فصلنامه علمی-ترویجی پدافند غیرعامل

سال چهارم، شماره ۱، بهار ۱۳۹۲، (پیاپی ۱۳): صص ۵۱-۶۱

تحمل‌پذیری خطا در شبکه‌های عصبی MLP با استفاده از افزونگی مؤلفه‌های سه‌گانه

محمدرضا حسنی‌آهنگر[۱]، مصطفی اخضمی[۲]

تاریخ دریافت: ۹۲/۰۱/۱۸
تاریخ پذیرش: ۹۲/۰۳/۲۰

چکیده

استفاده از سامانه‌ها و زیرساخت‌های پیچیده و بزرگ، برای انجام فعالیت‌های مختلف یک کشور حیاتی و مهم است و رعایت مسائل پدافند غیرعامل برای آن‌ها در شرایط بحران، که بتوانند سرویس‌های خود را به‌طور کامل یا بخشی از آن را ارائه نمایند، یکی از معیارهای ارزیابی این‌گونه سامانه‌ها می‌باشد. مدل‌سازی و شبیه‌سازی این سامانه‌ها، جهت تشخیص گلوگاه‌ها مهم است. رخداد خطا با وجود تمهیدات مختلف مانند پیش‌بینی خطا، جلوگیری از خطا، پوشش خطا و تحمل‌پذیری خطا، طبیعی به نظر می‌رسد. شبکه‌های عصبی مصنوعی به‌عنوان یکی از روش‌های مدل‌سازی و شبیه‌سازی، کاربردهای فراوانی در این خصوص برای بررسی عملکرد سامانه‌های پیچیده و حیاتی دارند. با توجه به این‌که شبکه‌های عصبی مصنوعی بر اساس الگوی شبکه‌های عصبی طبیعی که به‌طور ذاتی قابلیت تحمل‌پذیری خطا را دارا هستند طراحی شده‌اند، لذا باید بتوانند از قابلیت تحمل‌پذیری خطا بهره گیرند. در این مقاله روشی برای افزایش و بهبود تحمل‌پذیری خطا در شبکه‌های عصبی، مبتنی‌بر روش افزونگی مؤلفه‌های سه‌گانه (TMR) ارائه شده است. این روش نشان می‌دهد که بر اساس این تکنیک، تحمل‌پذیری خطا به شکل مطلوبی افزایش یافته است.

کلیدواژه‌ها: شبکه‌های عصبی چندلایه، تحمل‌پذیری خطا، افزونگی، پدافند غیرعامل، TMR

۱- استادیار و عضو هیئت علمی دانشگاه جامع امام حسین(ع) mrhassani@iust.ac.ir

۲- دانشجوی کارشناسی ارشد کامپیوتر دانشگاه جامع امام حسین(ع) mostafa_akhzami@yahoo.com - نویسنده مسئول

۱- مقدمه

در سال‌های اخیر، حرکتی مستمر از تحقیقات صرفاً تئوری به تحقیقات کاربردی به‌خصوص در زمینه پردازش اطلاعات، برای مسائلی که برای آن‌ها راه‌حلی موجود نیست و یا به راحتی قابل حل نیستند، به‌وجود آمده است. با عنایت به این امر، علاقه فزاینده‌ای در توسعه نظری سیستم‌های پویای هوشمند مدل آزاد[1] که مبتنی بر داده‌های تجربی هستند، ایجاد شده است. شبکه‌های عصبی مصنوعی[2] جزء این دسته از سیستم‌های پویا می‌باشند که با پردازش روی داده‌های تجربی، دانش یا قانون نهفته در ورای داده‌ها را به ساختار شبکه منتقل می‌کنند. به‌همین خاطر به این سیستم‌ها «هوشمند» گفته می‌شود، چرا که این سیستم‌ها بر اساس محاسبات روی داده‌های عددی یا مثال‌ها، قوانین کلی را فرا می‌گیرند[۱].

روش‌های مدل‌سازی به دو دسته هوشمند و کلاسیک طبقه‌بندی می‌شوند. در روش‌های کلاسیک از مدل‌سازی ریاضی و زبان‌های صوری استفاده می‌شود و اغلب برای سیستم‌های خطی مورد استفاده قرار می‌گیرند. روش‌های هوشمند برای سیستم‌های غیرقطعی و احتمالی به‌کار می‌روند. شبکه‌های عصبی، یکی از ابزارهای مدل‌سازی هوشمند به حساب می‌آیند[۲].

شبکه‌های عصبی مصنوعی یک الگوی جذاب برای طراحی و تجزیه‌وتحلیل سیستم‌های هوشمند جهت طیف گسترده‌ای از برنامه‌های کاربردی هوش مصنوعی را ارائه می‌کنند. با وجود انجام فعالیت‌ها و بررسی‌های گسترده در این زمینه از سال‌ها قبل، که منجر به ایجاد نتایج نظری و تجربی خوبی شد، هنوز طراحی شبکه‌های عصبی برای کاربردهای خاص، تحت برخی محدودیت‌های طراحی یک فرایند آزمون و خطا است. کارایی و هزینه شبکه‌های عصبی برای مسائل خاص شدیداً وابسته به انتخاب عناصر پردازشی (نرون‌ها)، معماری شبکه و الگوریتم آموزشی است[۴].

شبکه‌های عصبی کاربردهای بسیاری در علوم نظامی و دفاعی دارند. برخی از این کاربردها شامل، کنترل پهپادها، ردیابی انحراف هواپیماها، تشخیص اهداف نظامی، پردازش سیگنال‌های تصویری جهت مقایسه و تفسیر اطلاعات نظامی، هدایت جنگ‌افزارها، جلوگیری از اختلال در سیستم‌های مخابراتی، سیستم‌های راهنمای خودکار موشک و غیره می‌باشند. بنابراین، **تحقیق در خصوص روش‌های تحمل‌پذیری خطا[3] در شبکه‌های عصبی جهت تقویت مسائل مربوط به پدافند غیرعامل در سیستم‌هایی که بر پایه شبکه‌های عصبی طراحی شده‌اند، مهم‌ترین هدف این مقاله به‌شمار می‌آید.**

۱-۱- بیان مسئله

پیچیدگی سیستم‌های کاربردی روزافزون است و افزایش پیچیدگی روی قابلیت اطمینان سیستم، تأثیر بسزایی دارد. بنابراین، تولید سیستم بی‌عیب‌ونقص امکان‌پذیر نیست و باز هم احتمال خطا در سیستم وجود دارد. قابلیت اطمینان، یکی از صفات اتکاپذیری سیستم‌ها محسوب می‌شود. روش‌ها و فنونی که ساخت یک سیستم اتکاپذیر را ممکن می‌سازند، عبارت‌اند از اجتناب از خطا، پیش‌بینی خطا، رفع خطا و تحمل‌پذیری خطا. برای رسیدن به یک سیستم اتکاپذیر، باید تمهیداتی اندیشیده شود تا چنانچه خطایی در سیستم پیش بیاید، سیستم با وجود خطا، به کار عادی خود ادامه دهد. تحمل‌پذیری خطا زمانی مطرح است که علی‌رغم استفاده از روش‌های دیگر، از جمله اجتناب از خطا و رفع خطا، هنوز یک سری خطا در سیستم باقی مانده باشد. استفاده از روش‌های یادگیری و افزونگی در سطح نرون‌های میانی، دو رویکردی هستند که برای تحمل‌پذیری خطا در شبکه‌های عصبی مطرح هستند. عیب این دو رویکرد این است که فقط برخی از خطاها را پوشش می‌دهند و خطاهای مربوط به لایه خروجی را در نظر نمی‌گیرند. در این تحقیق به دنبال آن هستیم تا با ارائه یک روش مبتنی‌بر افزونگی مؤلفه‌های سه‌گانه (TMR)[4] بتوان تحمل‌پذیری خطا را در شبکه‌های عصبی افزایش داد.

۱-۲- ضرورت و اهمیت تحقیق

بروز خطا در شبکه‌های عصبی مورد استفاده در تجهیزات نظامی و دفاعی می‌تواند عواقب جبران‌ناپذیری به‌دنبال داشته باشد. بنابراین با توجه به تعریف پدافند غیرعامل که عبارت است از: مجموعه اقداماتی که باید انجام شود تا در صورت بروز جنگ، خسارات احتمالی به حداقل میزان خود برسد، باید راهکارهای مناسبی جهت کاهش آثار بروز خطا در شبکه‌های عصبی در نظر گرفته شود. برای کاربردهای نظامی با قابلیت اطمینان بالا، شبکه‌های عصبی مصنوعی باید تحمل‌پذیر خطا باشند. شبکه‌های عصبی مصنوعی در ابتدا باید درجه بالایی از تحمل‌پذیری خطا را دارا بوده و عملکرد آن‌ها باید به شکل مطلوبی تعداد خطاها را کاهش دهد.

۱-۳- اهداف تحقیق

در مدارهای دیجیتال، تحمل‌پذیری خطا از طریق افزونگی به‌صورت افزونگی سخت‌افزار یا زمانی قابل دستیابی است. هدف این مقاله، ارائه تکنیکی برای افزایش تحمل‌پذیری خطا در شبکه‌های عصبی پرسپترون چندلایه[5] با استفاده از افزونگی مؤلفه‌های سه‌گانه است.

1- Model Free
2- Artificial Neural Network
3- Fault Tolerance
4- Triple Modular Redundancy
5- Multi Layer Perceptron

۱-۴- سؤالات تحقیق

- چه عواملی در تحمل‌پذیری ذاتی شبکه‌های عصبی مؤثرند؟
- چه خطاهایی ممکن است در شبکه‌های عصبی رخ دهد؟
- سازوکارهای تحمل‌پذیری خطا در شبکه‌های عصبی چیست و معایب و مزایای آن‌ها کدام است؟
- چه راه‌کاری می‌تواند برای افزایش تحمل‌پذیری خطا در شبکه‌های عصبی ارائه شود که انواع خطاها را پوشش دهد؟

۱-۵- روش تحقیق و ابزار جمع‌آوری

از آن‌جایی که این تحقیق اقدام به حل مسئله می‌نماید، علمی است و چون نتایج آن به‌صورت کاربردی قابل استفاده در شبکه‌های عصبی است، از نوع کاربردی است. در نتیجه، این تحقیق از نوع علمی کاربردی است.

روش جمع‌آوری اطلاعات نیز از منابع کتابخانه‌ای و اینترنتی و اسناد و مدارک تخصصی موجود در این زمینه صورت گرفته است.

۱-۶- متغیرهای تحقیق

در این تحقیق، قابلیت اطمینان به‌عنوان متغیر وابسته به زمان در نظر گرفته شده است.

۱-۷- روش تجزیه و تحلیل

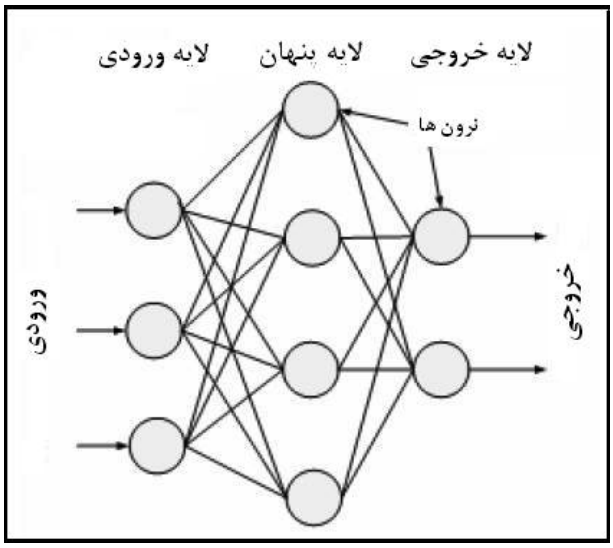
برای مدل‌سازی و تجزیه و تحلیل روش پیشنهادی در این تحقیق، از مدل مارکوف و نرم‌افزار شارپ استفاده شده است.

۱-۸- ساختار تحقیق

در این مقاله در بخش ۲، ابتدا شبکه‌های عصبی چندلایه و تحمل‌پذیری خطا در آن‌ها مطرح شده است. سپس در بخش ۳، سازوکارهای تحمل‌پذیری خطا در شبکه‌های عصبی از جمله روش‌های یادگیری و افزونگی مورد بررسی قرار گرفته است. در بخش ۴، ارائه روش پیشنهادی با استفاده از افزونگی مؤلفه‌های سه‌گانه (TMR) بیان شده است. در نهایت، این مقاله با نتیجه‌گیری و ذکر مراجع خاتمه می‌یابد.

۲- شبکه‌های عصبی چندلایه و تحمل‌پذیری خطا در آن‌ها

شبکه‌های عصبی چند لایه[1] دارای یک یا چند لایه مخفی بین لایه‌های ورودی و خروجی می‌باشند. یک واحد مخفی ممکن است به یک ورودی و یا خروجی و یا یک واحد مخفی متعلق به لایه‌های دیگر متصل باشد. لایه‌ها و گره‌های پنهان، نقش مهمی در موفقیت شبکه‌های عصبی ایفا می‌کنند. گره‌های مخفی در لایه‌های مخفی به شبکه عصبی اجازه می‌دهند تا خصوصیات داده‌ها را کشف و شناسایی نماید و به این وسیله نگاشت‌های غیرخطی پیچیده را بین متغیرهای ورودی و خروجی برقرار نمایند. شکل (۱) شبکه عصبی چندلایه را که در آن تمام واحدهای هر لایه به لایه بعد متصل است نشان می‌دهد. در هر یک از لایه‌ها تعدادی نرون جای گرفته است. به نرون‌های لایه ورودی تنها بردار ورودی اعمال می‌شود. امّا به نرون‌های دیگر لایه‌ها، خروجی‌های نرون‌های لایه پیشین و یک ورودی که آستانه نرون را توصیف می‌کند، اعمال می‌شود. تعداد نرون‌ها در لایه‌های ورودی و خروجی بستگی به تعداد ورودی‌ها و خروجی‌های مسئله مورد نظر دارد. در حالی‌که انتخاب تعداد نرون‌های لایه پنهان یک مسئله طراحی است. شبکه‌ای که در این مقاله بر روی آن تمرکز خواهد شد، شبکه عصبی پرسپترون چندلایه است.



شکل ۱- شبکه عصبی چندلایه با یک لایه پنهان

انواع خطاهایی که در شبکه‌های عصبی ممکن است رخ دهد، در چهار دسته زیر طبقه‌بندی شده‌اند. در حقیقت، این خطاها منجر به مقادیر نادرست برای خروجی می‌شوند[۳].

- سیگنال ورودی: این خطا ناشی از خطا در نرون قبلی یا خطا در اتصالات و یا اختلال خروجی- که روی سیستم تأثیر گذاشته است- می‌باشد.
- خطاهایی که بر روی وزن‌ها اثر می‌گذارند.
- خطاهایی که روی مجموع وزن‌دار داخل نرون تأثیر دارند.
- خطاهایی که روی تابع فعالیت نرون تأثیر دارند.

تحمل‌پذیری خطا عبارت است از توانایی سیستم برای استمرار ارائه خدمات و سرویس‌ها بر اساس مشخصات تعیین‌شده برای آن، در حالی که خطا در سیستم وجود داشته باشد. تحمل‌پذیری خطا به دلایل زیر مورد نیاز است:

1- Multi Layer Neural Network

- در واقع امکان ایجاد سیستم بدون عیب و نقص وجود ندارد. در این رابطه می‌توان به این موضوع اشاره نمود که اصلی‌ترین مسئله در رشد سیستم‌ها، پیچیدگی آن‌ها است و افزایش پیچیدگی موجب تخریب شدید قابلیت اطمینان سیستم است.
- اگرچه طراحان در تلاش هستند که سخت‌افزارهای بدون نقص و نرم‌افزارهای بدون اشکال[1] داشته باشند ولی تجربه نشان می‌دهد که این آرزو دست‌نیافتنی است[۵].

تحمل‌پذیری خطا در شبکه‌های عصبی یک ویژگی مهم است؛ به‌خصوص زمانی که از شبکه‌های عصبی در کاربردهای مهم نظامی و دفاعی استفاده می‌شود. برای افزایش تحمل‌پذیری خطا در شبکه‌های عصبی دو ایده اساسی وجود دارد:

- بروز قابلیت تحمل‌پذیری خطا به‌صورت ذاتی در آن‌ها.
- استفاده از سازوکارهای اضافی برای تحمل‌پذیری خطا[۶].

در یک شبکه عصبی، هر سلول به‌طور مستقل عمل می‌کند و رفتار کلی شبکه، برآیند رفتارهای محلی سلول‌های متعدد است. این ویژگی باعث می‌شود که خطای محلی از چشم خروجی نهایی دور بماند. به عبارت دیگر، سلول‌ها در یک روند همکاری، خطاهای محلی یکدیگر را تصحیح می‌کنند. این خصوصیت باعث افزایش مقاومت (تحمل‌پذیری در برابر خطا) در سیستم‌ها می‌گردد[۳].

نمونه‌ای از تحمل‌پذیری ذاتی در شبکه‌های عصبی مصنوعی، در آزمایشی که سکوین[2] و کلی[3] انجام دادند، قابل مشاهده است. در این آزمایش‌آن‌ها یک شبکه عصبی رو به جلو را برای دسته‌بندی تعدادی کاراکتر آموزش دادند. بعد از یادگیری وقتی تعدادی از نرون‌ها حذف شدند، شبکه همچنان به کار خود ادامه می‌داد[۳].

عوامل زیر در تحمل‌پذیری ذاتی شبکه‌های عصبی در برابر خطا موثر هستند:

• غیرخطی بودن یک شبکه عصبی

این ویژگی به خاطر غیرخطی بودن، تابع فعالیت نرون‌ها است. فرض کنید خروجی تابع فعالیت نرون، وقتی مجموع وزن‌دار ورودی‌ها بزرگتر از ۱۰ باشد، مقدار ۱ باشد. اگر یک خرابی باعث شود که مجموع وزن‌دار از ۲۰ به ۳۰ افزایش یابد یا از ۳۵ به ۱۵ کاهش یابد، در این‌صورت خروجی نرون بدون تغییر، همان مقدار ۱ باقی می‌ماند و خطا تأثیری در خروجی ندارد.

• روش توزیع‌شده ذخیره کردن اطلاعات

شبکه‌های عصبی، اطلاعات را درون وزن‌ها ذخیره می‌کنند که در کل شبکه توزیع شده است. خطا در بعضی وزن‌ها باعث از دست دادن فقط بخش کوچکی از اطلاعات می‌شود و تأثیر کمی در خروجی شبکه عصبی دارد.

• تعداد نرون‌ها در یک شبکه

اضافه کردن نرون‌های اضافی در لایه میانی شبکه باعث می‌شود اهمیت بعضی از نرون‌ها (یا وزن‌ها) از بعضی دیگر کمتر شود. با برداشتن این نرون‌های کم‌اهمیت فقط بخش کوچکی از اطلاعات از بین می‌رود و تأثیر کمی روی خروجی شبکه دارد.

• تفاوت بین کران خطا در مرحله یادگیری و عملیات

با افزایش کران خطا در مرحله عملیاتی نسبت به مرحله یادگیری، خرابی بعضی از نرون‌ها یا وزن‌ها در خروجی شبکه تأثیر نخواهد داشت[۳].

تحمل‌پذیری ذاتی شبکه‌ها در برابر خطا مشکلاتی دارد که از مهم‌ترین آن‌ها می‌توان به مقاومت در برابر تعداد محدودی از خطاها اشاره کرد. بعضی از خطاها مخصوصاً خطا در لایه خروجی باعث خطا در خروجی شبکه عصبی می‌شود[۶].

۳- روش‌های تحمل‌پذیری خطا در شبکه‌های عصبی

همان‌طور که ذکر شد برای افزایش تحمل‌پذیری خطا در شبکه‌های عصبی دو ایده اساسی وجود دارد که عبارت‌اند از: بروز قابلیت تحمل‌پذیری ذاتی شبکه‌های عصبی مانند استفاده از روش‌های یادگیری و استفاده از سازوکارهای اضافی برای تحمل‌پذیری خطا مانند روش‌های افزونگی. در ادامه به بررسی این دو روش و مرور کارهایی که با استفاده از این دو روش صورت گرفته، پرداخته شده است.

۳-۱- تحمل‌پذیری خطا با استفاده از روش‌های یادگیری

در روش‌های یادگیری با تغییر الگوریتم‌های یادگیری، از توانایی تطبیق‌پذیری[4] شبکه‌های عصبی برای مقابله با خطا استفاده می‌شود. این تغییرات باعث می‌شود در شرایطی که بعضی از اجزاء شبکه عصبی مورد نظر از کار بیفتند، شبکه بدون مشکل، وظیفه خود را به درستی انجام دهد.

۳-۱-۱- تکنیک تزریق خطا

تکنیک تزریق خطا[5] از خاصیت سازگاری ذاتی شبکه‌های عصبی در یادگیری، به‌منظور جبران خطاهای احتمالی استفاده می‌کند. در این روش، شبکه‌های عصبی را در مرحله یادگیری در معرض خطا قرار می‌دهند. خطاهایی که در مرحله یادگیری به شبکه اعمال می‌شود شامل خطاهای زیر است.

- خرابی نرون‌ها: خروجی‌های نرون کمترین مقدار یا بیشترین مقدار یا یک مقدار میانی بین این دو حد می‌گیرند.

1- Bug
2- Sequin
3- Clay

4- Adaptive Capability
5- Fault Injection Techniques

- خرابی وزن‌ها: وزن‌ها یا با یک درصد آشفتگی[1]، یا با کمترین، یا بیشترین و یا یک مقدار میانی خراب می‌شوند.
- خرابی ورودی‌ها: شامل تزریق نویز در داده‌های ورودی است. در اصل، این کار مانند خرابی در لایه ورودی است[۳].

روش تزریق خطا را به دلیل شباهتی که بین این روش و واکسیناسیون ویروس‌ها وجود دارد، روش واکسیناسیون می‌نامند. به این ترتیب شبکه‌های عصبی در مقابل خطاها با تزریق خطا در مرحله آموزش واکسینه می‌شوند و یاد می‌گیرند که هنگام مواجهه با خطا در مرحله عملیاتی چگونه بر این خطا غلبه کنند. در ادامه، چند نمونه از کارهایی که از این تکنیک برای تحمل‌پذیری خطا استفاده کرده‌اند، بررسی شده است.

۳-۱-۱-۱- روش N-FTBP

در [۷] یک الگوریتم یادگیری که خطا را به نرون‌ها تزریق می‌کند، به نام روش N-FTBP پیشنهاد شده است. این روش، آموزش را برای شبکه‌های عصبی با خطایی که در خروجی نرون‌ها در لایه‌های پنهان است، انجام می‌دهد.

در روش N-FTBP، یادگیری برای شبکه‌های عصبی با الگوی خطای معین f^n انجام می‌شود. f^n به عنوان nامین انتخاب الگوی خطا است. m تعداد خطاهای تزریق شده در واحد زمان بر روی نرون‌ها و N تعداد نرون‌ها در لایه پنهان است. علاوه بر این، f_1^n خطای Stuck-at-0 از nامین نرون در لایه پنهان و $\{f_2^n,\cdots f_m^n\}$ خطاهای Stuck-at-0 از m-1 نرون که به‌صورت تصادفی از nامین نرون در لایه پنهان است، انتخاب شده‌اند، که رابطه (۱) آن را نشان می‌دهد:

$$f^n=\{f_1^n,f_2^n,\cdots f_m^n\} \qquad (1\le n\le N) \qquad (۱)$$

در این‌جا f^0 نشان‌دهنده این مورد است که خطایی وجود ندارد. سپس E به صورت رابطه (۲) تعریف می‌شود:

$$E_r=\sum_{f^n}\sum_p\sum_i(t_i^p-o_i^{(p)(f^n)})^2/2 \qquad (۲)$$

در رابطه (۲)، $o_i^{(p)(f^n)}$ خروجی نرون iام در لایه خروجی می‌باشد (موقعی که p امین نمونه ورودی یادگیری وارد شبکه با الگوی خطای f می‌شود). از رابطه (۲)، این‌گونه استنتاج می‌شود که وزن‌ها در موردی که شبکه عصبی یک الگوی خطای F دارد، با رابطه (۳) اصلاح می‌شوند.

$$\Delta w=\sum_{f^n}\Delta w_{f^n} \qquad (۳)$$

در این‌جا Δw_{f^n} یک مقدار اصلاحی برای وزن‌های شبکه عصبی نسبت به الگوی خطای f^n تزریق شده است. همان‌طور که در بالا مشاهده شد، رویکرد N-FTBP یادگیری را با تزریق الگوی خطای f^nهای متوالی انجام می‌دهد، به‌گونه‌ای که الگوی خطا f^{n+1} بعد از این‌که یادگیری f^n تمام شد، بر روی شبکه‌های عصبی تزریق می‌شود. اگر وضعیت آموزش پایانی رضایت‌بخش باشد، آن‌گاه آموزش با موفقیت به اتمام می‌رسد.

۳-۱-۲- الگوریتم به حداقل رساندن وزن‌های موثر در خطای خروجی

در [۸] یک الگوریتم آموزشی برای تحمل‌پذیری خطای شبکه‌های پرسپترون چندلایه مطرح شده است. در این مقاله ابتدا لینک‌های اساسی در شبکه شناسایی می‌شوند. یک لینک اتصال، لینک اساسی نامیده می‌شود؛ اگر که آن اتصال موجب یک خطا در خروجی شود. در این مقاله، برای پیدا کردن تعداد لینک‌های اساسی، یک لینک قطع می‌شود (وزن سیناپسی آن برابر صفر) و همه الگوهای آموزشی به شبکه عصبی اعمال می‌شوند. اگر شبکه نتواند تمامی الگوهای آموزشی را تشخیص دهد آن لینک یک لینک اساسی است.

در این الگوریتم، ارتباط وزن‌های سیناپسی با خطای خروجی در هر چرخه آموزش، از اگوریتم پس‌انتشار تخمین زده می‌شود. این کار با استفاده از بسط تیلور انجام می‌شود. پس از آن مقدار وزنی که بیشترین تأثیر بر روی خطای خروجی در فاز آموزش دارد، کاهش می‌یابد. در این مقاله، کاهش وزنی پیشنهاد می‌شود که بالاترین ارتباط را همانند رابطه (۴) تولید کند.

$$W_{ij}\rightarrow\frac{W_{ij}}{1+\lambda R(W_{ij})} \qquad (۴)$$

در رابطه (۴)، λ یک عدد ثابت واقعی است که فاکتور جریمه نامیده می‌شود. فاکتور $1+\lambda R(W_{ij})$ جریمه وزنی - که بالاترین ارتباط و بیشترین تأثیر را روی خروجی دارد- می‌باشد. λ خیلی کوچک هیچ تأثیری ندارد و λ بزرگ موجب جریمه بیش از حد قوی می‌شود و به این ترتیب، روند آموزش طولانی می‌شود. از آن‌جا که کاهش وزنی که دارای بیشترین تأثیر در خروجی است، ممکن است همگرایی فرایند آموزش را مختل کند. فاکتور جریمه از مقدار صفر تا یک زمان ثابت T_λ مقداردهی می‌شود.

الگوریتم نهایی به این صورت است که در مرحله اول، ابتدا مقداردهی اولیه ماتریس وزن، سپس تعیین پارامترهای آموزشی و تعیین فاکتور جریمه λ و T_λ صورت می‌گیرد. مرحله دوم که برای هر چرخه آموزشی انجام می‌شود، شامل انجام استاندارد پس‌انتشار

1- Perturbation

برای به‌روزرسانی وزن‌ها، محاسبه ارتباط همه وزن‌ها و پیدا کردن وزنی که حداکثر ارتباط را تولید می‌کند، می‌باشد. در نهایت در مرحله ۳، در صورت همگرا شدن الگوریتم، کار پایان یافته است. امّا اگر الگوریتم همگرا نشود و زمان T_λ خاتمه یابد، مقدار λ صفر شده و به مرحله ۲ برمی‌گردیم. در غیر این‌صورت، وزن w_{i0j0} را مطابق رابطه (۴) کاهش داده و به مرحله ۲ برمی‌گردیم.

از آن‌جا که وزنی که دارای بیشترین ارتباط (تأثیر) است، کاهش می‌یابد، الگوریتم پیشنهادی معمولاً کندتر از الگوریتم پس‌انتشار است. برای غلبه بر این ضعف، یک زمان حداکثر T_λ بعد از این‌که فاکتور جریمه با مقدار صفر مقداردهی شد، در نظر گرفته می‌شود. به این ترتیب، الگوریتم، معادل الگوریتم پس‌انتشار می‌شود.

۳-۱-۳- روش PAWMA برای کاهش وزن

در [۹] مطابق نمونه کاری که در [۸] ذکر شد، انجام شده است. با این تفاوت که برای رسیدن به تحمل‌پذیری خطا، تابع ارزیابی E‘(w) برای ارزیابی نه تنها خطای ورودی بلکه هم‌چنین مجموع مربعات همه وزن‌ها تعریف شده است. الگوریتم پس‌انتشار با تابع ارزیابی E‘(w) باید شبکه عصبی چندلایه حساس به خطا و تحمل‌پذیر در برابر خطا را ایجاد کند، زیرا الگوریتم نه تنها E(w) بلکه مجموع مربعات وزن‌ها را به‌طور همزمان کاهش می‌دهد.

$$E'(w) = E(w) + \frac{\lambda}{2}\sum_{i,j} w_{ij}^2 \qquad (۵)$$

در رابطه (۵)، سیگما به‌معنی جمع همه وزن‌ها و λ پارامتری برای تعیین تأثیر مدت اضافی است. مدت اضافی (مجموع مربعات) به‌عنوان مدت جریمه عمل می‌کند تا اتصالات قوّی را تحت تأثیر قرار دهد. از آن‌جا که قدر مطلق هر کاهش وزن، تعداد اتصالات قوی را نیز کم می‌کند، لذا با انتخاب λ مناسب، این روش علاوه بر نگاشت مطلوب، تحمل‌پذیری خطا را نیز به دنبال دارد. در غیر این‌صورت، اگر λ خیلی بزرگ باشد باید تمام وزن‌ها به صفر میل کنند و اگر λ خیلی کوچک باشد، نباید هیچ اختلافی با الگوریتم پس‌انتشار داشته باشد. این روش به نام روش کاهش وزن (WMA) نامیده می‌شود.

اشکالی که روش WMA دارد این است که ابتدا تمایل به همگرایی E و سپس کاهش وزن دارد. با این وجود، نیاز به مراحل آموزشی بعد از همگرایی E است. برای غلبه بر این مشکل، تعدیل سازگار پارامتر λ پیشنهاد می‌شود. هدف این روش، کاهش E و مجموع مربعات به‌طور همزمان است.

$$w = \frac{1}{2}\sum_{i,j} w_{ij}^2 \quad , \quad \lambda = \lambda' \frac{grad(E)}{grad(w)} \qquad (۶)$$

در رابطه (۶)، λ' پارامتری ثابت است. پارامتر λ هر مرحله آموزش را سازگار می‌کند، به‌طوری که هر دو شرط E(w) تقریباً تأثیر یکسانی دارند. این روش به عنوان روش AWMA (WMA سازگار) نامیده می‌شود.

اگر چه هدف هر دو روش WMA و AWMA کاهش تعداد اتصالات قوی است، امّا این دو روش تمایل دارند به شدت بر روی ارتباطات بین واحدهای ورودی و پنهان (اتصال IH) به جای ارتباط بین لایه پنهان و خروجی (اتصال HO) تأثیر بگذارند. از سوی دیگر، یک خطا روی اتصال HO باعث آسیب بیشتری نسبت به یک خطا روی اتصال IH می‌شود. برای غلبه بر این تناقص، رویکرد جزئی از روش AWMA پیشنهاد شده است. این روش تنها وزن HO را به‌جای همه وزن‌ها کاهش می‌دهد. هدف این روش فقط کاهش وزن‌های HO که باعث آسیب‌های جدی می‌شوند، است. این روش، PAWMA (AMWBP جزئی) نامیده شده است.

۳-۱-۴- روش به حداقل رساندن وزن و محدود کردن فعالیت (WMRA)

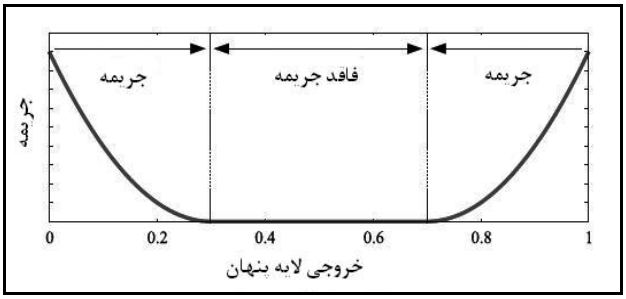
در [۶] آموزش الگوریتم تحمل‌پذیری خطا برای شبکه‌های عصبی چندلایه با تمرکز بر فعالیت واحدهای پنهان ارائه شده است. در این مقاله، برعکس نظر PAWMA نشان داده شده که بدترین خطا همیشه بین لایه پنهان و لایه خروجی واقع نمی‌شود. سپس یک الگوریتم آموزش تحمل‌پذیر جدید مبتنی بر WMA پیشنهاد شده است. تمرکز در این مقاله بر روی فعالیت واحدهای لایه پنهان است، زیرا نه تنها استحکام و قوت وزن، بلکه فعالیت واحدهای پنهان، نفوذ زیادی بر روی تحمل‌پذیری خطا دارد.

در این مقاله برای محدود نمودن خروجی واحدهای پنهان، یک اصطلاح جریمه جدید برای تابع ارزیابی، جهت آموزش معرفی شده است. این اصطلاح جریمه همان‌طور که در رابطه (۷) نشان داده شده است، برای واحدهای پنهانی که خروجی آن‌ها نزدیک صفر یا یک است، تعریف شده است.

$$P_{oh} = \sum_{i \in H} \rho(y_i, r)^2$$

$$\rho(y_i, r) = \begin{cases} y_i - 0.5 - r & If \quad y_i > 0.5 + r \\ y_i - 0.5 + r & If \quad y_i < 0.5 - r \\ 0 & otherwise \end{cases} \qquad (۷)$$

H مجموعه‌ای از واحدهای پنهان است. پارامتر r محدوده‌ای که شامل جریمه نمی‌شود را مشخص می‌کند. r>=0.5 به معنی عدم جریمه است. در این حالت الگوریتم پیشنهادی مشابه WMA است. مورد r=0 خروجی واحدهای پنهان برای هر ورودی ۰٫۵ را می‌سازد.



شکل ۲- تابع جریمه

در شکل (۲)، P_{oh} برای r=0.2 نشان داده شده است. باید مشخص شود که پارامتر r موجب اختلال فرآیند آموزش در این مورد نشود. با استفاده از این اصطلاح و جریمه تعریف‌شده برای WMA، تابع ارزیابی به‌صورت رابطه (۸)، اصلاح می‌شود.

$$E = E_{bp} + \lambda P_{wma} + \mu P_{oh} \qquad (۸)$$

در رابطه (۸)، μ پارامتری برای تصمیم‌گیری قدرت جریمه است. با بررسی روش‌هایی که ذکر شد، مشاهده می‌شود که این روش‌ها دارای نقاط ضعفی هستند که عبارت‌اند از:

- در روش‌های ذکر شده، بیشتر خطای وزن‌ها و خطاهایی که بر روی نرون‌های لایه میانی رخ می‌دهد، در نظر گرفته شده است. هیچ‌کدام از این روش‌ها به بررسی خطاهایی که نرون‌های لایه خروجی را تحت تأثیر قرار می‌دهد، نپرداخته‌اند. خطاهای نرون‌های لایه خروجی به‌دلیل آن‌که مستقیماً روی خروجی تأثیر دارند، بسیار مهم هستند.
- محدودیت‌های این روش‌ها این است که این روش‌ها در مقابل تعداد محدودی از خطاهایی که در شبکه عصبی رخ می‌دهد، مقاوم هستند. مشکل دیگر این است که این روش‌ها مربوط به بعد از یادگیری هستند. اگر موقعی که شبکه در حالت عملیاتی است، یک خطا رخ دهد، شبکه قادر به تشخیص خطا و مکان خطا نمی‌باشد.

۳-۲- تحمل‌پذیری خطا با استفاده از افزونگی

روش‌های افزونگی دسته دوم، روش‌هایی هستند که برای مقاوم کردن شبکه‌های عصبی در برابر خطا توسعه داده شده‌اند. افزونگی عبارت است از توانایی عملکردی سیستم که در شرایطی که محیط عاری از خطا باشد، آن توانایی لازم نیست. افزونگی می‌تواند مانند افزودن مؤلفه‌های سخت‌افزاری، افزودن بیت چک (کنترلی) به داده‌ها، یا افزودن چند خط دستور به مجموعه دستورات برنامه باشد، تا بررسی و تامین صحت نتایج اجرای برنامه را انجام دهد.

انواع افزونگی‌هایی که ممکن است وجود داشته باشد عبارت‌اند از:

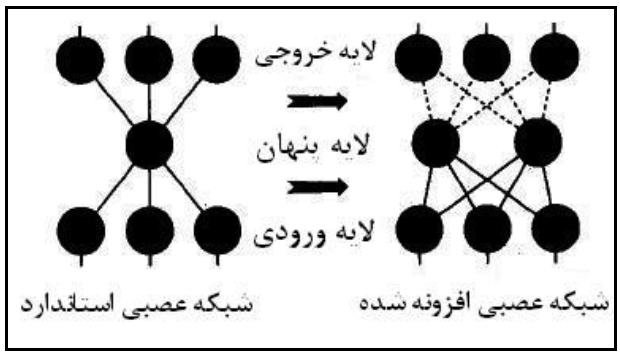
- افزونگی مکانی: مؤلفه‌ها، توابع و یا داده‌های اضافی که در شرایط عملیات بدون خطا ضروری نمی‌باشند.
- افزونگی زمانی: عملیات محاسباتی یا انتقال داده، تکرار شده و نتیجه حاصل از آن، با نتایج قبلی که ذخیره شده‌اند، مقایسه می‌شود.

ماهیت برخی خطاها در سیستم به‌گونه‌ای است که نمی‌توان آن‌ها را پیدا و رفع نمود، و یا حتی از وقوع آن‌ها پیشگیری کرد. خطاهای رخ‌داده در سیستم ممکن است بخش‌هایی از سیستم را تحت تاثیر خود قرار دهند. اگر سیستم بتواند براساس قابلیت‌های پیش‌بینی شده در آن، در چنین شرایطی، سرویس‌ها و خدمات پیش‌بینی شده خود را ارائه نماید، گفته می‌شود سیستم تحمل‌پذیر خطا است. ولی اگر برخی از سرویس‌ها را نتواند در سطح پیش‌بینی‌شده ارائه نماید یا اصلاً نتواند سرویس‌های خاصی را ارائه نماید گفته می‌شود سیستم به صورت تنزیل‌یافته[1]، کار می کند و توان آن تقلیل یافته است[۵].

در ادامه، چند کار مرتبط که از افزونگی برای تحمل‌پذیری خطا در شبکه‌های عصبی استفاده نموده‌اند، مورد بررسی واقع شده‌اند.

۳-۲-۱- روش افزودن نرون‌ها در لایه میانی

در [۱۰] از تکنیکی به نام افزودن[2] استفاده شده است. در این روش، نرون‌های لایه پنهان را مطابق شکل (۳) با ضریب n تکرار می‌کنند (n عددی بزرگتر از ۱ است) و وزن‌های آن لایه را بر n تقسیم می‌کنند. در نتیجه، تأثیر خطای نرون‌ها و وزن‌ها کم می‌شود. نتایج این کار در این مقاله نشان می‌دهد که شبکه‌هایی که از این روش استفاده می‌کنند نسبت به شبکه‌های نرمال در برابر خرابی اتصالات مقاوم‌تر هستند.



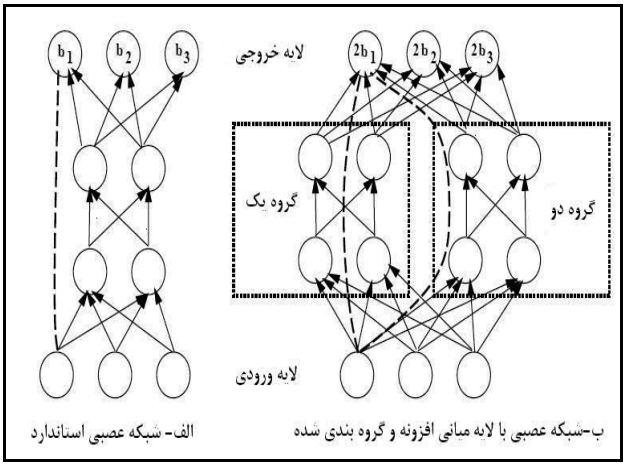
شکل ۳- افزودن نرون‌ها در لایه میانی

۳-۲-۲- روش افزودن و گروه‌بندی نرون‌ها

در [۱۱] روشی برای مقاوم نمودن شبکه‌های عصبی مصنوعی

1- Degradable
2- Augmentation

چندلایه توسعه داده شده است. رویه ارائه شده، نرون‌های لایه میانی شبکه عصبی را گروه‌بندی[1] و سپس این گروه‌ها را همان‌طور که در شکل (۴) مشاهده می‌شود با ضریب n تکرار و خروجی این گروه‌ها را با همین ضریب تنظیم می‌کند.



شکل ۴-تکرار و گروه‌بندی لایه‌های میانی

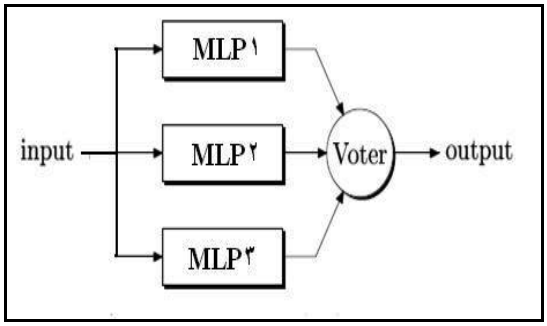
۳-۲-۳-روش افزونگی زمانی برای تحمل‌پذیری خطا در شبکه‌های عصبی

در مرجع [۱۲] روشی برای تحمل‌پذیری خطا در شبکه‌های عصبی چندلایه با استفاده از افزونگی زمانی مطرح شده است. در این روش با تکرار محاسبات و مقایسه نتایج به‌دست آمده از محاسبات مختلف، که بر روی یک شبکه انجام شده است، می‌توان تحمل‌پذیری خطا را در شبکه ایجاد نمود. در بعضی موارد افزونگی موقتی با اضافه کردن زمان، سخت افزار مورد نیاز برای افزونگی مکانی را کاهش می‌دهد.

۴- ارائه روش پیشنهادی

در روش‌هایی که از یادگیری و افزونگی استفاده نموده‌اند، بیشتر خرابی وزن‌ها و خرابی که بر روی نرون‌های لایه میانی رخ می‌دهد، در نظر گرفته شده است. هیچ‌کدام از این روش‌ها به بررسی خطاهایی که نرون‌های لایه خروجی را تحت تأثیر قرار می‌دهد، نپرداخته‌اند. خطاهای نرون‌های لایه خروجی به‌دلیل آن‌که مستقیماً روی خروجی تأثیر دارد، بسیار مهم هستند. هم‌چنین در سیستم‌های مهم و حیاتی، مانند سیستم‌های نظامی و دفاعی که از شبکه‌های عصبی برای کاربردهای خود استفاده می‌کنند، باید بتوان همیشه سیستم را در حالت عملیاتی حفظ نمود. برای این سیستم‌ها مهم این است که خروجی شبکه عصبی همواره در حالت پایدار و صحیح باشد. در ادامه، یک روش پیشنهادی جهت تحمل‌پذیری خطا در شبکه‌های عصبی چندلایه مورد بحث و بررسی قرار گرفته است.

در روش پیشنهادی، از سیستم افزونگی مؤلفه‌های سه‌گانه (TMR) برای تحمل‌پذیری خطا در شبکه‌های عصبی پرسپترون چندلایه استفاده شده است. در این روش، همان‌طور که در شکل (۵) نشان داده شده است، سه شبکه عصبی MLP با هم موازی شده‌اند و از یک رأی‌گیری اکثریت برای تعیین نتیجه، صحیح استفاده شده است. ورودی و خروجی هر یک از سه شبکه عصبی با هم برابر است، امّا ساختار داخلی شبکه‌های عصبی می‌تواند با یکدیگر متفاوت باشد. در نتیجه خطاهایی که ممکن است در هر یک از شبکه‌های عصبی رخ دهد، با یکدیگر متفاوت است. اگر خروجی یکی از شبکه‌های عصبی MLP با خطا مواجه شود، در این صورت رأی‌گیری با تشخیص صحیح نتایج شبکه‌های عصبی فاقد خطا، نتیجه مؤلفه خراب را می‌پوشاند.



شکل ۵- معماری پیشنهادی برای تحمل‌پذیری خطا در شبکه‌های عصبی چندلایه

سیستم TMR فقط می‌تواند یک مؤلفه دارای خطا را بپوشاند. چنان‌چه خرابی در سایر مؤلفه‌ها باشد سبب می‌شود که رأی‌گیر نتیجه نادرست را تولید نماید. رویکرد پوشش خطا در این معماری از روش رأی‌گیری برای حکمیت کردن بین نتایج شبکه‌های عصبی MLP افزونه‌شده استفاده می‌کند. رأی‌گیر برای مقایسه نتایج شبکه‌های عصبی افزونه‌شده، از یک سیستم جهت تعیین و تصمیم‌گیری نتیجه درست استفاده می‌کند. رأی‌گیر به‌عنوان نقطه واحد شکست در این معماری مطرح است. اگر رأی‌گیر در این معماری دچار خطا شود، کل سیستم دچار خطا شده و قابلیت اطمینان سیستم به شدت کاهش می‌یابد. بنابراین رأی‌گیر باید از نظر طراحی و توسعه‌یافتگی، دارای قابلیت اطمینان و کارایی بالایی باشد. روش رأی‌گیری در این معماری به‌صورت رأی‌گیری اکثریت است. در صورتی که حداقل ۲ مؤلفه به‌صورت صحیح کار کنند، نتیجه رأی‌گیر یک نتیجه درست است و اگر بیش از یک مؤلفه دچار خطا شود، آن‌گاه نتیجه رأی‌گیر یک نتیجه خطا است.

برای تخمین و برآورد اعتبار و نفوذ قابلیت اطمینان شبکه عصبی افزونه‌شده، لازم است که قابلیت اطمینان مؤلفه‌ها را به‌دست آوریم.

1- Grouping

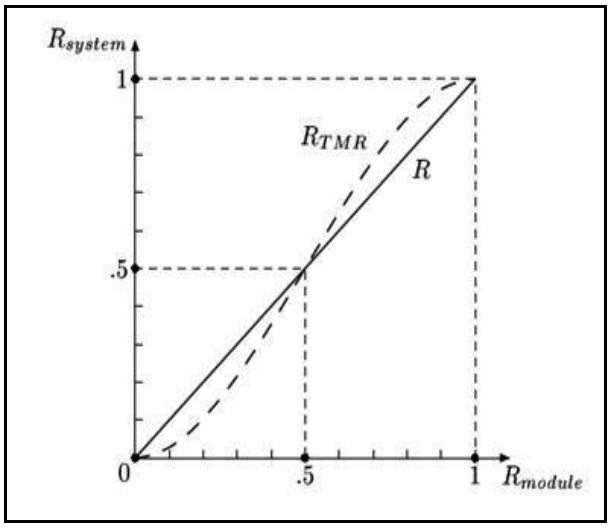
فرض بر این است که خرابی مؤلفه‌ها دوبه‌دو از هم مستقل هستند. در چنین شرایطی قابلیت اطمینان این سیستم مطابق رابطه (۹) است.

$$R_{TMR} = (R_{MLP1} * R_{MLP2} * R_{MLP3} + ((1-R_{MLP1}) * R_{MLP2} * R_{MLP3}) + (R_{MLP1} * (1-R_{MLP2}) * R_{MLP3}) + (R_{MLP1} * R_{MLP2} * (1-R_{MLP3}))) * R_{Voter} \quad (۹)$$

اگر معادله $R_{TMR1} = R_{MLP2} = R_{MLP3} = R$ برقرار باشد رابطه فوق به‌صورت رابطه (۱۰) خواهد بود.

$$R_{TMR} = (3R^2 - 2R^3) R_V \quad (۱۰)$$

شکل (۶)، نمودار قابلیت اطمینان شبکه عصبی افزونه‌شده را با قابلیت اطمینان یک شبکه عصبی ساده که شامل یک مؤلفه با قابلیت اطمینان R است، مقایسه می‌کند.

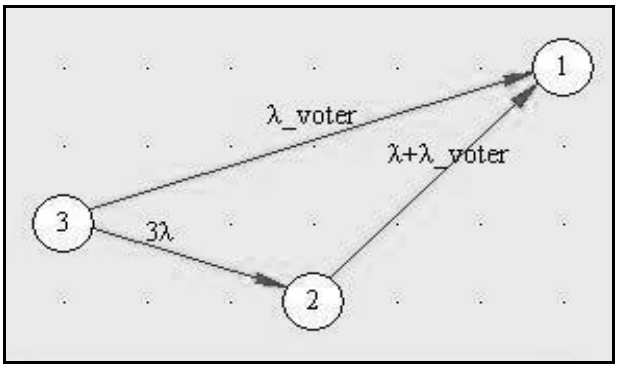


شکل ۶- مقایسه قابلیت اطمینان شبکه عصبی افزونه شده با شبکه عصبی فاقد افزونگی

همان‌طور که در شکل (۶) نشان داده شده است، نقطه‌هایی که در آن، قابلیت اطمینان دو سیستم مساوی است، سه نقطه ۰ و ۰,۵ و ۱ است. این موارد تحمل‌پذیری خطا و قابلیت اطمینان را توضیح می‌دهد. سیستم می‌تواند تحمل‌پذیر خطا باشد، ولی هنوز قابلیت اطمینان آن کم است. به‌عنوان مثال اگر شبکه عصبی افزونه‌شده، شامل مؤلفه‌های کیفیت پایین با R= 0.2 باشد، قابلیت اطمینان سیستم پایین خواهد بود و در این صورت $R_{TMR} = 0.136$ خواهد شد.

۵- مدل‌سازی و ارزیابی شبکه عصبی افزونه‌شده با استفاده از مدل مارکوف

برای مدل‌سازی سیستم پیشنهادی همان‌طور که در شکل (۷) آمده، از مدل مارکوف استفاده شده است.



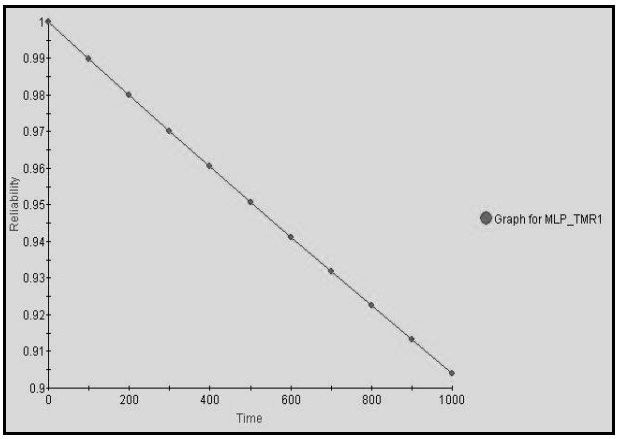
شکل ۷- مدل مارکوف شبکه عصبی افزونه شده

در این مدل، سیستم در گره‌های ۲ و ۳ در حالت عملیاتی است و به‌طور صحیح کار می‌کند و در گره ۱، سیستم معیوب است. در ابتدا که خطایی در شبکه وجود ندارد و هر سه مؤلفه MLP و رأی‌گیر به درستی عمل می‌کنند، سیستم در حالت (۳) قرار دارد. اگر در یکی از ۳ مؤلفه، خطایی به‌وجود آید با نرخ خرابی ۳λ، سیستم به حالت (۲) می‌رود. در حالت (۲) نیز سیستم همچنان به فعالیت خود ادامه می‌دهد و از آن‌جا که رأی‌گیری به‌صورت اکثریت انجام می‌شود، نتیجه رأی‌گیری که از دو مؤلفه بی‌عیب گرفته شده، به‌عنوان خروجی شبکه افزونه‌شده در نظر گرفته می‌شود. اگر سیستم در حالت (۲) قرار داشته باشد، با خرابی هریک از دو مؤلفه که به‌طور صحیح عمل می‌کنند، سیستم با نرخ خرابی λ به حالت (۱) می‌رود. در حالت (۱)، چون دو مؤلفه دارای خرابی می‌باشند، رأی‌گیر نمی‌تواند به اجماع برسد و در نتیجه، سیستم از حالت عملیاتی خارج می‌شود. نکته‌ای که در این‌جا مطرح است این است که اگر سیستم در حالت‌های (۲ و ۳) باشد و در رأی‌گیری خطایی با نرخ λ_voter ایجاد شود، سیستم به حالت (۱) می‌رود و از حالت عملیاتی خارج می‌شود.

برای مدل‌سازی این شبکه، از نرم‌افزار شارپ استفاده شده است. در این نرم‌افزار، ابتدا نرخ خرابی مؤلفه‌های MLP که در سیستم استفاده شده برابر با $1*10^{-4}$ و نرخ خرابی رأی‌گیر برابر با $1*10^{-6}$ در نظر گرفته شده است. شکل (۸) نمودار قابلیت اطمینان شبکه عصبی افزونه‌شده بر مبنای زمان را نشان می‌دهد.

چون در این معماری، رأی‌گیری بدون خطا فرض نشده و هریک از سه مؤلفه نیز دارای یک نرخ خرابی معینی هستند، بنابراین همان‌طور

که در شکل (۸) نشان داده شده است، با گذشت زمان، قابلیت اطمینان سیستم کاهش می‌یابد. قابلیت اطمینان با زمان رابطه معکوس دارد و هرچه زمان می‌گذرد، قابلیت اطمینان سیستم کاهش می‌یابد و این یک امر طبیعی است. با توجه به شکل (۸)، ابتدا قابلیت اطمینان سیستم در لحظه صفر برابر با یک است. پس از گذشت زمان در لحظه ۱۰۰۰، قابلیت اطمینان سیستم کاهش یافته و به عدد ۰/۹۱ می‌رسد.

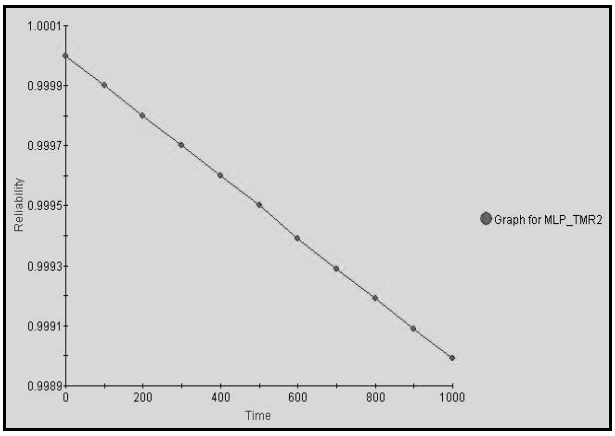


شکل ۸- نمودار قابلیت اطمینان شبکه عصبی افزونه‌شده بر مبنای زمان

نکته‌ای که در این‌جا وجود دارد این است که هرچه بتوان نرخ خرابی را در مؤلفه‌ها و رأی‌گیر کاهش داد، می‌توان قابلیت اطمینان بالاتری را به‌دست آورد و شیب منحنی نیز کمتر می‌شود. به عنوان مثال اگر نرخ خرابی در مؤلفه‌ها به عدد ۱×۱۰^{-۶} و نرخ خرابی در رأی‌گیر به عدد ۱×۱۰^{-۸} تغییر پیدا کند، آن‌گاه قابلیت اطمینان سیستم مطابق شکل (۹) است. همان‌طور که در این شکل نشان داده شده است، در لحظه صفر، قابلیت اطمینان سیستم برابر ۱ است و با گذشت زمان در لحظه ۱۰۰۰، قابلیت اطمینان سیستم به عدد ۰/۹۹۹۰ می‌رسد. مشاهده می‌شود که قابلیت اطمینان سیستم در این حالت نسبت به حالتی که در شکل (۸) نشان داده شده است، بیشتر است. بنابراین هرچه بتوان نرخ خرابی را در مؤلفه‌ها و رأی‌گیر کاهش داد می‌توان به قابلیت اطمینان بالاتری دست یافت.

۶- خلاصه و نتیجه‌گیری

در این مقاله ابتدا شبکه‌های عصبی چندلایه و تحمل‌پذیری خطا در آن‌ها مورد بررسی قرار گرفت. سپس روش‌های ارائه‌شده برای افزایش قابلیت تحمل‌پذیری خطا در شبکه‌های عصبی بررسی شد. به‌طور



شکل ۹- نمودار قابلیت اطمینان شبکه عصبی افزونه‌شده بر مبنای زمان با تغییر در نرخ خرابی مؤلفه‌ها و رأی‌گیر

کلی دو روش جهت تحمل‌پذیری شبکه‌های عصبی وجود دارد. دسته اول شامل روش‌هایی است که با تغییر الگوریتم یادگیری یا اضافه کردن برخی ویژگی‌ها در حین آموزش، مانند تزریق خطا و آموزش شبکه در حین این خطا، سعی در افزایش تحمل‌پذیری خطا در شبکه‌های عصبی دارند. روش دوم که در این مقاله به آن اشاره شده، روش‌های افزونگی برای تحمل‌پذیری خطا در شبکه‌های عصبی هستند. این روش‌ها شامل افزونگی سخت‌افزاری و زمانی می‌باشند. روش‌هایی که تاکنون مطرح شد بیشتر خطای وزن‌ها و خطایی که بر روی نرون‌های میانی رخ می‌دهد را بررسی نموده‌اند. هیچ‌کدام از این روش‌ها به بررسی خطاهایی که نرون‌های لایه خروجی را تحت تأثیر قرار می‌دهد، نپرداخته‌اند. خرابی نرون‌های لایه خروجی به‌دلیل این‌که مستقیماً روی خروجی تأثیر می‌گذارند، بسیار مهم هستند. برای رفع این عیب، در این مقاله یک معماری افزونگی مؤلفه سه‌لایه (TMR) برای شبکه‌های MLP پیشنهاد شد. ساختار این معماری شامل سه شبکه عصبی MLP که به‌صورت موازی عمل می‌نمایند و یک رأی‌گیر اکثریت که عمل مقایسه بین نتایج خروجی این سه شبکه را انجام می‌دهد، تشکیل شده است. مزیت این معماری در این است که اگر در هر یک از مؤلفه‌های داخلی شبکه‌های عصبی (شامل نرون‌های ورودی، میانی و خروجی، وزن‌ها و تابع محرک) خطایی به‌وجود آید، سیستم این خطا را با استفاده از نتیجه دو مؤلفه‌ی دیگر می‌پوشاند و خروجی صحیح را ایجاد می‌نماید. در پایان ذکر این نکته لازم است که مباحث تحمل‌پذیری خطای شبکه‌های عصبی که در کاربردهای نظامی و دفاعی کشور استفاده می‌شود، به‌عنوان یکی از موضوعات اساسی پژوهش‌ها، جهت تحقق مباحث پدافند غیرعامل مطرح خواهد بود.

مراجع

۱. صالحی، مصطفی؛ پیاده‌سازی شبکه‌های عصبی بر روی FPGA؛ پایان‌نامه کارشناسی ارشد؛ دانشگاه علم و صنعت ایران؛ تهران؛ (**۱۳۸۴**).

۲. حسنی آهنگر، محمدرضا؛ پایش سیستم‌های بی‌درنگ جهت تشخیص خطا با استفاده از فنون محاسبات نرم؛ رساله دکتری؛ دانشگاه علم و صنعت ایران؛ تهران؛ (**۱۳۹۰**).

۳. احمدی، علی؛ پیاده‌سازی سخت‌افزاری شبکه‌های عصبی با قابلیت تحمل‌پذیری خرابی؛ پایان‌نامه کارشناسی ارشد، دانشگاه تهران، تهران؛ (**۱۳۸۷**).

4. Fiszelew, A., Britos, P., Ochoa, A., Merlino, H., Fernandez, E., Garcia-Martinez, R; Finding Optimal Neural Network Architecture Using Genetic Algorithms; Advances in Computer Science and Engineering Research in Computing Science 27;page 15-24; (**2007**).
5. Dubrova, Elena; 'Fault Tolerant Desing:AN Introduction'; Department of Microelectronics and Information Technology Royal Institute of Technology Stockholm, Sweden; Kluwer Academic Publishers; (**2008**).
6. Haruhiko,Takase, KITA Hidehiko, HAYASHI Terumine; 'Fault Tolerant Training Algorithm For Multi-Layer Neural Networks Focused On Hidden Unit Activities', IEEE; (**2006**).
7. Ito, Takehiro, Takanami, Itsuo; 'On Fault Injection Approaches for Fault Tolerance of Feedforward Neural Networks', IEEE, 1081-7735/9, page 88-93; (**1997**).
8. Charif Cammadi, Nait, Ito, Hideo; 'A Learning Algorithm for Fault Tolerant Feedforeard Neural Networks', IEICE TRAN; (**1996**).
9. Takase, Haruhiko, Hidehiko, Kita, Tetumine, Hayashi; ' Effect of Regularization Term upon Fault Tolerant Training'; IEEE, pp1048-1053; (**2003**).
10. D.Emmerson, Martin, I.Damper, Robert; 'Derermining and Improving the Fault Tolerance of Multilayer Perceptrons in a Pattern-Recognition Application'; IEEE Trans-action ON Neural Networks,Vol 4,No 5; (**1993**).
11. D. S. Phatak, I. Koren, 'Complete and Partial Fault Tolerance of Feedforward Neural Nets'; IEEE Transaction ON Neural Nets, Vol 6,No 2; pp 446-456; (**1995**).
12. Yuang-Ming Hsu, E. Swartzlander ,Earl, Vincenzo Piuri, Jr,;'Recomputing by Operand Exchanging: a Time-redundancy Approach for Fault-tolerant Neural Networks', IEEE,pp54-65; (**1995**).

Fault Tolerance in the MLP Neural Networks Using Triple Modular Redundancy

M. R. Hasani Ahangar[1]

M. Akhzami [2]

Abstract

The use of complex and large systems and infrastructure is vital to carry out various activities in a country, and observance issues of passive defense in critical situations that could provide complete or partial services, is considered one of the evaluation criteria for these systems. Modeling and simulation of these systems to identify bottlenecks is important. Fault occurrance seems natural despite various provisions such as fault forecasting , fault prevention, fault coverage and fault tolerance. Artificial neural networks as a method of modeling and simulation, have many applications in monitoring complex and critical systems. Since artificial neural networks have been designed based on natural neural networks model, that possess the inherent capability of fault tolerance. Therefore, they must be able to take advantage of fault tolerance ability. This article presents a method to enhance the fault tolerance and improve neural networks based on Triple Modular Redundancy (TMR). It shows that, based on this technique, the desired fault tolerance has been favorably increased.

Key Words: *Multi-Layered Neural Networks*, *Fault Tolerance*, *Redundancy*, *Passive Defense*, *TMR*

1- Assistant Professor and Academic Member of Imam Hossein Comprehensive University (mrhassani@iust.ac.ir)
2- M.S Candidate of Computer , of Imam Hossein Comprehensive University (mostafa_akhzami@yahoo.com) - Writer in Charge