

**پیش بینی قابلیت اطمینان نرم افزار با استفاده از روش محاسبات نرم**

**چکیده**

در این مقاله، مدل‌های گروهی برای دقت پیش‌بینی قابلیت اطمینان نرم‌افزار توسعه یافته است. روش‌های آماری (رگرسیون خطی چندگانه و رگرسیون خطی چند متغیره) و تکنیک‌های هوشمند (شبکه عصبی آموزش دیده پس انتشار، سیستم استنتاج عصبی-فازی پویا و TreeNet) گروه‌ها را ارائه می‌دهند. سه گروه خطی و یک گروه غیرخطی طراحی و تست شده است. براساس آزمایش روی قابلیت اطمینان داده‌های بدست آمده از نرم‌افزار، مشاهده شده است که مجموعه گروه‌های غیرخطی عملکرد بهتری نسبت به گروه‌های دیگر و همچنین روش‌های آماری و هوشمند دارند.

**کلمات‌کلیدی:** قابلیت اطمینان پیش‌بینی نرم‌افزار؛ ریسک عملیاتی؛ مدل‌های پیش‌بینی گروه؛ تکنیک‌های هوشمند؛ محاسبات نرم

**1. معرفی**

قابلیت اطمینان نرم‌افزار به‌عنوان احتمال خطای عملیات نرم‌افزار برای یک دوره مشخص از زمان در یک محیط مشخص شده است (تعریف ANSI). مدل‌سازی قابلیت اطمینان نرم‌افزار اهمیت بسیاری در سالهای اخیر به‌دست آورده است. بحران نرم‌افزار در بسیاری از برنامه‌های کاربردی حاضر رو به افزایش است. کاربرد شبکه‌های شبکه هوشمند و هیبرید عصبی به‌جای روش‌های آماری سنتی بهبود قابل توجهی در پیش‌بینی قابلیت اطمینان نرم‌افزار در سال‌های اخیر نشان داده است. در میان روش‌های هوشمند و آماری شناسایی بهترین آسان است زیرا عملکرد آن‌ها با تغییر در داده متفاوت است.

در این مقاله، یک رویکرد مبتنی بر گروه برای پیش‌بینی قابلیت اطمینان نرم‌افزار تشریح شده است. به‌طورخاص، شبکه عصبی غیرخطی پس انتشار با استفاده از گروه آموزش دیده (BPNN) ارائه شده است. روش پیشنهادی تمام قابلیت‌های تکنیک پیش‌بینی را به سمت داده و اختصاص مناسب وزن به هر از تکنیک‌های مبتنی بر عملکرد آنها برده است.

ساختار مقاله به‌صورت ذیل سازمان یافته است. در بخش 2، بررسی مختصری از کاهای انجام گرفته در پیش‌بینی قابلیت اطمینان نرم‌افزار ارائه شده است. در بخش 3، روش‌هایی مختلف هوشمند که در این مقاله به‌کار برده شده است به‌طور خلاصه شرح دادیم. در بخش 4، چهار گروه توسعه داده شده ارائه شده است. در بخش 5 روش تجربی ارائه شده است. بحث در مورد نتایج را در بخش 6 بیان کرده‌ایم. در بخش 7، کاربرد مدلسازی دقیق عملیاتی خطر در بانک‌ها ارائه شده است. در نهایت، بخش 8 نتیجه‌گیری مقاله را بیان می‌کند.

**2. کارهای گذشته**

در سال‌های اخیر بسیاری از مطالعات تحقیقاتی انجام شده در این زمینه در مورد مدل‌سازی قابلیت اطمینان نرم‌افزار و پیش‌بینی آن است. که شامل استفاده از شبکه‌های عصبی، مدل منطق فازی؛ الگوریتم ژنتیک (GA) براساس شبکه‌های عصبی، شبکه‌های عصبی راجعه، شبکه‌های عصبی بیزی و ماشین بردار پشتیبان (SVM) بر اساس تکنیک‌هایی، به‌نام Cai و همکارانش (1991) برای حمایت از توسعه‌ی مدل قابلیت اطمینان نرم‌افزار فازی به‌جای مدل قابلیت اطمینان نرم‌افزار احتمالاتی (PSRMs) بود.

استدلال آنها در اثبات براساس قابلیت اطمینان نرم‌افزار فازی در طبیعت بود. همچنین چگونگی توسعه‌ی یک مدل فازی برای توصیف قابلیت اطمینان نرم‌افزار ارائه شد. Karunanidhi و همکارانش (1992) یک مطالعه‌ی دقیق برای توضیح استفاده از مدل‌های پیش‌بینی در رشد قابلیت اطمینان نرم‌افزار انجام دادند. نتایج تجربی نشان داده است که مدل‌های پیوندگرا انطباق خوبی در سراسر مجموعه داده‌های مختلف دارند و نمایش بهتری در پیش‌بینی دقت نرم‌افزار نسبت به مدل‌های قابلیت اطمینان تحلیلی شناخته شده دارند. Sitte (1999) یک مطالعه‌ی مقایسه‌ای در شبکه‌های عصبی و مدل پارامتری-کالیبراسیون مجدد در پیش‌بینی قابلیت اطمینان نرم‌افزار انجام داد و مشاهده کرد که شبکه‎‌های عصبی برای استفاده بسیار ساده‌تر هستند و همچنین پیش‌بینی بهتر دارند. همچنین، از طریق نتایج تجربی نشان داده که مدل‌های شبکه عصبی پیش‌بینی روند بهتری دارند. Ho و همکارانش (2003) یک مطالعه‌ی جامع از مدل‌های پیوندگرا و کاربرد آنها در پیش‌بینی قابلیت اطمینان نرم‌افزار انجام داد و دریافت که این مدل‌ها انعطاف‌پذیرتر از مدل‌های سنتی هستند. یک مطالعه‌ی مقایسه‌ای بین شبکه‌های عصبی راجعه پیشنهادی آن‌ها، با شبکه‌های عصبی محبوب‌تر و مدل جردن و برخی از مدل‌های نرم‌افزار اعتماد سنتی انجام گرفت. نتایج عددی نشان می‌دهد که معماری شبکه‌ی پیشنهادی نسبت به مدل‌های دیگر پیش‌بینی در شرایط بهتری قرار دارد. علی‌رغم پیشرفت‌های اخیر در مدل‌های رشدیافته‌ی قابلیت اطمینان نرم‌افزار مشاهده شد که مدل‌های متفاوت دارای قابلیت‌های پیش‌بینی متفاوتی هستند و همچنین هیچ الگوی واحدی تحت هر شرایطی مناسب نیست.

TianوNoore (2005a) یک مدل تطبیقی پیش‌بینی قابلیت اطمینان نرم‌افزار با استفاده از رویکرد تکاملی پیوندگرا براساس معماری ورودی‌های چند تاخیری و تک خروجی ارائه داده‌اند. روش پیشنهادی، توسط نتایج نشان داده شده و تا به‌حال عملکرد بهتری با توجه به گام بعدی پیش‌بینی در مقایسه با مدل‌های موجود شبکه عصبی برای پیش‌بینی زمان شکست داشته است. Tian و Noore (2005b) یک روش مدل‌سازی شبکه عصبی تکاملی برای پیش‌بینی زمان شکست نرم‌افزار ارائه کرده است. نتایج آنها نشان می‌دهد که نسبت به مدل‌های شبکه عصبی موجود بهتر است. همچنین نشان داده است که معماری شبکه‌های عصبی تاثیر به‌سزایی بر عملکرد شبکه دارد. باتوجه به Bai و همکارانش (2005) شبکه‌های بیزی توانایی قوی برای انطباق در مشکلات مربوط به عوامل پیچیده نشان می‌دهد. آنها یک مدل پیش‌بینی نرم‌افزار براساس شبکه‌های مارکوف بیزی و یک روش برای حل مدل شبکه پیشنهاد داده‌اند. Reformat (2005) یک رویکرد منجر به استخراج دانش چند تکنیکی پیشنهاد داد و یک سیستم پیش‌بینی متا مدل جامع در حوزه نگهداری اصلاحی از نرم‌افزار ارائه داد. سیستم براساس نظریه شواهد و تعدادی از مدل‌های مبتنی بر فاز بود. علاوه‌براین آنها یک مطالعه موردی دقیق برای تخمین تعداد نقص در یک سیستم تصویربرداری پزشکی با استفاده از روش ارائه شده انجام دادند. Pai و Hong (2006) ماشین بردار پشتیبان (SVM ها) را برای پیش‌بینی قابلیت اطمینان نرم‌افزار در الگوریتم شبیه‌سازی دوباره (SA) برای انتخاب پارامترهای مدل SVM مورد استفاده قرار دادند. نتایج تجربی نشان می‌دهد که مدل ارائه شده پیش‌بینی بهتری نسبت به روش‌های مقایسه شده دارد. Su و Huang (2006) نشان داد که چگونه شبکه‌های عصبی را برای پیش‌بینی قابلیت اطمینان نرم‌افزار به کار ببریم. بیشتر آنها از روش شبکه‌های عصبی برای ساخت یک مدل وزن‌دار ترکیبی پویا (DWCM) استفاده کردند و نتایج تجربی نشان می‌دهد که مدل پیشنهادی به‌طور قابل توجهی پیش‌بینی بهتری دارد. همچنین به‌تازگی، شبکه‌های عصبی برای پیش‌بینی خطا در نرم‌افزار شی‌گرا (Kanmani و همکارانش، 2007) به‌کار برده شده است. این مطالعه نشان می‌دهد که مدل‌های شبکه عصبی خیلی بهتر از روش‌های آماری هستند.

استفاده از تکنیک‌های هوشمند به‌جای تکنیک‌های آماری در سال‌های اخیر با فرازونشیب بسیار همراه بوده است. استفاده از تکنیک‌های محاسبات نرم در مهندسی قابلیت اطمینان نرم‌افزار به‌تازگی وارد عرصه شده است (Madsen و همکارانش، 2006). باوجود پیشرفت‌های اخیر در مدل‌های رشد قابلیت اطمینان نرم‌افزار، مشاهده می‌شود که مدل‌های متفاوت دارای قابلیت‌های متفاوت پیش‌بینی هستند و همچنین هیچ الگوی واحدی تحت هر شرایطی مناسب نیست. گروه از خروجی به‌دست آمده از ترکیبات منحصربه‌فرد به‌عنوان ورودی استفاده می‌کند و داده‌ها باتوجه به طراحی داور دروغ گفتن در قلب گروه پردازش می‌شوند.

**3. بررسی اجمالی تکنیک‌های به‌کار گرفته**

روش های زیر برای پیش‌بینی قابلیت اطمینان نرم‌افزار به‌کار می‌روند (1) شبکه عصبی پس‌انتشار (BPNN)، (2) شبکه‌های عصبی مبتنی بر آستانه-پذیرش-(TANN) (Ravi و Zimmermann، 2001)، (3) شبکه پی-سیگما (PSN)، (4) رگرسیون خطی چندمتغیره ​​(MARS)، (5) رگرسیون تعمیم شبکه عصبی (GRNN)، (6) رگرسیون چند خطی (MLR)، (7) سیستم استنتاج فازی عصبی پویا (DENFIS) و (8) TreeNet. بنابه‌این‌که BPNN و MLR بسیار محبوب هستند، در اینجا مورد بحث قرار نگرفتند. همه اجزای باقی مانده از گروه به‌طور خلاصه در زیر بخش‌های بعدی توضیح داده شده است.

**3.1 شبکه‌های عصبی مبتنی‌بر آستانه پذیرش**

آستانه‌ی پذیرش (TA)، در اصل توسط Dueck و Scheuer (1990) پیشنهاد شد که یک نوع سریع‌تر از الگوریتم اصلی ژنتیک بود که در آن پذیرش یک حرکت جدید یا راه‌حل توسط یک تعیین معیار قطعی به‌جای یک مقدار احتمالی انجام می‌گرفت. جنبه غالب TA این است که هر راه‌حل جدیدی را که خیلی بدتر از جریان فعلی نیست می‌پذیرد. مسئله‌ی دشوار در مواجه با الگوریتم آموزش براساس TA (Ravi و Zimmermann ، 2001، 2003؛ Ravi و همکارانش، 2005) برای شبکه عصبی رو به جلو این است که "پاس رو به جلو" از الگوریتم انتشار به عقب توزیع نشده است. اما، در پاس رو به عقب، که تمام وزن‌ها به‌روز می‌شود، TA به‌جای شدیدترین الگوریتمی در خطای پس‌انتشار استفاده می‌شود. در این زمینه، مجموعه وزن‌های شبکه عصبی (هر دو گره ورودی به پنهان و پنهان به خروجی) به بردار متغیرهای تصمیم‌گیری تبدیل می‌شوند. نویسنده دوم با TANN کدگذاری شده است.

**3.2 شبکه PI-Sigma (PSN)**

شبکه پی-سیگما (PSN) در اصل توسط Shin و Ghosh پیشنهاد شد (1991). که یک شبکه رو به جلو با یک لایه‌ی پنهان است و در آن تعدادی از واحدهای پنهان (به‌عنوان "واحد جمع" نامیده می‌شود) نشان‌دهنده‌ی سفارش از شبکه هستند، که می‌تواند بسته به تنوع مورد نیاز باشند. در لایه‌ی خروجی واحدهای محصول وجود دارد که خروجی آن عملکرد محصول از واحد خروجی جمع است. در هر تکرار الگوریتم، تا زمانی که معیار همگرایی برآورده شود، یکی از واحدهای جمع به‌طور تصادفی انتخاب خواهد شد و وزن‌های مربوطه از لینک‌ها به آن گره‌هایی که بنا به قانون یکسانی به روز شده‌اند به قانون دلتا متصل می‌شوند.

**3.3 رگرسیون خطی چندمتغیره (MARS)**

​​ رگرسیون خطی چندمتغیره ​​(MARS) بود توسط فریدمن معرفی شده است (1991). MARS یک ابزار مدل‌سازی نوآورانه و انعطاف‌پذیر است که ساختمان مدل‌های پیش‌بینی مستمر و دقیق و متغیرهای وابسته باینری را به‌صورت خودکار به‌کار می‌گیرد. که در پیدا کردن تحولات و تعاملات متغیر بهینه و ساختار داده‌های پیچیده که اغلب در داده‌های چندبعدی پنهان هستند برتری دارد. برای انجام این طرح، رویکرد جدیدی برای مدل رگرسیون، به‌طور موثر الگوهای داده‌های مهم و روابطی که مشکل هستند آشکار می‌کند و اگر غیرممکن نباشد، برای روش‌های دیگر نیز نشان می‌دهد. MARSکه در این مقاله مورد استفاده قرار گرفته است در این لینک دردسترس است (<http://salford-systems.com/>).

**3.4 رگرسیون شبکه‌های عصبی تعمیم یافته (GRNN)**

Specht در (1991) GRNN را معرفی کرد. می‌توان آن را به‌عنوان یک شبکه‌ی تابع شعاع پایه‌ای نرمال (RBF) در واحد پنهان که در هر مورد آموزش وجود دارد تصور کرد. واحدهای RBF '' kernel '' نامیده می‌شوند و معمولا تابع چگالی احتمال مانند گاوس هستند. وزن‌های پنهان-خروجی فقط مقادیر هدف هستند، بنابراین خروجی یک میانگین وزنی از مقادیر هدف نزدیک به موارد آموزش داده شده در ورودی داده هستند. تنها وزنی که نیاز به یادگیری دارد عرض واحدهای RBF است. این عرض‌ها (اغلب عرض تنها استفاده می‌شود) به‌نام '' پارامتر ساده '' یا '' پهنای باند '' نامیده می‌شوند و معمولا توسط اعتبار متقاطع یا با روش‌های باطنی که در شبکه عصبی به خوبی شناخته شده نیست انتخاب می‌شوند. گرادیان نزولی استفاده نمی‌شود. GRNN یک تخمین کننده‌ی جهانی برای توابع ساده است، بنابراین باید قادر به حل هر تابع تقریب ساده با داده‌های به‌اندازه کافی باشد. ایراد اصلی GRNN آن است که مثل روش هسته، به‌شدت از ضعف ابعاد رنج می‌برد. GRNN نمی‌تواند ورودی بی‌ربط به الگوریتم اساسی را بدون تغییرات عمده نادیده بگیرد. GRNN موجود در MATLAB 6.5 در این مقاله استفاده شده است.

**TreeNet 3.5**

TreeNet توسط فریدمن (1999) معرفی شد. که از یک مفهوم جدید '' یادگیری فوق‌العاده آهسته '' در لایه‌هایی که در آن اطلاعات به‌تدریج به ساختار داده‌ها تبدیل می‌شوند استفاده می‌کند. مدل TreeNet معمولا متشکل از صدها درخت کوچک است، که هر کدام فقط یک تنظیم کوچک به مدل کلی است. TreeNet به خطاهای داده حساس است و نیاز به هیچ پیش پ ردازش وقت‌گیر و یا انتساب ارزش از دست رفته‌ای ندارد. TreeNet به تمرین مقاوم است و از یک شبکه عصبی سریع‌تر است. TreeNet موجود در (http://salford-systems.com/) در این مقاله استفاده شده است.

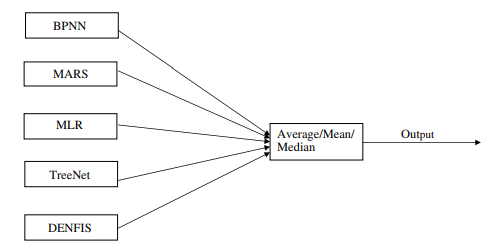
**3.6 سیستم استنتاج عصبی-فازی پویا (DENFIS)**

DENFIS توسط Kasabov (2002) معرفی شد. DENFIS از طریق افزایش، آموزش هیبریدی (تحت نظارت/بدون نظارت) تکامل یافته است و با داده‌های ورودی جدیدی تطابق یافته است، از جمله ویژگی‌های جدید، کلاس‌های جدید و غیره، از طریق تنظیم عنصر محلی است. قوانین فازی جدید ایجاد شده و در طول بهره‌برداری از سیستم به روز شده‌اند. در هر زمانی، خروجی DENFIS از طریق یک سیستم استنتاج فازی براساس قوانین فازی فعال، که به صورت پویا از یک مجموعه فازی انتخاب شده است محاسبه‌ می‌شود. مجموعه‌ای از قوانین فازی را می‌توان قبل از داده و یا در طول فرایند یادگیری به DENFIS داد. قوانین فازی همچنین می‌توانند در طول یا پس از فرایند یادگیری استخراج شوند. DENFIS موجود در نسخه‌ی دانش‌آموز که از ابزار NEWCOM ([www.aut.ac.nz](http://www.aut.ac.nz)) به‌دست آمده در این مقاله استفاده شده است.

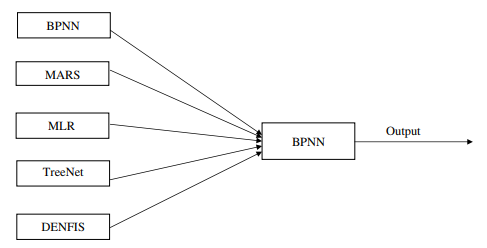
**4. مدل‌های پیش‌بینی گروه**

ایده‌ی پشت سیستم گروه، بهره برداری از ویژگی‌های منحصربه‌فرد برای تشکیل الگوهای مختلف است که در مجموعه داده وجود داشته باشد. هر دو نتایج نظری و تجربی نشان می‌دهد که می‌تواند یک راه موثر و کارآمد به‌منظور بهبود دقت باشد. Bates و Granger (1969) در کار خود نشان دادند که ترکیب خطی تکنیک‌های مختلف خطای واریانس کوچکتری از هر تکنیک در حالت مستقل دارد. از آن به بعد، بسیاری از محققان بر روی گروه و یا پیش‌بینی ترکیب کار می‌کنند. Makridakis و همکارانش (1982) ترکیب چندین مدل رایج در بهبود دقت پیش‌بینی را ارائه کردند. سپس، Pelikan و همکارانش (1992) ترکیب چند شبکه‌‌ی عصبی رو به جلو به‌منظور بهبود دقت زمان پیش‌بینی و صحت پیشنهاد کردند. برخی از تکنیک‌های گروه برای پیش‌بینی مشکلات با متغیر وابسته شامل گروه خطی (به‌عنوان مثال، میانگین ساده؛ Benediktsson و همکارانش، 1997)، وزن متوسط ​​(Perrone و Cooper، 1993) و رگرسیون انباشته (Breiman، 1996) و گروه غیرخطی (به‌عنوان مثال، شبکه عصبی مبتنی بر گروه غیرخطی (Yu و همکارانش، 2005)) هستند.

Hansen و همکارانش (1992) گزارش دادند که تعمیم توانایی یک سیستم شبکه عصبی می‌تواند به طور قابل توجهی بااستفاده از مجموعه‌ای از شبکه‌های عصبی بهبود یابد. هدف رسیدن به بهبود کلی دقت بر روی داده‌های تولید شده است. به‌طورکلی، برای طبقه‌بندی مشکلات، یک سیستم گروه، تصمیم‌گیری طبقه‌بندی منحصربه‌فرد را به‌طور معمول توسط رای اکثریت برای طبقه‌بندی نمونه‌های جدید ترکیب می‌کند. ایده اصلی، آموزش مجموعه‌ای از مدل‌ها (کارشناسان) و اجازه به رای‌دهی آن‌ها است. در طرح رای اکثریت، تمام مدل‌های منحصربه‌فرد اهمیت مساوی دارند. یکی دیگر از مدل‌های ترکیب از طریق رای‌گیری وزن‌دار است، که در آن مدل‌های منحصربه‌فرد از اهمیت نابرابری برخوردار هستند. که با اتصال برخی از وزن‌ها به پیش‌بینی داده شده توسط مدل منحصربه‌فرد و سپس ترکیب آنها همراه است. Olmeda و Fernandez (1997) یک سیستم گروه بر اساس الگوریتم ژنتیک ارائه کردند، که در آن GA ترکیب بهینه از مدل‌های فردی را به‌طوری‌که به حداکثر دقت برسند تعیین می‌کند. Zhou و همکارانش (2002) یک مطالعه دقیق بر روی شبکه‌های عصبی گروه انجام دادند و پیشنهاد دادند که استفاده از مجموعه‌ای از شبکه عصبی بهتر از استفاده از تمام شبکه‌های عصبی است. آنها یک روش پیشنهاد دادند که می‌تواند برای انتخاب شبکه‌های عصبی برای تبدیل شدن به بخشی از گروه مجموعه‌های موجود در شبکه‌های عصبی مورد استفاده قرار گیرد.



شکل 1: طراحی عمومی گروه خطی



شکل 2: طراحی کلی از گروه غیرخطی

الگوریتم ژنتیک برای نسبت دادن وزن به اجزاء شبکه‌ مورد استفاده قرار گرفت.

به‌طورکلی موردی است که برای یک مجموعه داده از روش هوشمند بهتر است و زمانی‌که یک مجموعه داده مختلف مورد استفاده قرار می‌گیرد به‌طورکامل نتایج می‌تواند مخالف باشد. به‌منظور از دست ندادن کلیت و همچنین ترکیب مزایای استفاده از تکنیک‌های هوشمند، گروه، از خروجی تکنیک مستقل هوشمند با هر سطح اولویت خاصی استفاده می‌کند و با کمک یک داور خروجی را فراهم می کند.

یک گروه از خروجی به دست آمده از ترکیبات فردی به‌عنوان ورودی استفاده می‌کند و داده‌های با توجه به طراحی داور پردازش می‌شود. چهار نوع مختلفاز گروه‌ها طراحی و مانند شکل 1 و 2 به‌کار برده می‌شود. این خدمات عبارتند از (1) گروه خطی براساس میانگین، (2) گروه خطی براساس میانگین وزنی، (3) گروه خطی براساس میانه وزن و در نهایت (4) یک گروه غیرخطی در BPNN است. این گروه‌ها به‌طور خلاصه به شرح زیر است.

**4.1 گروه خطی براساس میانگین**

برای هر مشاهده، مقادیر خروجی از مولفه‌های فردی به‌عنوان ورودی برای مجموعه گرفته می‌شود و میانگین این مقادیر توسط این گروه‌ها به‌عنوان خروجی تولید می‌شود. این ساده‌ترین نوع گروه برای تصور کردن است.

**4.2 گروه خطی براساس میانگین وزنی**

در این گروه، مقادیر خروجی همانگونه که هستند گرفته نمی‌شود، اما وزن داده براساس معیارهای خاص تعیین شده توسط کاربر هستند. در این مورد، معیارهای تنظیم وزن‌ها براساس میانگین مربعات خطای ​​ریشه نرمال شده (RMSE) بنا به وقفه در داده‌های آزمون هستند. پایین‌ترین میانه و بالاترین وزن با شرط این‌که مجموع تمام وزن‌ها برابر با یک است. این مسئله به تنظیم اولویت نسبت به روش براساس عملکرد کمک می‌کند.

**4.3 گروه خطی براساس متوسط ​​وزن**

این گروه شبیه به گروه خطی براساس وزن میانه است، به جز این‌که متوسط ​​ مقدار RMSE تکنیک‌های فردی در داده‌های آزمون به‌جای مقدار میانگین، وزن‌های اختصاصی در نظر گرفته می‌شود.

**4.4 شبکه‌های عصبی مبتنی برگروه غیرخطی**

در اینجا، هیچ فرض در مورد ورودی داده شده به گروه وجود ندارد. مقادیر خروجی تکنیک‌های فردی به یک داور، که یک شبکه‌های عصبی پس انتشار (BPNN) است داده می‌شود.

**5. طراحی تجربی**

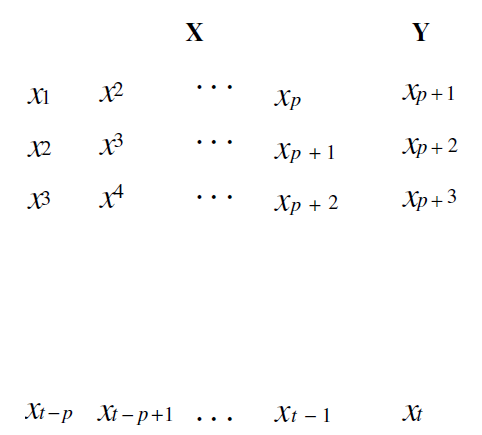
از آنجا که پیش‌بینی قابلیت اطمینان نرم‌افزار تنها یک متغیر وابسته است و متغیرهای توضیحی در معنای دقیق ندارد و از آنجایی که ما یک سری زمانی داریم، به‌دنبال مدل پیش‌بینی سری زمانی در انجام آزمایش‌ها هستیم که در فرم زیر نشانداده شده است (همانطور که در معادله 1 نشان داده شده است):



که در آن X بردار متغیرهای lag، است. ازاین‌رو کلید برای پیدا کردن راه‌حلی برای پیش‌بینی مشکلات تقریب تابع f است. این مسئله می‌تواند توسط تکرار تنظیم وزن در مدل‌سازی روند انجام شود.

یک تصویر از چگونگی آموزش الگو می‌تواند در فرایند مدل‌سازی شبکه‌های عصبی ارائه شده در شکل 3 طراحی شود (Xu و همکارانش (2003)). در این شکل،P نشان‌دهنده‌ی تعداد متغیرهای عقب مانده و(t-p) نشان‌دهنده‌ی تعداد کل آموزش نمونه است. در این نمایش،X مجموعه‌ای از بردارهای (t-p) P بُعدی وY یک بردار (t-p) بُعدی است. بنابراین، در مجموعه داده تبدیل شده، X وY نشان‌دهنده‌ی بردار متغیرهای توضیحی و متغیر وابسته هستند.

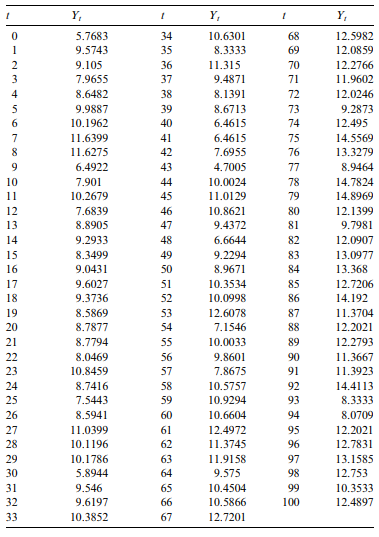
در این مقاله، نرم‌افزار خرابی داده، ارائه شده در جدول 1، توسط Musa (1979) به‌دست آمده است. که برای نشان دادن عملکرد پیش‌بینی از گروه پیشنهادی استفاده می‌شود.



شکل 3: طراحی داده‌های یادگیری

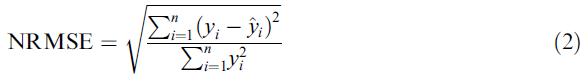
داده ها شامل 101 مشاهده از جفت (t، Yt) مربوط به خرابی نرم‌افزار است. در اینجا Yt نشان‌دهنده‌ی زمان شکست نرم‌افزار پس از اصلاح t ام ساخته شده است. SPSS 14.0 موجود در ( http: // [www.spss.com](http://www.spss.com)) برای پیدا کردن وقفه بهینه با توجه به داده‌های سری زمان استفاده شده است.

جدول1: داد‌ه‌های شکست نرم‌افزار



ما آزمایش‌های "تابع خودهمبستگی" و " تابع خودهمبستگی جزئی" را هامنند روش Box-Jenkins در سری‌های زمانی پیش‌بینی با استفاده از نرم‌افزار SPSS 14.0 در مجموعه داده‌ها انجام دادیم و دریافتیم که تاخیر 1 برای مجموعه داده‌ها کافی بود. با این حال، ما قادر به بررسی مقادیر NRMSE برای بهبود بیشتر برای وقفه بالاتر بودیم و تاخیر بالای 5 را بدست آوردیم. در بحث فوق در تولید مجموعه داده از سری‌های زمانی اصلی مانند این، پنج مجموعه داده مربوط به تاخیر 1، 2، 3، 4 و 5، به‌ترتیب بدست آوردیم.

از آنجا که داده‌های سری زمانی هستند، انجام 10 برابر اعتبارسنجی متقابل معنی ندارد، به‌عنوان مثال انتخاب نمونه‌ها به‌صورت تصادفی است و سپس جهت زمان داده پنهان و نادیده گرفته می‌شود. اعتبارسنجی متقابل در ارزیابی عملکرد یک مدل 10برابر قدرتمند و مفید است و باعث می‌شود کمتر با سری‌های زمانی و یا داده‌های سری سروکار داشته باشیم. از این رو، ما مجموعه داده را به‌ترتیب به 80٪ و 20٪ برای آموزش و تست جدا می‌کنیم. در حقیقت، این بررسی در بسیاری از داده‌های تجاری محبوب/ ابزارهای آماری گنجانده شده است. داده‌های آموزش داده شده، برای شناسایی پارامترهای بهینه برای مدلی که با توجه به معیارهای خطا و آن پارامترهایی که برای پیش‌بینی مقادیر در مجموعه آزمون استفاده می‌شود. مقدار میانگین مربعات خطای ریشه نرمال (NRMSE) به‌عنوان شاخص اندازه‌گیری مورد استفاده قرار گرفته است.



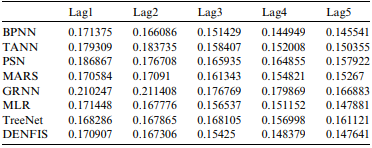
که در آن n تعداد مشاهدات پیش‌بینی است. Yi مقدار واقعی در دوره i و Yi مقدار پیش‌بینی قابلیت اطمینان نرم‌افزار در دوره i است.

**6. نتایج و بحث**

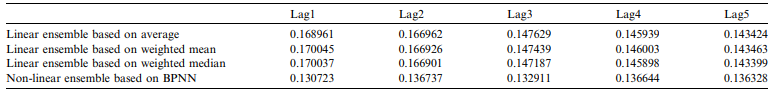
برای هر یک از روش‌ها، پارامترهای مناسب مشخص شده توسط الگوریتم، برای دریافت مطلوب‌ترین نتایج بهینه‌سازی می‌شود. جدول2 مقادیر مختلف NRMSE را از اطلاعات به‌دست آمده از تکنیک‌های مختلف نشان می‌دهد. پارامترهای بهینه‌سازی تا حداقل مقدار NRMSE با استفاده از معادله (2) محاسبه می‌شود و بهترین مقادیر در جدول 2 رای تاخیر داده شده ارائه شده است، نتایج آزمون به‌دست آمده از این تکنیک‌های فردی به گروه‌های مختلف ارائه شده است. مقادیر NRMSE به‌دست آمده از گروه برای تأخیر در جدول 3 ارائه شده است.

پارامترهایی که بر روی BPNN، TANN، PSN و GRNN بهترین نتایج را دارند در جدول 4-7 خلاصه شده است.

جدول2: مقدار RMSE در داده‌های آزمون برای روش‌های مختلف در حالت مستقل



جدول 3: مقدار RMSE بر روی داده‎های تست برای گروه‌های مختلف

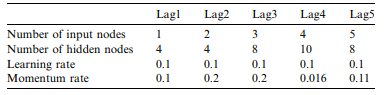


همچنین، پارامترهای گروه غیرخطی آموزش دیده با BPNN بهترین نتایج را در جدول 8 نشان می‌دهند. این ارزش‌ها توسط آزمون و خطا به‌دست آمده است. در انتخاب ترکیبات برای این گروه، عملکرد تکنیک‌های فردی بر روی همه‌ی وقفه‌ها (جداول 2 و 3) درنظر گرفته شده و بر اساس پنج تکنیک برتر BPNN، MLR، MARS، TreeNet و DENFIS برای تبدیل شدن به بخشی از گروهانتخاب می‌شوند. براین اساس، TANN، PSN و GRNN در گروه با توجه به عملکرد بد گنجانده نمی‌شوند.

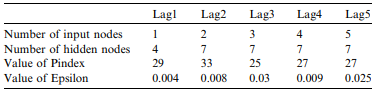
مشاهدات جالب را می‌توان از جداول 2 و 3 دریافت. ابتدا، به نظر می‌رسد ارتباطی بین شماره تاخیر و مقدار NRMSE مربوطه وجود دارد. ما متوجه شدم که با افزایش تاخیر مقدار NRMSE کاهش می‌یابد. ثانیا، برای وقفه‌های فردی، به نظر می‌رسید BPNN بهتر از تمام تکنیک‌های دیگر در حالت مستقل باشد، اگر چه روش‌های دیگر مانند MLR، DENFIS و MARS به‌طور مداوم بیش از همه وقفه‌ها انجام می‌شوند. ثالثا، گروه به نتایج بهتری از هر یک از تکنیک‌های منحصربه فرد با چند استثنا دست می‌یابد. به‌عنوان مثال، برای lag1، TreeNet بهتر از سه گروه خطی دیگر و برای lag2، BPNN عملکرد بهتری نسبت به گروه‌های خطی دارد. در نهایت، گروه غیرخطی ساخته شده با استفاده از BPNN به‌عنوان داور، عملکرد بهتری نسبت به سایر تکنیک‌های تشکیل‌دهنده در حالت مستقل و گروه خطی بیش از هر وقفه‌ای دارد. در گروه غیرخطی، کمترین مقدار NRMSE برای lag1 به دست آمده و همچنین تفاوت مقدار NRMSE از همه‌ی وقفه‌های دیگر بسیار کم است.

برای اصلاح بیشتر آن، داده‌های lag1 TreeNet حداقل ارزش NRMSE را برروی داده‌های آزمون دارد درحالی‌که BPNN، MARS ، DENFIS و MLR کاملا نزدیک هستند. برای داده‌های lag2، 3، 4 و 5، BPNN کمترین مقدار NRMSE را برای داده‌های آزمون به‌همراه داشت. DENFIS بهترین نتایج بعدی را به‌دست آورد. TANN برای داده‌های lags3، 4 و 5 عملکرد خوبی داشت. PSN اگرچه بهترین نتیجه را نداد ولی برای سرعت اجرایش مشخص شد. به‌جز برای داده‌های lag1 و lag2، که در آن برخی از تکنیک‌ها بهتر هستند، گروه خطی عملکرد بهتری از تکنیک‌های مستقل نشان داد.

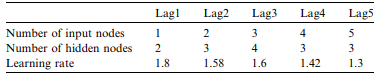
جدول 4: جزئیات ساختار BPNN و پارامترهای بیش از تأخیر



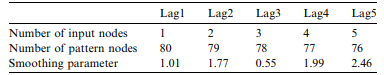
جدول 5: جزئیات ساختار TANN و پارامترهای بیش از تأخیر



جدول 6: جزئیات ساختار PSN و پارامترهای بیش از تأخیر



جدول 7: جزئیات ساختار GRNN و پارامترهای بیش از تأخیر



گروه غیرخطی بهتر از هر روش دیگر و یا گروهی برروی داده‌ها است. در میان گروه‌های خطی، میانگین وزنی و وزن متوسط گروه ​​براساس گروه مقدار RMSE مشابهی برای همه‌ی وقفه‌ها دارد.

در این ارتباط، مشاهده می‌کنیم که گروه وقت‌گیرتر از استفاده از روش هوشمند در حالت مستقل است. به‌علاوه، ما اشاره می‌کنیم زمانی‌که، پیش‌بینی قابلیت اطمینان به‌دقت در زمانی آفلاین ساخته شود، زمان هیچ محدودیت ندارد و گروه غیرخطی باید ترجیح داده شود. بااین‌حال، هنگامی که زمان محدودیت باشد، سپس روش بر خط مانند DENFIS باید به‌خاطر نیاز به تنها یک پاس یا یک تکرار برای پیش‌بینی ترجیح داده شود.

علاوهبراین، مشاهده می‌کنیم که Pai و Hong (2006) نیز از داده‌های مشابه‌ای برای آزمایش اثربخشی بردارپشتیبانی ماشین آلات و شبیه‌سازی روش (SVMSA) استفاده می‌کنند. بااین‌حال، از آنجایی که آنها را از داده‌های وقفه در آزمایش خود استفاده نمی‌کنند نتایج ما نمی‌تواند با آنها مقایسه شود. علاوه‌براین، آنها مجموعه داده‌های 101 مشاهده را به آموزش (33 مشاهده)، اعتبار‌سنجی (8 مشاهده) و آزمون (60 مشاهده) تقسیم کردند. از آنجا که تقسیم مجموعه داده برای آزمایش یک روش غیراستاندارد اس ، ما تصمیم گرفتیم تا نتایج خود را با آنها مقایسه کنیم. مقدار NRMSE به‌دست آمده در آزمون توسط آزمایش 0.1562 بود، که به‌خوبی نتایج حاصل از مدل ارائه شده نیست.

**7. نرم‌افزار مدل‌سازی ریسک عملیاتی در بانک ‌ها**

مدیریت ریسک یک عملکرد مهم در سازمان و حتی بیشتر در مورد بانک‌ها است. ازاین‌رو، نیاز فوری به مدیریت بهتر وجود دارد چون عواقب بسیار جدی برای تبدیل شدن بانک به ورشکسته وجود دارد. این امر می‌تواند برای گروه مدیریت خطر آسان‌تر باشد در صورتی که اطلاعات بیشتری در فرمت قابل استفاده در دست داشته باشند، که می‌تواند با یک توصیف مدل خطرات مرتبط، احتمال بروز آنها و تاثیر آنها بر وقوع بهبود یابد. به‌عنوان بخشی از کار ما در حال تلاش برای بدست آوردن مدلی هستیم که می‌تواند برای مدیریت ریسک در یک بانک مفید واقع شود. درسال‌های اخیر استفاده از نرم‌افزار در برنامه‌های بانکی به‌طور چشمگیری افزایش یافته است. بنابراین، داشتن نرم‌افزار قابل‌اعتماد برای کار موثر در بانک ها بسیار ضروری است. خطر شکست نرم‌افزار، با توجه به BIS (2001)، تحت ریسک عملیاتی برای بانک‌ها است.

ریسک عملیاتی به‌عنوان خطر از دست دادن و در نتیجه تعریف فرآیندهای داخلی ناکافی و یا شکست خورده، مردم و سیستم‌ها و یا وقایع خارجی (BIS، 2001) یاد می‌شود.

اگر چه، خطر ابتلا به نارسایی در سیستم‌ها شامل هر دو شکست سخت افزار و نرم‌افزار است، ما تنها خطر شکست نرم‌افزار را در اینجا بیان می‌کنیم. پیش‌بینی قابلیت اطمینان نرم‌افزار یک کار است که در آن ما سعی بر پیش‌بینی شکست آینده و هزینه بااستفاده از داده‌های شکست گذشته از نرم‌افزار داریم. از طریق پیش‌بینی قابلیت اطمینان نرم‌افزار ما قصد داریم بخشی نرم‌افزاری فن‌آوری عملیات خطر را پوشش دهیم. نتایج پیش‌بینی به تیم مدیریت خطر داده می‌شود تا اقدامات مناسب برای غلبه بر خطر انجام شود و در نتیجه به یک مدیریت ریسک بهتر منجرشود.

**8. نتیجه‌گیری**

در این مقاله، مدل گروه برای پیش‌بینی کارآمد قابلیت اطمینان نرم‌افزار توسعه یافت. سه گروه خطی و یک گروه غیرخطی برای پیش‌بینی قابلیت اطمینان نرم‌افزار توسعه یافته و مورد آزمایش قرار گرفت. تکنیک‌های آماری مختلف و هوشمند، گروه‌ها را تشکیل می‌دهند. آنها رگرسیون خطی چندگانه (MLR) و رگرسیون خطی چندمتغیره (MARS)؛ شبکه عصبی آموزش پس انتشار (BPNN)، سیستم استنتاج عصبی-فازی پویا (ANFIS) و TreeNet هستند. بر اساس آزمایش‌های عددی ما در قابلیت اطمینان نرم‌افزار که از اطلاعات کارهای قبلی بدست آمده متوجه شدیم که گروه غیرخطی عملکرد بهتری نسبت به همه گروه‌های دیگر و همچنین از روش‌های آماری و هوشمند دارد. علاوه‌براین، متوجه شدیم که گروه‌های خطی نیز عملکرد بهتری نسبت به تکنیک‌های تشکیل دهنده از lag3 به بعد دارند. که در نتیجه، گروه توسعه یافته می‌تواند به‌عنوان جایگزین مناسب برای روش‌های موجود در پیش‌بینی قابلیت اطمینان نرم‌افزار دارد.

**References**

Bai, C.G., Hu, Q.P., Xie, M., Ng, S.H., 2005. Software failure prediction based on Markov Bayesian network model. The Journal of Systems and Software 74 (3), 275–282.

Bates, J.M., Granger, C.W.J., 1969. The combination of forecasts. Operations research Quarterly 20, 451–468.

Benediktsson, J.A., Sveinsson, J.R., Ersoy, O.K., Swain, P.H., 1997. Parallel consensual neural networks. IEEE Transactions on Neural Networks 8, 54–64.

BIS, 2001, Basel Committee on Banking Supervision (2001), Consultative Document, The New Basel Capital Accord. <http://www.bis.org/>

Breiman, L., 1996. Bagging predictors. Machine Learning 24, 123–140. Cai, K.Y., Wen, C.Y., Zhang, M.L., 1991. A critical review on software reliability modeling. Reliability Engineering and System Safety 32 (3), 357–371.

Dueck, G., Scheuer, T., 1990. Threshold accepting: a general-purpose optimization algorithm appearing superior to simulated annealing. Journal of Computational Physics 90, 161–175.

Friedman, J.H., 1991. Multivariate adaptive regression splines (with discussion). Annals of Statistics 19, 1–141.

Friedman, J.H., 1999. Stochastic gradient boosting. Computational Statistics and Data Analysis 38 (4), 367–378.

Hansen, J., McDonald, J., Slice, J., 1992. Artificial intelligence and generalized qualitative response models: an empirical test on two audit decision-making domains. Decision Science 23, 708–723.

Ho, S.L., Xie, M., Goh, T.N., 2003. A study of connectionist models for software reliability prediction. Computers and Mathematics with Applications 46 (7), 1037–1045.

Kanmani, S., Uthariaraj, V.R., Sankaranarayanan, V., Thambidurai, P., 2007. Object-oriented software failure fault prediction using neural networks. Information and Software Technology 49, 483–492.

Karunanithi, N., Whitley, D., Maliya, Y.K., 1992. Prediction of software reliability using connectionist models. IEEE Transactions on Software Engineering 18, 563–574.

Kasabov, N.K., 2002. DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction. IEEE Transactions on fuzzy systems 10 (2).

Madsen, H., Thyregod, P., Burtschy, B., Albeanu, G., Popentiu, F., 2006. On using soft computing techniques in software reliability engineering. International Journal of Reliability, Quality and Safety Engineering 13 (1), 61–72.

Makridakis, S., Anderson, A., Carbone, R., Fildes, R., Hibdon, M., Lewandowski, R., Newton, J., Parzen, E., Winkler, R., 1982. The accuracy of extrapolation (time series) methods: results of a forecasting competition. Journal of Forecasting 1, 111–153.

Musa, J.D., 1979. Software reliability data. IEEE Computer Society – Repository.

Olmeda, I., Fernandez, E., 1997. Hybrid classifiers for financial multicriteria decision making: the case of bankruptcy prediction. Computational Economics 10, 317–335.

Pai, P.F., Hong, W.C., 2006. Software reliability forecasting by support vector machines with simulated vector machines with simulated annealing algorithms. The Journal of Systems and Software 79, 747– 755.

Pelikan, E., De Groot, C., Wurtz, D., 1992. Power consumption in WestBohemia: improved forecasts decorrelating connectionist networks. Neural Network World 2, 701–712.

Perrone, M.P., Cooper, L.N., 1993. When networks disagree: ensemble methods for hybrid neural netwoks. In: Mammone, R.J. (Ed.), Neural Networks for speech and Image processing. Chapman Hall, pp. 126– 142.

Ravi, V., Zimmermann, H.J., 2001. A neural network and fuzzy rule base hybrid for pattern classification. Soft Computing 5 (2), 152–159.

Ravi, V., Zimmermann, H.J., 2003. Optimization of neural networks via threshold accepting: a comparison with backpropagation algorithm. In: Conference on Neuro-Computing and Evolving Intelligence, Auckland, New Zealand.

Ravi, V., Murty, B.S.N., Dutt, N.V.K., 2005. Forecasting the properties of carboxylic acids by a threshold accepting-trained neural network. Indian Chemical Engineer 47 (3), 147–151.

Reformat, M., 2005. A fuzzy-based multimodel system for reasoning about the number of software defects. International Journal of Intelligent Systems 20 (11), 1093–1115.

Shin, Y., Ghosh, J., 1991. The Pi–Sigma networks: an efficient higherorder neural network for pattern classification and function approximation. In: Proceedings of International Joint Conference on Neural Networks, 1, Seattle, Washington, pp. 13–18.

Sitte, R., 1999. Comparison of software-reliability-growth predictions: neural networks vs parametric-recalibration. IEEE Transactions on Reliability 48 (3), 285–291.

Specht, D.F., 1991. A general regression neural network. IEEE Transactions on Neural Networks 2 (6), 568–576.

Su, Y.-S., Huang, C.-Y., 2006. Neural-network-based approaches for software reliability estimation using dynamic weighted combinational models. Journal of Systems and Software 80 (4), 606–615.

Tian, L., Noore, A., 2005a. On-line prediction of software reliability using an evolutionary connectionist model. The Journal of Systems and Software 77, 173–180.

Tian, L., Noore, A., 2005b. Evolutionary neural network modeling for software cumulative failure time prediction. Reliability Engineering and System Safety 87, 45–51.

Xu, K., Xie, M., Tang, L.C., Ho, S.L., 2003. Application of neural networks in forecasting engine systems reliability. Applied Soft Computing 2, 255–268.

Yu, L., Wang, S.Y., Lai, K.K., 2005. A novel non-linear ensemble forecasting model incorporating GLAR and ANN for foreign exchange rates. Computers and Operations Research 32 (10), 2523– 2541.

Zhou, Z-H., Wu, J., Tang, W., 2002. Ensembling neural networks: many could be better than all. Artificial Intelligence 137, 239–263.