سیزدهمین کنفرانس مهندسی برق ایران
۲۲-۲۰ اردیبهشت ۱۳۸۴

یک الگوریتم برای ساختن شبکه‌های عصبی کوچک با قدرت تعمیم بالا

مجید انجیدنی محمد رضا میبدی

آزمایشگاه محاسبات نرم
دانشکده مهندسی کامپیوتر و فناوری اطلاعات
دانشگاه صنعتی امیرکبیر
تهران ایران
meybodi@ce.aut.ac.ir

چکیده: یافتن ساختار مناسب (نزدیک بهینه) از جمله مسائل مهم در شبکه‌های عصبی می‌باشد که الگوریتمهای متعددی برای آن ارائه شده است. نمونه‌هایی از این الگوریتمها، الگوریتمهای هرس، سازنده، ترکیبی، تکاملی و الگوریتمهای مبتنی بر اتوماتاهای یادگیر هستند که با هدف ایجاد شبکه‌های کوچک ارائه شده اند. یکی از الگوریتمهای مبتنی بر اتوماتاهای یادگیر، الگوریتم بقاء[1] می‌باشد. در این مقاله یک نسخه اصلاح شده از الگوریتم بقاء نورون[2] ارائه می‌گردد. تغییرات انجام گرفته در الگوریتم بقاء نورون با هدف ایجاد شبکه‌های کوچکتر با قدرت تعمیم بالا می‌باشد. الگوریتم پیشنهادی با الگوریتم بقاء و دو مورد از الگوریتمهای هرس با نامهای S&D[3] و بازگشتی[4] برای مسایل مختلف مقایسه می‌گردد. گردید. نتایج آزمایشها برتری الگوریتم پیشنهادی را در مقایسه با سه الگوریتم فوقالذکر نشان می‌دهد.

واژه‌های کلیدی: شبکه‌های عصبی، انتشار خطا به عقب، الگوریتم تنظیم ساختار شبکه، قدرت تعمیم، اتوماتاهای یادگیر

۱) مقدمه

یکی از موضوعات مورد بحث در شبکه‌های عصبی چند لایه یافتن ساختار مناسب(نزدیک به بهینه) برای حل مسئله می‌باشد. در روشهای کلاسیک، طراح شبکه در ابتدای آموزش ساختاری را برای شبکه تعیین و سپس شبکه را آموزش می‌دهد و ساختار تعیین شده در حین آموزش ثابت نگه داشته می‌شود. در این روشها تعیین تعداد نورونهای لایه مخفی و وزنهای آن تاثیر بسزایی برروی کارایی شبکه دارد. یک شبکه با اندازه کوچک، خروجی دقیق تولید نمی‌کند و شبکه‌ای با اندازه بزرگ، بسیار کند و پر هزینه 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[5] [۲][۳][۴][۵]، الگوریتمهای سازنده[6]

[1] Survival
[2] Neuron Survival Algorithm (NSA)
[3] Sietsma and Dow
[4] Iterative

[5] Pruning Algorithms
[6] Constructive Algorithms

[۶][۷]، الگوریتمهای ترکیبی[۷] [۸][۹]، الگوریتمهای تکاملی[۸]: [۱۰][۱۱] و الگوریتم‌های بر اساس اتوماتاهای یادگیر[۹] تقسیم کرد. تنها الگوریتم گزارش شده بر اساس اتوماتاهای یادگیر الگوریتم بقاء نام دارد که توسط بیگی و میبدی ارائه گردیده است[۱۲][۱۳][۱۴]. دو نسخه متفاوت از الگوریتم بقاء موجود است که یکی برای تعیین حداقل تعداد نورونها (الگوریتم بقاء نورون) و دیگری برای تعیین حداقل تعداد وزنها (الگوریتم بقاء وزن) به کار برده می‌شود. الگوریتم بقاء 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 بقاء 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].

در این مقاله یک نسخه اصلاح شده الگوریتم بقاء ارائه می‌گردد. هدف از تغییرات انجام گرفته ایجاد نسخه‌ای از الگوریتم بقاء می‌باشد که بتواند در مقایسه با الگوریتم بقاء شبکه‌های کوچکتری که دارای قدرت تعمیم بالاتری هستند تولید کند و از این طریق کارایی آن افزایش یابد. از جمله این تغییرات، جایگزینی اتوماتای یادگیر مهاجرت اشیاء که از زمره اتوماتاهای یادگیر با ساختار ثابت می‌باشد با یک اتوماتای یادگیر با ساختار متغیر با نام L_{RP} می‌باشد. تغییر دیگر مربوط به چگونگی کاهش فعالیت نورونهای خاموش در الگوریتم بقاء نورون می‌باشد. الگوریتم پیشنهادی با الگوریتم بقاء و دو مورد از الگوریتمهای هرس مقایسه می‌گردد. الگوریتم پیشنهادی بر روی سه مسئله *ارقام دست نویس فارسی*، *اعداد نویزی شده انگلیسی* و *پریتی چهاربیتی نویزی شده* آزمایش گردید. نتایج آزمایشها برتری الگوریتم پیشنهادی را در مقایسه با سه الگوریتم فوقالذکر نشان می‌دهد.

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) اتوماتای یادگیر و الگوریتم انتشار خطا به عقب

اتوماتای یادگیر: اتوماتای یادگیر[۱] یک مدل انتزاعی است که تعداد محدودی اقدام را می‌تواند انجام دهد. هر اقدام انتخاب شده توسط محیطی احتمالی ارزیابی می‌گردد و پاسخی به اتوماتای یادگیر داده می‌شود. اتوماتای یادگیر از این پاسخ استفاده نموده و اقدام خود را برای مرحله بعد انتخاب می‌کند. اتوماتاهای یادگیر به دو گروه اتوماتای یادگیر با ساختار ثابت[۱۰] و اتوماتای یادگیر با ساختار متغیر[۱۱] تقسیم می‌شوند. **اتوماتاهای** $L_{2N,2}$، $G_{2N,2}$ **واتوماتای مهاجرت اشیاء**[۱۲] مثالهایی از اتوماتاهای یادگیر با ساختار ثابت و اتوماتاهای یادکیر L_{RP}[۱۳]، $L_{R\varepsilon P}$[۱۴] و L_{RI}[۱۵] مثالهایی از اتوماتاهای یادگیر با ساختار متغییر مباشند. برای اطلاعات بیشتر درباره اتوماتاهای یادگیر می‌توان به مراجع [۱۲] و[۱۵] مراجعه نمود.

الگوریتم انتشار خطا به عقب[۱۶]: این الگوریتم یک الگوریتم بازگشتی کاهش گرادیان است که برای آموزش شبکه‌های عصبی پس خور 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 ممنتم و G گرادیان تابع خطا می‌باشد که به صورت زیر محاسبه می‌شود:

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

که E_p برابر مجموع مربعات خطای زیر است:

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

---

[۶] Hybrid Algorithms
[۷] Evolutionary Algorithms
[۸] Learning Automata

[۱۰] Fixed Structure
[۱۱] Variable Structure
[۱۲] Object Migrating Automata
[۱۳] Linear Reward Penalty
[۱۴] Linear Reward Epsilon Penalty
[۱۵] Linear Reward Inaction
[۱۶] Back-propagation Algorithm

بطوریکه $T_{p,j}$ و $O_{p,j}$ به ترتیب خروجی *خواسته شده*[17] و *واقعی*[18] 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 این الگوریتم از یک اتوماتا مهاجرت اشیاء برای تعیین تعداد نورونهای لایه مخفی یک شبکه سه لایه استفاده شده است. وظیفه این اتوماتا تقسیم بندی نورونهای لایه مخفی به دو گروه روشن و خاموش می‌باشد. این اتوماتا به صورت شش تایی $\langle \alpha, H, \Phi, \beta, F, G \rangle$ نشان داده می‌شود که در آن $\underline{\alpha} = \{\alpha_1, \alpha_1\}$ اقدامهای اتوماتا یادگیر می‌باشد. اتوماتا دارای دو اقدام است: *اقدام شماره* یک، اقدام مناسب 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. $\underline{H} = \{H_1, H_2, ..., H_n\}$ نورونهای مخفی هستند که روشن و خاموش کردن آنها به عهده اتوماتا می‌باشد. اگر نورون H_i در اقدام شماره یک ظاهر شود به معنای روشن بودن آن و در غیر این صورت خاموش خواهد بود. $\underline{\Phi} = \{\Phi_1, \Phi_2, ..., \Phi_{2N}\}$ حالتهای اتوماتان بوده و N عمق حافظه می‌باشد. حالتهای اتوماتان به دو گروه تقسیم می‌شوند: $\{\Phi_1, \Phi_2, ..., \Phi_N\}$ و $\{\Phi_{N+1}, \Phi_2, ..., \Phi_{2N}\}$. بر این اساس نورونهای روشن با مجموعه $ON = \{H_i \mid 1 \le State(H_i) \le N\}$ و نورونهای خاموش با مجموعه $OFF = \{H_i \mid N+1 \le State(H_i) \le 2N\}$ نشان داده می‌شوند.

نورونهایی که در وضعیتهای مربوط به این اقدام واقع شوند برای آموزش مورد استفاده قرار نمی‌گیرند. $\underline{\beta} = \{0,1\}$ ورودیهای اتوماتا 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 مناسب نیستند جریمه شده و نورونهای با عملکرد مناسب 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 زیر 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.

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:

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

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:

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

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: 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:

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

[17] Desired
[18] Actual

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.

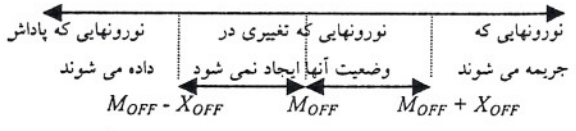
واریانس نورونهای خاموش بصورت زیر محاسبه می‌شود:

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

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:

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

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



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:

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

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.

۳-۴) الگوریتم بقاء نورون اصلاح شده

تغییرات اعمال شده در الگوریتم بقاء نورون به شرح زیر است:

۱) بر خلاف الگوریتم بقاء که از اتوماتای مهاجرت اشیاء 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 با ساختار متغیر می‌باشد، استفاده شده است.

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 کاهش می‌دهیم(λ). این کار فعالیت نورون را تغییر خواهد داد. بعد از مدتی که نورون خاموش است تمام وزنهای نورون و بایاس آن به صفر نزدیک می‌شوند و با توجه به تابع فعالیت نورون ($f(x) = 1/(1+e^{-x})$)، مقدار فعالیت نورون خاموش برابر ۰/۵ خواهد شد. در این هنگام می‌توان نورون را حذف کرده و به بایاس نورونهای لایه بعد مقدار $0.5 \times w_{ij}$ اضافه نمود(w_{ij} وزن مابین نورون خاموش(i) و نورون لایه بعد(j) می‌باشد).

این تغییر موجب کاهش محاسبات شده و از آن مهمتر با اعمال این روش، الگوریتم انتشار خطا و شبکه به حداقل تغییر نیازمند بوده و تنها نیاز است که بعد از هر epoch وزنها و بایاس واحدهای خاموش را با ضریبی کاهش داده و نورونهای خاموش را در تصحیح وزنها و انتشار خطا شرکت ندهیم.

۴) نتایج پیاده سازیها

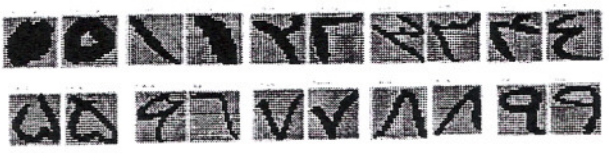
الگوریتم پیشنهادی بر روی دو مسئله اعداد فارسی دست نویس و اعداد نویزی شده انگلیسی آزمایش شده و نتایج حاصل از آن با

نتایج الگوریتم بقاء نورون و دو الگوریتم هرس S&D و بازگشتی مقایسه شده است که نتایج آن در ادامه ارائه می‌گردد. برای هر سه مسئله، مقادیر پارامترهای الگوریتمهای بقاء، $\lambda_{OFF} = 0.015, \lambda_{ON} = 0.006, \lambda = 0.1$ و پارامترهای اتوماتای L_{RP} ، a=b=0.1 در نظر گرفته شده است. برای الگوریتم BP ، پارامتر ممنتم ۰/۹ و نرخ یادگیری ۰/۱ در نظر گرفته شده است.

۴-۱) نتایج پیاده سازی برای اعداد فارسی

برای این مسئله، یک پایگاه داده شامل ۶۰۰ تصویر از اعداد نوشته شده فارسی، مورد استفاده قرار گرفته است[]. این تصاویر پیش پردازش شده و به تصاویر سیاه و سفید با ابعاد ۲۰×۲۰ برای استفاده توسط شبکه تبدیل گردیده است. چند نمونه از تصاویر ۲۰×۲۰ شده این اعداد در شکل ۱ آمده است. در ابتدا تعدادی از تصاویر به عنوان نمونه‌های فاز آموزش در نظر گرفته می‌شوند و تصاویر باقیمانده برای فاز آزمایش استفاده می‌شوند.

ستون اول جداول ۱، ۲، ۳ و ۴، تعداد نمونه‌های در نظر گرفته شده برای فاز آموزش را نشان می‌دهد. ستون دوم تعداد نورون لایه میانی شبکه را قبل از آموزش شامل می‌شوند. تعداد نورون لایه میانی بعد از اتمام آموزش در ستون سوم قرار دارد. نرخ تشخیص برای نمونه‌های آموزش و نمونه‌های آزمایش(قدرت تعمیم) به ترتیب در ستونهای چهارم و پنجم آمده و در ستون ششم خطای شبکه قرار داده شده است. هر الگوریتم ۲۰ بار اجرا شده و میانگین ۵ تا از بهترین اجراها برای هر الگوریتم در جداول ذکر شده است. معیار انتخاب بهترین اجرا، کوچک بودن ساختار شبکه (تعداد نورونهای لایه میانی) می‌باشد و در صورت یکسان بودن ساختار چند شبکه، بیشترین قدرت تعمیم(نرخ تشخیص برای داده‌های آزمایشی) ملاک انتخاب قرار می‌گیرد. با توجه به این جداول بالاترین قدرت تعمیم(بالاترین نرخ تشخیص برای نمونه‌های آزمایشی) و کوچکترین شبکه‌ها توسط الگوریتم پیشنهادی ایجاد شده است. شکل۲، خطای یک اجرای نمونه از شبکه را در حین آموزش نشان می‌دهد.



شکل ۱: چند نمونه از اعداد فارسی ۲۰×۲۰ شده.

جدول ۱) خلاصه نتایج الگوریتم بقاء مبتنی ($L_{RP}(a=b=0.1)$)

تعداد نمونه‌های آموزشی	تعداد نورونهای لایه میانی قبل از آموزش	تعداد نورونهای لایه میانی بعد از آموزش	نرخ تشخیص برای داده‌های آموزشی	نرخ تشخیص برای داده‌های آزمایشی	خطای شبکه
100	25	10.4	99.2	56.72	0.009
150	40	14.8	98.92	62.32	0.005
200	50	10.8	98.7	65.46	0.005
250	50	15	99.84	72.98	0.006
300	50	15.8	99.12	73.86	0.006
350	55	19	99.06	74.24	0.005

جدول ۲) خلاصه نتایج الگوریتم بقاء مبتنی بر OMA

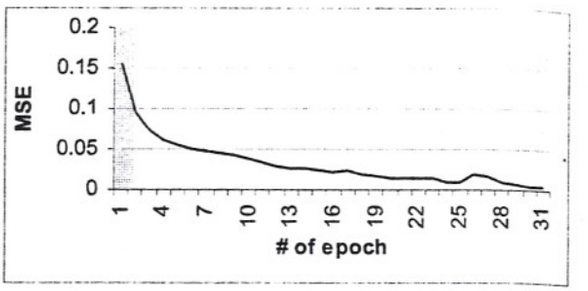
تعداد داده‌های آموزشی	تعداد نورونهای لایه میانی قبل از آموزش	تعداد نورونهای لایه میانی بعد از آموزش	نرخ تشخیص برای داده‌های آموزشی	نرخ تشخیص برای داده‌های آزمایشی	خطای شبکه
100	25	10.8	98.2	57.88	0.01
150	40	15.2	97.58	60.62	0.006
200	50	17.8	97.5	62.68	0.005
250	50	16.8	97.6	70.86	0.006
300	50	15.2	97.48	70.74	0.005
350	55	23	98.06	72.08	0.006

جدول ۳) خلاصه نتایج الگوریتم بازگشتی

تعداد داده‌های آموزشی	تعداد نورونهای لایه میانی قبل از آموزش	تعداد نورونهای لایه میانی بعد از آموزش	نرخ تشخیص برای داده‌های آموزشی	نرخ تشخیص برای داده‌های آزمایشی	خطای شبکه
100	25	16	98.2	55.12	0.008
150	40	21	98	58.38	0.008
200	50	25.6	98.2	60.82	0.006
250	50	30.6	98	67.96	0.007
300	50	28.2	98.34	67.92	0.007
350	55	34.4	98.24	68.16	0.006

جدول ۴) خلاصه نتایج الگوریتم س&د

تعداد داده‌های آموزشی	تعداد نورونهای لایه میانی قبل از آموزش	تعداد نورونهای لایه میانی بعد از آموزش	نرخ تشخیص برای داده‌های آموزشی	نرخ تشخیص برای داده‌های آزمایشی	خطای شبکه
100	25	21.8	98	55.28	0.006
150	40	33.2	98.4	59.24	0.006
200	50	34.8	98.2	62.08	0.006
250	50	43.4	98.72	68.64	0.005
300	50	49.4	99.6	67.32	0.005
350	50	44.6	98.18	69.92	0.005



شکل ۲: نمودار خطای شبکه برای الگوریتم بقاء پیشنهادی.

۴-۲) نتایج برای مسئله اعداد نویزی شده انگلیسی

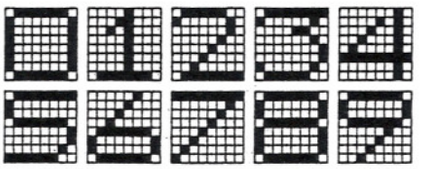
نمونه‌های بدون نویز برای این مسئله در شکل ۳ آمده است. ۴۰۰ نمونه از طریق اضافه کردن نویز سفید گاوسی[۱۹] با قدرت[۲۰] 1 dBW به ۱۰ نمونه بدون نویز تولید می‌شود. ۲۰۰ نمونه از آنها برای فاز آموزش و ۲۰۰ نمونه برای فاز آزمایش مورد استفاده قرار می‌گیرد.

ستون اول جداول ۵، ۶، ۷ و ۸، شماره شبکه آموزش یافته را نشان می‌دهد. ستون دوم تعداد نورون لایه میانی شبکه را بعد از آموزش شامل می‌شوند. نرخ تشخیص برای نمونه‌های آموزش و نمونه‌های آزمایش(قدرت تعمیم) به ترتیب در ستونهای سوم و چهارم آمده و در ستون پنجم خطای شبکه قرار داده شده است. هر الگوریتم ۵۰ بار اجرا شده و ۵ مورد از بهترین اجراها برای هر الگوریتم در جداول آمده است. معیار انتخاب بهترین اجرا، کوچک بودن ساختار شبکه (تعداد نورونهای لایه میانی) می‌باشد و در صورت یکسان بودن ساختار چند شبکه، بیشترین قدرت تعمیم(نرخ تشخیص برای داده‌های آزمایشی) ملاک انتخاب قرار می‌گیرد. آموزش با شبکه‌ای دارای ۲۰ نورون میانی، روی ۲۰۰ نمونه آموزشی انجام شده است. نتایج موجود در جداول، کوچک‌تر بودن شبکه‌های ایجاد شده و بالاتر بودن قدرت تعمیم شبکه آموزش یافته با الگوریتم پیشنهادی را نسبت به الگوریتمهای دیگر نشان می‌دهد.

شکل ۴ نمودار خطای یک اجرای نمونه الگوریتم را نشان می‌دهد. خطای شبکه به دلیل وجود نمونه‌های نویزی تغییرات غیر قابل پیش‌بینی از خود نشان می‌دهد. فاز آموزش با شرط حصول نرخ یادگیری ۱۰۰ درصد پایان یافته است.

[۱۹] White Gaussian Noise

[۲۰] Power



شکل ۳: نحوه نمایش اعداد بدون نویز انگلیسی

جدول ۵) خلاصه نتایج الگوریتم بقاء مبتنی $L_{RP}(a=b=0.1)$

شبکه	تعداد نورونهای لایه میانی بعد از آموزش	تعداد epoch	نرخ تشخیص برای داده‌های آموزشی	نرخ تشخیص برای داده‌های آزمایشی	خطای شبکه
1	6	14	100	98.5	0.6033
2	5	56	100	99.5	0.6125
3	6	29	100	99	0.5856
4	4	45	100	99	0.6367
5	5	19	100	98.5	0.627
Average	**5.2**	**32.6**	**100**	**98.9**	**0.61302**

جدول ۶) خلاصه نتایج الگوریتم بقاء مبتنی بر OMA

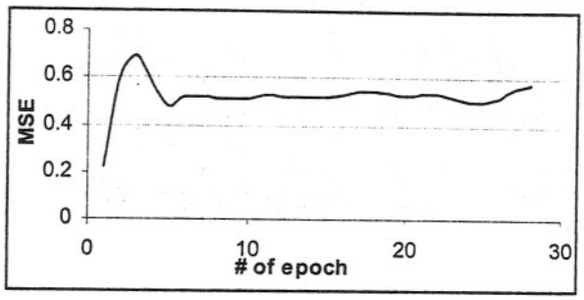
شبکه	تعداد نورونهای لایه میانی بعد از آموزش	تعداد epoch	نرخ تشخیص برای داده‌های آموزشی	نرخ تشخیص برای داده‌های آزمایشی	خطای شبکه
1	9	11	100	98.5	0.6678
2	9	14	100	99.5	0.6211
3	8	16	100	99	0.6498
4	8	14	100	98.5	0.6
5	9	12	100	99	0.6366
Average	**8.6**	**13.4**	**100**	**98.9**	**0.63506**

جدول ۷) خلاصه نتایج الگوریتم بازگشتی

شبکه	تعداد نورونهای لایه میانی بعد از آموزش	تعداد epoch	نرخ تشخیص برای داده‌های آموزشی	نرخ تشخیص برای داده‌های آزمایشی	خطای شبکه
1	8	11	100	99.5	0.6314
2	8	10	100	98.5	0.6377
3	9	10	100	98.5	0.6376
4	7	9	100	98.5	0.6244
5	9	11	100	98.5	0.6167
Average	**8.2**	**10.2**	**100**	**98.7**	**0.62956**

جدول ۸) خلاصه نتایج الگوریتم س&د

شبکه	تعداد نورونهای لایه میانی بعد از آموزش	تعداد epoch	نرخ تشخیص برای داده‌های آموزشی	نرخ تشخیص برای داده‌های آزمایشی	خطای شبکه
1	11	10	99.5	98	0.6242
2	13	9	99.5	99.5	0.621
3	14	9	99.5	98.5	0.6444
4	14	15	97.5	97.5	0.6141
5	13	9	99	98.5	0.6344
Average	**13**	**10.4**	**99**	**98.4**	**0.62762**



شکل ۴: نمودار خطای شبکه برای یک اجرای نمونه از مسئله اعداد نویزی شده انگلیسی

۵) نتیجه گیری

در این مقاله، یک نسخه بهبود یافته الگوریتم بقاء نورون برای تعیین ساختار شبکه‌های عصبی سه لایه ارائه گردید. الگوریتم پیشنهادی از اتوماتای یادگیر L_{RP} و الگوریتم یادگیری انتشار خطا به عقب برای تعیین ساختار شبکه استفاده می‌کند. الگوریتم پیشنهادی شبکه‌های کوچک با قدرت تعمیم بالا ایجاد می‌کند. الگوریتم پیشنهادی با الگوریتم بقاء نورون و دو الگوریتم هرس به نام‌های S&D و بازگشتی مقایسه گردید. الگوریتم پیشنهادی بر روی مسائل اعداد فارسی دست نویس و اعداد نویزی شده انگلیسی آزمایش گردید و نشان داده شد که الگوریتم بقاء اصلاح شده موفق به ایجاد ساختارهای نزدیک به بهینه با قدرت تعمیم بالا شده است.

مراجع

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

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

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

[4] R. Reed, "Pruning Algorithms---A survey" IEEE Trans. on Neural Networks, Vol. 4, No. 5, PP. 740-747,193.

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

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

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

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

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

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

[12] H. Beigy and M. R. Meybodi, "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, 1999.

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

[14] M. R. Meybodi and H. Beigy "Neural Network engineering using learning automata: determination of desired size for three layer feedforward neural network." Technical Reports, Computer Eng. Dept. Amirkabir University of Technology, Tehran, Iran, 1999

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

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

[18]M. R. Meybodi, and H. Beigy, "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]P. Adibi, P., M. R. Meybodi, 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, No. 64, pp.335-357, 2005.

[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. pp. 495-609, March 2003.