورت زیر محاسبه می‌شود:

$$X_{ON} = \lambda_{ON} \frac{|ON| + |OFF|}{ON} \times \frac{\text{Max}(\delta_{ON})}{\text{Min}(\delta_{ON})}$$

در معادله بالا ثابت λ_{ON} ضریب پهنای روشنی نامیده می‌شود. مقدار آستانه پایین $M_{ON} - X_{ON}$ و مقدار آستانه بالا $M_{ON} + X_{ON}$ می‌باشد.

۳-۲- نحوه تمایز بین نورونهای خاموش

نورونهای خاموش در آموزش شبکه شرکت نمی‌کنند. در الگوریتم بقاء نورون، مدت زمان خاموش بودن هر نورون برحسب تعداد epoch آموزشی، به عنوان پارامتری نگهداری می‌شود (n). فعالیت یک نورون خاموش برای یک الگو بر اساس آخرین مقدار

فعالیت این نورون در زمان روشن بودن برای آن الگو محاسبه می‌شود. وقتی یک نورون برای مدت زیادی خاموش باشد، ارزش فعالیت نورون کاهش یافته و بتدریج باعث کم رنگ شدن نقش نورون خاموش می‌شود. به صورت روشن تر، فعالیت یک نورون خاموش را در زمانی خاص بصورت زیر محاسبه می‌کنیم:

$$U_{lk}(n+1) = U_{lk}(n) e^{-\lambda_d |U_{lk}(n)|}$$

در معادله بالا ثابت λ_d ، ضریب کاهش فعالیت نامیده شده و n زمان را نشان می‌دهد. بنابراین مقدار فعالیت یک نورون خاموش به تدریج کاهش می‌یابد.

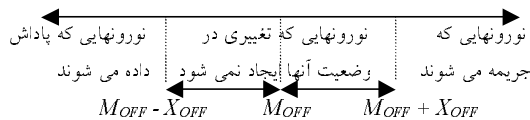
واریانس نورونهای خاموش بصورت زیر محاسبه می‌شود:

$$\delta_l = \sqrt{\frac{\sum_{k=1}^P (|U_{lk}| - \mu_l)^2}{P}} \quad l \in OFF$$

که U_{lk} مقدار فعالیت نورون شماره l برای الگوی شماره K بوده و μ_l مقدار متوسط فعالیت نورون خاموش است که به صورت زیر بیان می‌شود:

$$\mu_l = \frac{\sum_{k=1}^P |U_{lk}|}{P} \quad l \in OFF$$

پس از محاسبه واریانس نورونهای خاموش، نورونهایی که واریانس فعالیت آنها از یک مقدار آستانه کمتر است پاداش دیده و نورونهایی که واریانس فعالیت آنها بین این دو مقدار آستانه می‌باشد نه جریمه و نه پاداش داده می‌شوند (کل ۲).



شکل ۲: نحوه پاداش و جریمه نورونهای خاموش

مقدار M_{OFF} که مقدار متوسط واریانس نورونهای خاموش می‌باشد بصورت زیر محاسبه می‌شود:

$$M_{OFF} = \frac{\sum_{k \in OFF} \delta_k}{|OFF|}$$

پهنای X_{OFF} به صورت زیر محاسبه می‌شود:

$$X_{OFF} = \lambda_{OFF} \frac{|OFF| + |ON|}{OFF} \times \frac{\text{Max}(\delta_{OFF})}{\text{Min}(\delta_{OFF})}$$

در معادله بالا ثابت λ_{OFF} ضریب پهنای خاموشی نامیده می‌شود. مقدار آستانه پایین $M_{OFF} - X_{OFF}$ و مقدار آستانه بالا $M_{OFF} + X_{OFF}$ می‌باشد.

در [21] الگوریتم بقاء نورون اصلاح شده ارائه گردیده است. اصلاحات اعمال شده به الگوریتم بقا بشرح زیر است :

(۱) به جای اتوماتای یادگیر مهاجرت اشیاء که توسط الگوریتم بقا جهت تعیین تعداد نورونهای لایه مخفی در یک شبکه سه لایه استفاده می‌شد، در الگوریتم اصلاح شده از اتوماتای یادگیر L_{RP} (با پارامترهای $a=b=0.1$) که یک اتوماتای یادگیر با ساختار متغیر می‌باشد برای این منظور استفاده میشود.

(۲) در الگوریتم بقاء، فعالیت یک نورون خاموش (خروجی نورون) در زمان $n+1$ بر حسب خروجی نورون در زمان n طبق رابطه زیر محاسبه می‌گردد:

$$U_{lk}(n+1) = U_{lk}(n) e^{-\lambda_d |U_{lk}(n)|}$$

در الگوریتم بقاء نورون اصلاح شده، بعد از هر epoch وزنهای ورودی به نورونهای خاموش و بایاس آنها را با ضریبی (λ) کاهش داده میشود که باعث تغییر در مقدار فعالیت نورون می‌گردد. اگر برای مدتی یک نورون خاموش بماند کلیه وزنهای نورون و بایاس آن به صفر نزدیک شده که با توجه به تابع فعالیت نورون ($f(x) = 1/(1+e^{-x})$)، باعث نزدیک شدن مقدار فعالیت نورون خاموش به مقدار 0.5 میشود. در این هنگام این چنین نورونی حذف گردیده و سپس به بایاس نورونهای لایه بعد مقدار $0.5 \times w_{ij}$ اضافه میشود. وزن مابین نورون خاموش (i) و نورون لایه بعد (j) می‌باشد.

این تغییر موجب کاهش محاسبات شده چرا که الگوریتم انتشار خطا و شبکه به حداقل تغییر نیازمند میباشد. بعد از هر epoch وزنها و بایاس واحدهای خاموش با ضریبی کاهش داده میشود و نورونهای خاموش در تصحیح وزنها و انتشار خطا شرکت داده نمیشود.

۳-۵- الگوریتم بقاء نورون تعمیم یافته

در این بخش روشی جهت تعمیم الگوریتم بقاء اصلاح شده برای شبکه‌هایی با بیش از چند لایه مخفی ارائه می‌گردد. تغییرات انجام گرفته بر روی الگوریتم بقا به قرار زیر میباشد.

(۱) الگوریتم بقاء و الگوریتم بقاء اصلاح شده هر دو برای شبکه‌های سه لایه ارائه شده‌اند. در این الگوریتمها پارامترهای X_{ON} , M_{ON} , X_{OFF} و M_{OFF} برای تنها لایه میانی محاسبه شده و روشن یا خاموش شدن هر نورون میانی بر اساس مقایسه واریانس خروجی نورون (δ) با آستانه‌های $M_{ON} \pm X_{ON}$ و $M_{OFF} \pm X_{OFF}$ تعیین می‌گردد (طبق روابط بخشهای ۱-۳ و ۲-۳). در الگوریتم بقاء تعمیم یافته هر لایه مخفی شبکه، شامل پارامترهای X_{ON} , M_{ON} , X_{OFF} و M_{OFF} مربوط به خود بوده و برای هر لایه پارامترهای λ_{ON} , λ_{OFF} به طور مجزا تعیین می‌شوند. پارامتر δ برای هر نورون مخفی شبکه (واریانس خروجیهای نورون به ازای الگوهای ورودی) محاسبه شده و روشن یا خاموش بودن

آن با توجه به پارامترهای لایه مخفی مربوط به آن نورون و طبق روابط بخشهای ۱-۳ و ۲-۳ محاسبه می‌شوند. به این ترتیب الگوریتم برای شبکه‌های با ساختار لایه‌ای دلخواه قابل استفاده می‌باشد.

(۲) در شبکه با ساختار لایه‌ای دلخواه، چند روش برای فعال نمودن اتوماتها در لایه‌های میانی می‌توان در نظر گرفت که به شرح زیرند: الف) بعد از هر epoch، اتوماتهای همه لایه‌ها همزمان فعال شوند. ب) بعد از هر epoch، اتوماتای مربوط به یک لایه فعال شود (به ترتیب).

ج) بعد از هر epoch، اتوماتهای لایه‌ها یک در میان (چند در میان) فعال شوند.

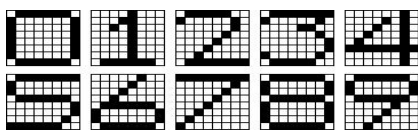
همانطور که مشاهده می‌شود برای فعال سازی اتوماتها در لایه‌های میانی استراتژیهای مختلفی وجود دارد که بسته به شرایط ممکن است یکی از آنها مناسب باشد. جهت کسب اطلاعات بیشتر به [۲۲] مراجعه نمایید.

۴- نتایج شبیه سازیها

شبیه سازیها برای شبکه‌های با دو و سه لایه میانی انجام شده و برای فعال سازی اتوماتها در لایه‌های میانی شبکه از روش (الف) در بخش ۳-۵ استفاده شده است. دو اتوماتای کرایلو و مهاجرت اشیاء در الگوریتم بقاء تعمیم یافته مورد آزمایش قرار گرفته‌اند. اتوماتای مهاجرت اشیاء که در الگوریتم بقاء سنتی استفاده می‌شد در شبکه‌های عصبی با دو و سه لایه میانی، نتایج پایین تری نسبت به اتوماتای کرایلو از خود نشان داده است.

دو الگوریتم مبتنی بر اتوماتای یادگیر و دو الگوریتم هرس تکراری و S&D در شبکه‌های با دو و سه لایه میانی و مسائل ارقام انگلیسی، ارقام دست نویس فارسی و حروف چینی آزمایش شده‌اند. هر یک از این مسائل در ادامه شرح داده شده‌اند. در این مسائل، مولفه ۰ می‌بایست با مقدار ۱- جایگزین شود.

ارقام انگلیسی: ارقام انگلیسی به شکل ماتریسهای 8×8 در شکل ۳ نمایش داده شده‌اند. خانه‌های سیاه به معنای مولفه ۱ در ورودی و خانه‌های سفید به معنای مولفه ۰ هستند. این ارقام در قالب ورودیهای ۶۴ مولفه‌ای به شبکه ارائه می‌شوند. تابع تبدیل ورودی شبکه به خروجی می‌تواند به صورت یکی از توابع f_1 یا f_2 که در جدول ۱ ارائه شده باشد.



شکل ۳: نحوه نمایش ارقام انگلیسی

ارقام دست نویس فارسی: برای این مسئله، یک پایگاه داده شامل ۶۰۰ تصویر از ارقام دست نویس فارسی، مورد استفاده قرار گرفته است. این تصاویر پیش پردازش شده و به تصاویر سیاه و سفید با ابعاد

Krylov	۷,۲	۸۱۳,۶	۱۰۰	۰۰۱۲۷
OMA	۹,۴	۸۳,۲	۱۰۰	۰۰۱۸۱
Iterative	۱۳,۲	۲۶	۹۰	۰۰۲۱۲
S&D	۱۶	۲۶,۶	۱۰۰	۰۰۱۹۸

جدول ۲: نتایج الگوریتمهای مختلف را برای مسئله ارقام دست نویس فارسی در شبکه با دو لایه میانی نشان می دهد. هر الگوریتم ۱۰۰ بار اجرا شده و میانگین ۱۰ مورد از بهترین اجراها برای هر الگوریتم در جداول آمده است.

جدول ۲: مسئله ارقام دست نویس فارسی در شبکه با دو لایه میانی

نام روش	تعداد نورونهای میانی	تعداد epoch آموزشی	نرخ تشخیص	خطای حداقل مربعات
Krylov	۱۵,۴	۲۱۶,۳	۹۹,۴۴	۰۰۰۷۷۱
OMA	۱۷,۵	۳۰۵,۹	۹۹,۸۴	۰۰۱۲۱
S&D	۲۸,۴	۶۲,۶	۹۵,۹۲	۰۰۲۱۱۶
Iterative	۲۹,۲	۶۵,۱	۹۳,۴	۰۰۲۰۷۳

۴-۲- مقایسه الگوریتمها در شبکههای با سه لایه میانی

برای مسائل بررسی شده در این بخش، مقادیر پارامترهای الگوریتمهای مبتنی بر اتوماتای یادگیر، برای لایه اول میانی $\lambda_{OFF} = 0.02$, $\lambda_{ON} = 0.01$ و برای لایههای دوم و سوم میانی $\lambda_{OFF} = 0.02$, $\lambda_{ON} = 0.002$ در نظر گرفته شده است. مقدار λ برای هر سه لایه ۰,۰۵ می باشد. جداول ۳ نتایج الگوریتمهای مختلف را برای سه مسئله ارقام انگلیسی و مسئله حروف چینی در شبکه با سه لایه میانی نشان می دهند. هر الگوریتم ۵۰ بار اجرا شده و میانگین ۱۰ مورد از بهترین اجراها برای هر روش در جداول آمده است.

جدول ۳: مسئله ارقام انگلیسی در شبکه با سه لایه میانی

نام روش	تعداد نورونهای میانی	تعداد epoch آموزشی	نرخ تشخیص	خطای حداقل مربعات
Krylov	۸,۱	۱۹۶۳,۶	۱۰۰	۰۰۱۱۲
OMA	۱۱,۴	۱۱۸۶,۷	۱۰۰	۰۰۱۴۷
Iterative	۱۵,۳	۱۹۱,۸	۹۱	۰۰۱۹۶
S&D	۱۷,۷	۲۱۹,۹	۷۴	۰۰۲۱

جدول ۴: نتایج الگوریتمهای مختلف را برای مسئله ارقام دست نویس فارسی در شبکه با سه لایه میانی نشان می دهد. هر الگوریتم ۱۰۰ بار اجرا شده و میانگین ۱۰ مورد از بهترین اجراها برای هر الگوریتم در جداول آمده است.

جدول ۴: مسئله ارقام دست نویس فارسی در شبکه با سه لایه میانی

نام روش	تعداد نورونهای میانی	تعداد epoch آموزشی	نرخ تشخیص	خطای حداقل مربعات
---------	----------------------	--------------------	-----------	-------------------

۲۰ × ۲۰ برای استفاده توسط شبکه تبدیل گردیده است. چند نمونه از تصاویر ۲۰ × ۲۰ شده این ارقام در شکل ۴ آمده است. در ابتدا تعدادی از تصاویر به عنوان نمونه های فاز آموزش در نظر گرفته می شوند و تصاویر باقیمانده برای فاز آزمایش استفاده می شوند. تابع تبدیل برای این مسئله مشابه تابع $f_2(x)$ در جدول ۱ است.



شکل ۴: چند نمونه از اعداد فارسی ۲۰ × ۲۰ شده.

جداول ۱ تا ۴ نتایج شبیه سازیها را نشان می دهند. نتایج موجود در این جداول مرتب بوده و مطلوبترین نتایج در سطرهای ابتدایی ذکر شده و به سمت پایین از اهمیت نتایج کاسته می شود. نقاطی که در جداول هاشور خورده، نرخ تشخیص پایین شبکه را به ازای استفاده از یک الگوریتم نشان می دهند.

در مساله ارقام انگلیسی آموزش شبکهها با ۲۰ نورون میانی شروع می شود. در شبکه با دو لایه میانی، ۱۰ نورون به هر یک از دو لایه میانی اختصاص می یابد. در شبکه با سه لایه میانی، ۱۰ نورون به لایه میانی اول و ۵ نورون برای هر یک از لایه های دوم و سوم میانی در نظر گرفته می شود. در مسئله ارقام دست نویس فارسی و شبکه با دو لایه میانی، ۲۰ نورون به هر یک از دو لایه میانی اختصاص می یابد. برای شبکه با سه لایه میانی، ۲۰ نورون در لایه میانی اول و ۱۵ نورون برای هر یک از لایه های دوم و سوم میانی در نظر گرفته می شود. معیار انتخاب بهترین اجرا در تمام شبیه سازیها، کوچک بودن ساختار شبکه (تعداد نورونهای لایه میانی) می باشد و در صورت یکسان بودن ساختار چند شبکه، بیشترین نرخ تشخیص ملاک انتخاب قرار می گیرد.

۴-۱- مقایسه الگوریتمها در شبکههای با دو لایه میانی

برای مسائل بررسی شده در این بخش، مقادیر پارامترهای الگوریتمهای مبتنی بر اتوماتای یادگیر، برای لایه اول میانی $\lambda_{OFF} = 0.02$, $\lambda_{ON} = 0.01$ و برای لایه دوم میانی $\lambda_{OFF} = 0.02$, $\lambda_{ON} = 0.002$ در نظر گرفته شده است. مقدار λ برای هر دو لایه ۰,۰۵ می باشد.

جدول ۱، نتایج الگوریتمهای مختلف را برای مسئله ارقام انگلیسی در شبکه با دو لایه میانی نشان می دهد. هر الگوریتم ۲۰۰ بار اجرا شده و میانگین ۱۰ مورد از بهترین اجراها برای هر الگوریتم در جداول آمده است.

جدول ۱: مسئله ارقام انگلیسی در شبکه با دو لایه میانی

نام روش	تعداد نورونهای میانی	تعداد epoch آموزشی	نرخ تشخیص	خطای حداقل مربعات
---------	----------------------	--------------------	-----------	-------------------

الگوریتم یک نسخه تعمیم یافته از الگوریتم بقاء می باشد که از اتوماتای یادگیر و الگوریتم یادگیری انتشار خطا به عقب برای تعیین ساختار شبکه استفاده می کند. الگوریتم بقاء تعمیم یافته برای شبکه های لایه ای، ساختاری نزدیک به بهینه یعنی شبکه هایی با ساختار کوچک ایجاد می کند.

۰.۰۲۳۵	۱.۰۰	۱۳.۰۴	۲۶.۴	Krylov
۰.۰۰۸۷	۹۹.۴۸	۲۵۳.۲	۳۱	OMA
۰.۰۳۰۶	۹۳.۵۲	۱۵۱.۸	۴۴.۹	S&D
۰.۰۳۰۹	۹۳.۳۲	۱۴۷.۶	۴۶.۶	Iterative

برای مشاهده آزمایشهای بیشتر میتوان به [22] مراجعه کرد

۵- نتیجه گیری

در این مقاله، الگوریتمی بر اساس اتوماتای یادگیر برای تعیین ساختار شبکه های عصبی با ساختار لایه ای دلخواه ارائه گردیده است. این

مراجع

Technical Reports, Computer Eng. Dept. Amirkabir University of Technology, Tehran, Iran (In Persian).

[15] Narendra, K. S. and Thatachar, M. A. L. (1974). "Learning Automata: A Survey," IEEE Trans. on Systems, Man, and Cybernetic, Vol. SMC-4, PP. 323-334.

[16] Meybodi, M. R. and Beigy, H., "A Note on Learning Automata Based Schemes for Adaptation of BP Parameters", Journal of Neurocomputing, Vol. 48, No. 4, pp. 957-974, October 2002.

[17] Beigy, H. and Meybodi, M. R. "Backpropagation Algorithm Adaptation Parameters Using Learning Automata", International Journal of Neural System, Vol. 11, No. 11, No. 3, PP. 219-228, 2001.

[18] Meybodi, M. R. and Beigy, H., "New Learning Automata Based Algorithms for Adaptation of Backpropagation Algorithm Parameters", International Journal of Neural System, Vol. 12, No. 1, PP. 45-67, 2002.

[19] Adibi, P., Meybodi, M. R. and R. Safabakhsh, "Unsupervised Learning of Synaptic Delays based on Learning Automata in an RBF-Like Network of Spiking Neurons for Data Clustering", Journal of Neurocomputing, Elsevier Publishing Company, Accepted for publication.

[20] Mashoufi, B., Mehaj, M. B., Motamedi, A., and Meybodi, M. R., "Introducing an Adaptive VLR Algorithm Using Learning Automata for Multilayer Perceptron", IEICE Transactions on Information and Systems, Vol. E86-D, No. 3, pp. 495-609, March 2003.

[21] Anjidani, M. and Meybodi, M. R., "An Algorithm for designing Small Neural Networks with High Generalization Using Learning Automata", Proceedings of 13th Iranian Electrical Engineering Conference, University of Zanjan, Zanjan, Iran, pp. 326-332, May 10-12 2005

[22] Anjidani, M. and Meybodi, M. R., "Neural Network Engineering Using Learning Automata: Determination of the Number of Hidden Units for Multi-Layer Neural Networks and Adaptation of Vigilance Parameter of ART Neural Network", MS. Thesis, Amirkabir, Tehran, Iran, 2005.

[23] Narendra, K. S., and Thathachar, M. A. L., Learning Automata: An Introduction, Printice-Hall, 1989.

[1] Lin, J. H. and Vitter, J. S. (1991). "Complexity Results on Learning by Neural Nets." Machine Learning, Vol. 6, PP. 211-230.

[2] Castellano, G., Fanelli, A. M. and Pelillo, M. (1997). "An Iterative Pruning Algorithm for Feed forward Neural Networks", IEEE Transactions on Neural Networks, Vol. 8, No. 3, PP. 519-531.

[3] Kruschke, J. H. (1989). "Improving generalization in backpropagation networks." Proc. of Int. Joint Conf. on Neural Networks, Vol. 1, PP. 443-447.

[4] Reed, R. (1993). "Pruning Algorithms---A survey" IEEE Trans. on Neural Networks, Vol. 4, No. 5, PP. 740-747.

[5] Sietsma, J. and Dow, R. J. F. (1991). "Creating Artificial Neural Networks That Generalize", Neural Networks, Vol. 4, PP. 67-79.

[6] Beigy, H. and Meybodi, M. R. (1998). "A fast method for determining the number of hidden units in feedforward neural networks." Proc. of CSIC-97, Tehran, Iran, PP. 414-420 (In Persian).

[7] Kwok, T. Y. and Yeng, D. Y. (1997). "Constructive algorithms for structure learning in feedforward neural networks for regression problems." IEEE Trans. on Neural Networks, Vol. 8, No. 3, PP. 630-645.

[8] Hirose, Y., Yamashita, K. and Hijya, S. (1991). "Back-propagation algorithm which varies the number of hidden units." Neural Networks, Vol. 4, No. 1, PP. 61-66.

[9] Nabhan, T. M. and Zomaya, A. Y. (1994). "Toward neural networks structures for function approximation." Neural Networks, Vol. 7, No. 1, PP. 89-99.

[10] Angeline, P. J., Saunders, G. M. and Pollack, J. B. (1994). "Evolutionary algorithm that construct recurrent neural networks." IEEE Trans. on Neural Networks, Vol. 5, No. 1, PP. 54-65.

[11] Yao, X. and Liu, Y. (1997). "A new evolutionary system artificial neural networks." IEEE Trans. on Neural Networks, Vol. 8, No. 3, PP. 694-713.

[12] Beigy, H. and Meybodi, M. R. (1999). "Optimization of topology of neural networks using learning automata." Proc. of 4th Annual Int. Computer Society of Iran Computer Conf. CICC-98, Tehran, Iran, PP. 417-428 (In Persian).

[13] Beigy, H. and Meybodi, M. R. (1999). "A learning automata based algorithm for determination of optimal number of hidden units in three layers feedforward neural networks." Journal of Amirkabir, Tehran, Iran (In Persian).

[14] Meybodi, M. R. and Beigy, H. (1999). "Neural Network engineering using learning automata: determination of desired size for three layer feedforward neural network."