بسم الله الرحمن الرحیم

پیش بینی سری زمانی به کمک

شبکه عصبی جردن

چکیده

در این مقاله قصد داریم پیش بینی یك سری زمانی را به کمك شبکه عصبی جردن شبیه سازی کرده و نتایج آن را با عملکرد شبکه‌های المان و پرسپترون چندلایه مقایسه کنیم.

کلمات کلیدی

شبکه عصبی، الگوریتم BP، پس‌انتشارخطا، MLP، المان، جردن.

مقدمه

سنگ بنای شبکه‌های عصبی به سال ۱۹۱۱ برمی‌گردد، زمانی که سگال[1]، بافت شناس و آسیب شناس اسپانیایی موفق به کشف نرون شد. حدود ۳۰ سال بعد از آن در دهه ی ۴۰ قرن بیستم بود که دیدگاه جدیدی در رابطه با شبکه‌های عصبی مطرح شد و وارن مك کلوث[2] و والتر پیتز[3] نشان دادند که شبکه‌های عصبی می‌توانند هر تابع حسابی و منطقی را محاسبه کنند. از آن زمان تاکنون شبکه‌های عصبی سرگذشت پر فراز و نشیبی را پشت سر گذاشته اند و حالا، در ابتدای قرن بیست و یکم، به یکی از بروزترین و پرکاربردترین مباحث دانش بشری تبدیل شده اند. امروزه این شبکه‌ها در مسائل کنترلی، مدلسازی، شبیه سازی، بهینه‌سازی، طبقه بندی، شناسایی و تشخیص الگو، پردازش سیگنال، پیش بینی سری‌های زمانی و. . . استفاده گسترده ای پیدا کرده اند. در این نوشتار قطره ای از این دریای بیکران را برای شما به ارمغان می‌آوریم، باشد که مقبول تشنگان راه دانش افتد.

در فصل اول این تحقیق، ابتدا به آشنایی مختصری با شبکه‌های عصبی و روش‌های آموزش آنها می‌پردازیم، در فصل دوم ساده ترین نوع شبکه عصبی پیشخور را معرفی می‌کنیم و به خصوص روش پس‌انتشارخطا که یکی از

- Santiago Ramón y Cajal[1]
- Warren McCulloch[2]
- Walter pittz[3]

پرکاربردترین ابزارهای آموزش شبکه‌های باناظر است را توضیح خواهیم داد.

در فصل سوم به بررسی چند شبکه عصبی برگشتی معروف خواهیم پرداخت و الگوریتم‌های آموزش آنها به وسیله روش پس‌انتشارخطارا معرفی خواهیم کرد. در فصل چهارم این شبکه‌ها را به کمک نرم‌افزار مطلب در قالب مثالی برای پیش‌بینی سری‌زمانی، شبیه سازی خواهیم کرد و در پایان نتایج شبیه سازی‌ها را با یکدیگر مقایسه میکنیم.

فهرست

چکیده ۳

کلماتکلیدی ۳

مقدمه ٤

فصل ۱ آشناییباشبکه‌هایعصبی ۱۰

۱-۱ مقدمه ۱۰

۱-۲ شبکه‌هایعصبی ۱۰

۱-۲-۱ قابلیتیادگیری ۱۱

۱-۲-۲ پراکندگیاطلاعات ۱۲

۱-۲-۳ قابلیتتعمیم ۱۲

۱-۲-٤ پردازشموازی ۱۳

۱-۲-٥ مقاومبودن ۱۳

۱-۳ شبکهعصبیازدیدکاربر ۱۳

۱-٤ شبکهعصبیبیولوژیکی ۱٤

۱-٥ فرایندیادگیری ۱۸

۱-٥-۱ یادگیریباناظر ۱۹

۱-۵-۲ یادگیریبدوناظر ۲۰
۱-۵-۳ یادگیریتشدیدی ۲۱
فصل ۲ شبکهعصبیMLP ۲۲
۲-۱ مدلریاضینرون ۲۲
۲-۲ توابعمحرك ۲٤
۲-۲-۱ تابعمحرکخطی: ۲۵
۲-۲-۲ تابعمحرکتانژانت هیپربولیك: ۲٦
۲-۳ مدلنرونچندورودی ۲۷
۲-٤ فرمخلاصهشدهنرونچندورودی ۲۸
۲-۵ ساختارشبکههایعصبی ۲۹
۲-٦ شبکهعصبیتكلایه ۲۹
۲-۷ شبکههایعصبیچندلایه ۳۰
۲-۸ مزیتشبکههایعصبیچندلایه ۳۱
۲-۹ شبکهعصبیMLP ۳۳
۲-۱۰ الگوریتمپسانتشارخطا ۳۵
۲-۱۰-۱ انتشار ۳٦
۲-۱۰-۲ بروزرسانیوزنها ۳٦

۲- ۱۱ فرمولبندیالگوریتمپسانتشارخطابراییکشبکهMLP ۳۷

۲- ۱۲ محدودیتهایالگوریتمپسانتشارخطا ۳۹

۲-۱۳ خلاصه ۳۹

فصل ۲ شبکههایبرگشتیالمانوجردن ۴۰

۳-۱ مقدمه ۴۰

۳-۲ شبکهعصبیالمان ۴۱

۳-۳ فرمولبندیالگوریتمپسانتشارخطابرایشبکهعصبیالمان ۴۱

۳- ٤ شبکهعصبیجردن ٤۳

۳-٥ فرمولبندیالگوریتمپسانتشارخطابرایشبکهعصبیجردن ٤۳

فصل ٤ نتایجبدستآمدهازشبیهسازیشبکههایعصبی ٤٥

٤-۱ پیشبینیسریزمانیبهکمکشبکهعصبیMLP ٤٦

٤-۲ پیشبینیسریزمانیبهکمکشبکهعصبیالمان ٥۳

٤-۳ پیشبینیسریزمانیبهکمکشبکهعصبیجردن ٦۰

٤-٤ مقایسهنتایجبدستآمدهازشبیهسازیشبکهها ٦۷

٤-٤-۱ انواعخطاهایاندازهگیریشده ٦۷

٤-٤-۲ نتایجشبیهسازی ٦۸

فصل ٥ نتیجهگیریوپیشنهادات ٦۹

پیوست ۱ - کد تولید کننده سری زمانی ۷۱

پیوست ۲ - کد تشخیص خطا ۷۲

پیوست ۳- تأثیر تغییرات در پارامترهای طراحی شبکه ۷۳

مراجع ۷۸

فصل ۱

آشنایی با شبکه‌های عصبی

۱-۱ مقدمه

چرا شبکه‌های عصبی؟ چه مزایایی در این شبکه‌ها نسبت به سایر روش‌های حل وجود دارد که آنها را چنین پرکاربرد کرده است؟ اساساً دلیل استفاده از این شبکه‌ها چیست و آنها چه کارهایی میتوانند انجام دهند؟ همگی اینها سوالاتی است که در این بخش و در کل این تحقیق سعی میکنیم جوابی برای آنها پیدا کنیم، پیش از هر چیز اجازه دهید اندکی راجع به این شبکه‌ها توضیح دهیم.

۱-۲ شبکه‌های عصبی[٤]

بلوك‌های ساختاری در هر دو شبکه مصنوعی و بیولوژیکی دستگاه‌های محاسباتی خیلی ساده ای هستند و علاوه بر این نرون‌های مصنوعی از سادگی بیشتری برخوردار می‌باشند، در این شبکه‌ها ارتباط بین نرون[٥]ها عملکرد شبکه را تعیین می‌کند. شبکه‌های عصبی مصنوعی با وجود اینکه با سیستم عصبی طبیعی قابل مقایسه

- neural network[٤]
- neuron[٥]

نیستند ویژگی‌هایی دارند که آنها را در بعضی از کاربردها مانند تفکیک الگو، رباتیک، کنترل و بطور کلی در هرجا که نیاز به یادگیری یك نگاشت خطی و یا غیرخطی باشد، ممتاز می‌نماید. در ادامه به این ویژگی‌هااشاره می‌کنیم.

۱-۲-۱ قابلیت یادگیری

استخراج نتایج تحلیلی از یك نگاشت غیرخطی که با چند مثال مشخص شده، کار ساده ای نیست. پیاده سازی راه حلی برای چنین موضوعی با یك الگوریتم معمولی و بدون قابلیت یادگیری نیاز به دقت و مراقبت زیادی دارد. در چنین حالتی سیستمی که خود بتواند این رابطه را استخراج کند بسیار سودمند است. خصوصاً افزودن مثال‌های احتمالی در آینده به یك سیستم با قابلیت یادگیری بمراتب آسانتر از انجام آن در یك سیستم بدون چنین قابلیتی است، چرا که در سیستم اخیر افزودن یك مثال جدید به منزله تعویض کلیه کارهای انجام شده قبلی است.

قابلیت یادگیری یعنی توانایی تنظیم پارامترهای شبکه در مسیر زمان که محیط شبکه تغییر می‌کند و شبکه شرایط جدید را تجربه می‌کند. با این هدف که اگر شبکه برای یك وضعیت خاص آموزش دیده و تغییر کوچکی در شرایط محیطی آن رخ داد، شبکه بتواند با آموزش مختصر برای شرایط جدید نیزکارآمد باشد.

۲-۲-۱ پراکندگی اطلاعات

آنچه که شبکه فرا میگیرد در وزن‌های سیناپسی مستتر است. رابطه یك به یك بین ورودی‌ها و وزن‌های سیناپتیکی وجود ندارد، میتوان گفت که هر وزن سیناپسی مربوط به همه ورودی‌ها است ولی به هیچکدام از آنها بطور منفرد و مجزا مربوط نیست. به عبارت دیگر فعالیت هر نرون در شبکه، از فعالیت سایر نرون‌هامتأثرمی‌باشد. در نتیجه اطلاعات بصورت مفهومی توسط شبکه‌های عصبی پردازش می‌شود. بر این اساس چنانچه بخشی از سلول‌های شبکه حذف شوند و یا عملکرد غلط داشته باشند باز هم احتمال رسیدن به پاسخ صحیح وجود دارد،اگرچه این احتمال برای تمام ورودی‌ها کاهش یافته ولی برای هیچ یك از بین نرفته است.

۳-۲-۱ قابلیت تعمیم

پس از آنکه مثال‌های اولیه به شبکه آموزش داده شد، شبکه میتواند در مقابل یك ورودی آموزش داده نشده قرارگیرد و یك خروجی مناسب ارائه نماید. این خروجی بر اساس مکانیسم تعمیم، که چیزی جز فرآیند درونیابی نیست به دست می‌آید. به عبارت روشنتر، شبکه تابع را فرا میگیرد، الگوریتم را می‌آموزد و یا رابطه تحلیلی مناسبی برای تعدادی نقاط در فضا به دست می‌آورد.

۱-۲-٤ پردازش موازی[٦]

پردازش موازی اصلی ترین برتری شبکه عصبی بیولوژیکی بر شبکه عصبی مصنوعی است، هنگامی که شبکه عصبی در قالب سخت افزار پیاده‌سازیمی‌شود سلول‌هایی که در یك تراز قرار میگیرند میتوانند بطور همزمان به ورودی‌های آن تراز پاسخ دهند. این ویژگی باعث افزایش سرعت پردازش می‌شود.

۱-۲-٥ مقاوم بودن

در یك شبکه عصبی هر سلول بطور مستقل عمل می‌کند و رفتار کلی شبکه برایند رفتارهای محلی سلول‌های متعدد است. این ویژگی باعث می‌شود تا خطاهای محلی از چشم خروجی نهایی دور بمانند. به عبارت دیگر، سلول‌ها در یك روند همکاری، خطاهای محلی یکدیگر را تصحیح می‌کنند. این خصوصیت باعث افزایش مقاومت در سیستم شده و تحمل پذیری خطا را افزایش میدهد.

۱-۳ شبکه عصبی از دید کاربر

یکی از محاسن بزرگ شبکه‌های عصبی، سادگی کار با آنها برای کاربران عادی است. پس از فرایند طراحی شبکه، کاربر میتواند بطور کاملاً ماژولار با سیستم برخورد کند و احتیاج به دانشی در مورد شبکه‌های عصبی ندارد. به عبارت دیگر شبکه مانند یك جعبه

---

[٦] - Parallel processing

سیاه[۷]است که راه حل مسئله شما را درون خود دارد و شما با دادن ورودی مناسب، خروجی مناسب خود را دریافت خواهید کرد.

۱-٤ شبکه عصبی بیولوژیکی[۸]

مغز به عنوان یك سیستم پردازش اطلاعات با ساختاری موازی، از ۱۰۰ تریلون (10^{11}) نرون به هم مرتبط با تعداد 10^{16} ارتباط تشکیل شده است. نرون‌ها ساده ترین واحد ساختار سیستم عصبی هستند. بافت‌هایی که عصب نامیده می‌شوند اجتماعی از نرون‌ها هستند که اطلاعات و پیام‌ها را از یك قسمت بدن به قسمت دیگر منتقل می‌کنند. این پیام‌ها از نوع ایمپالسهای الکتروشیمیایی هستند.

میلیون‌ها نرون در بدن انسان وجود دارند، حتی ساده ترین کارهای روزمره انسان مانند پلك زدن، تنها از طریق همکاری همه جانبه این نرون‌ها میسر است. بیشترین تعداد نرون‌ها در مغز و باقی در نخاع و سیستم‌های عصبی جانبی تمرکز یافته اند. اگرچه همه نرون‌ها کارکرد یکسانی دارند، ولی اندازه و شکل آنها بستگی به محل استقرارشان در سیستم عصبی دارد. بیشتر نرون‌ها از سه قسمت اساسی تشکیل شده اند.

۱- بدنه سلول (که شامل هسته و قسمت‌های حفاظتی دیگر می‌باشد)

- Black box[۷]
- Biological neural network[۸]

۲- دندریت[۹]

۳- اکسون[۱۰]

که دو قسمت آخر عناصر ارتباطی نرون را تشکیل می‌دهند. شکل‌های ۱-۱ و ۱-۲ ساختمان سلول عصبی را نشان می‌دهند. نرون‌ها بر اساس ساختارهایی که بین آنها پیام‌ها هدایت می‌شوند به سه دسته تقسیم می‌گردند:

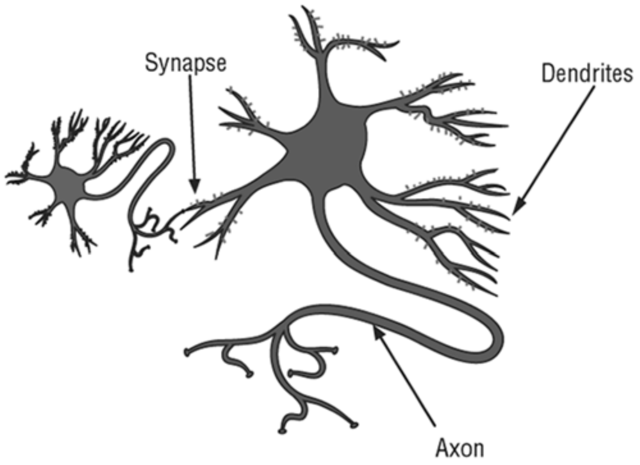
نرون‌های حسی: که اطلاعات را از ارگان‌های حسی به مغز و نخاع می‌فرستند.

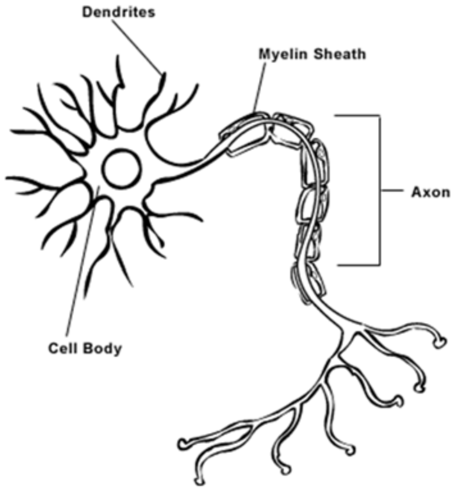
نرون‌های محرك: که سیگنال‌های فرمان را از مغز و نخاع به ماهیچه‌ها و غدد هدایت می‌کنند.

نرون‌های ارتباطی: که نرون‌ها را به هم متصل می‌کنند.

---

[۹] - dendrite
[۱۰] - axon





شکل ۱- ۱و ۱-۲ نواحی اصلی یک سلول عصبی بیولوژیک

فعالیت دو نرون اول (حسی و محرک) توسط نرون‌های ارتباطی به هم مربوط می‌شوند. روابط بین نرون‌های ارتباطی موجبات انجام کارهای پیچیده روزمره، از قبیل تفکر، احساسات، ادراک و محفوظات را فراهم می‌آورد. در انسان تعداد این نرون‌ها خیلی بیشتر از تعداد نرون‌های حسی و محرک می‌باشد. تخمین زده می‌شود که برای هر نرون محرک بیش از ۴۰۰۰ نرون ارتباطی موجود است.

دندریت‌ها به عنوان مناطق دریافت سیگنال‌های الکتریکی، شبکه‌هایی تشکیل یافته از فیبرهای سلولی هستند که دارای سطح نامنظم و شاخه‌های انشعابی بیشمار می‌باشند، به همین علت به آنها شبکه‌های دریافتی «درخت گونه»می‌گویند. دندریت‌هاسیگنال‌های الکتریکی را به هسته سلول منتقل می‌کنند. بدنه سلول، انرژی لازم را برای فعالیت نرون فراهم نموده و بر روی سیگنال‌های دریافتی عمل می‌کند، که با یك «عمل ساده جمع و مقایسه با یك سطح آستانه» مدل میگردد. اکسون بر خلاف دندریت‌ها از سطحی هموارتر و تعداد شاخه‌های کمتر برخوردار است. اکسون طول بیشتری دارد و سیگنال الکتروشیمیایی دریافتی از هسته سلول را به نرون‌های دیگر منتقل می‌کند. محل تلاقی یك اکسون از یك سلول به دندریت‌های سلول دیگر را سیناپس[11]می‌گویند. سیناپس‌ها واحدهای ساختاری کوچکی هستند که ارتباط بین نرون‌ها را برقرار می‌سازند. لازم به ذکر است که پیام‌های عصبی تنها بصورت یکطرفه حرکت می‌کنند: از دندریت‌ها به بدنه سلول و سپس به اکسون. هنگامی که پیام به انتهای اکسون میرسد، فیبرهای ماهیچه‌ای، سلول‌های غددی و نرون‌های دیگری که به اکسون متصل هستند را تحریك می‌کند. بطور خلاصه آنچه که مهم است، ترتیب نرون‌ها و شدت سیناپس‌های بین هر دو نرون می‌باشد، که توسط یك فرایند شیمیایی خیلی پیچیده به دست می‌آید. بعضی ساختارهای عصبی هنگام تولد ایجاد می‌شوند و

---

[11] - synapse

قسمت‌های دیگر، در خلال یادگیری به عنوان ارتباط‌های جدید به وجود می‌آیند و یا از بین می‌روند.

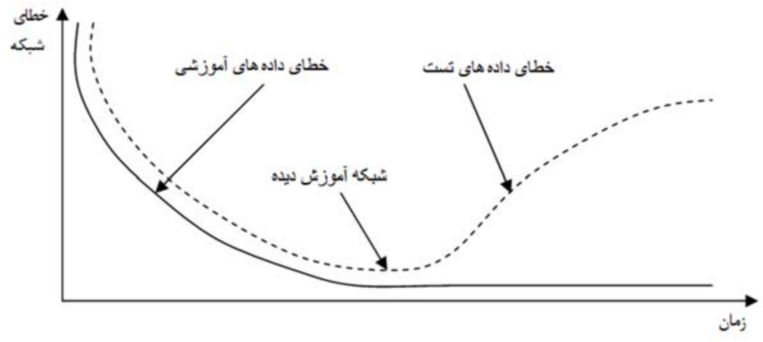
۱-۵ فرایند یادگیری[۱۲]

سیستم‌های یادگیر سیستم‌هایی هستند که با مشاهده عملکردشان، میتوانند رفتارشان را در جهت دستیابی به هدف و مقصد خاصی بهبود بخشند. اگر مقاصد و اهداف بطور کامل تعریف شده باشند، آنگاه دیگر به فرآیند یادگیری احتیاجی نیست. زمانی به پروسه یادگیری نیاز است که اطلاعات کامل در مورد اهداف موجود نباشد، جایی که میدانیم به علت عدم قطعیت در شرایط محیطی، سیستمی که دارای خواص و پارامترهای ثابت باشد نمیتواند بطور کامل عمل کند. رفتار سیستم‌های یادگیر توسط الگوریتم‌های بازگشتی بیان می‌شود. به همین خاطر به این الگوریتم‌ها قوانین یادگیری می‌گویند و عموماً توسط معادلات تفاضلی (دیفرانسیلی) بیان می‌شوند. این الگوریتم‌ها روی اطلاعات موجود چنان پردازش می‌کنند که شاخص اجرایی مشخص شده ای که عموماً تقریبی است از هدف خاصی که مقصود پروسه یادگیری می‌باشد، بهینه گردد و اینکار تنها راه جبران نمودن نقصان اطلاعات اولیه است. میدانیم که تجربه‌ها در مسیر زمان حاصل می‌شوند و به عبارت دیگر هیچکس آینده خود را تجربه نکرده است. میزان یادگیری ما به درجه کامل بودن اطلاعات قبلی ما بستگی دارد. در

---

[۱۲] - Learning Process

حالت کلی سه نوع یادگیری موجود است: یادگیری باناظر، یادگیری بدون ناظر و یادگیری تشدیدی(تقویتی). بطور کلی در فرایند یادگیری سه مورد باید به ترتیب انجام شوند، اول سیستم یادگیرنده توسط محیط تحریک شود، سپس قانون یادگیری با رجوع به نتیجه تحریک، پارامترهای سیستم یادگیر را تغییر دهد و در انتها سیستم یادگیرنده به خاطر تغییراتی که در ساختار داخلی آن اتفاق افتاده است، پاسخ مناسبتری به محیط بدهد. یکی از مسائلی که باید مورد توجه قرار بگیرد آموزش ندادن بیش از حد شبکه است، این مسئله میتواند باعث عملکرد ضعیف شبکه در محیط‌های نویزی شود، این مسئله در شکل ۱-۳ نشان داده شده است.



شکل ۱-۳ خطای ناشی از آموزش بیش از حد شبکه

۱-۵-۱ یادگیری باناظر[۱۳]

در یادگیری باناظر، به قانون یادگیری مجموعه‌ای از زوج‌های داده به نام «داده‌های یادگیری» $\{(p^i, t^i)\ i = 1,2, \dots, l\}$ می‌دهند، که در آن p^i ورودی به شبکه و t^i خروجی

---

۱۳ - Supervised learning

مطلوب شبکه برای ورودی p^i است. پس از اعمال ورودی p^i به شبکه عصبی، در خروجی شبکه a^i با t^i مقایسه شده و سپس خطای یادگیری محاسبه، و از آن جهت تنظیم پارامترهای شبکه استفاده می‌شود. به گونه ای که اگر دفعه بعد به شبکه همان ورودی p^i اعمال شود، خروجی به t^i نزدیکتر می‌شود. میزان نزدیکی عموماً توسط نرم دوم بردارها سنجیده می‌شود. یادگیری باناظر بدین مفهوم است که ما t^i های مناسب با p^i ها را داریم، در واقع معلم سیستمی است که بر محیط واقف است و این معلم است که پاسخ‌های مطلوب را برای آموزش شبکه در اختیار آن میگذارد.

پارامترهای شبکه عصبی توسط دو سیگنال ورودی و خطا تنظیم می‌شوند. بدین صورت که پس از چند تکرار الگوریتم یادگیری، پارامترهای شبکه به پارامترهای فضای یادگیری (معلم) همگرا می‌شوند، که برای آنها خطای یادگیری بسیار کوچك است و عملاً شبکه عصبی معادل معلم می‌شود. به عبارت دیگر، اطلاعات مربوط به محیط (نگاشت بین t^i و p^i) که برای معلم روشن است به شبکه عصبی منتقل می‌شود.

۲-۵-۱ یادگیری بدون ناظر[۱۴]

در یادگیری بدون ناظر یا یادگیری خود سازمانده، پارامترهای شبکه عصبی تنها توسط پاسخ سیستم اصلاح و تنظیم می‌شوند. به عبارتی تنها اطلاعات دریافتی از محیط به شبکه را بردارها ورودی p^i تشکیل می‌دهند. در واقع به شبکه عصبی هیچ نمونه‌ای از

---

[۱۴] - Unsupervised learning

تابعی که قرار است بیاموزد داده نمی‌شود. در نگاه اول به نظر می‌رسد یادگیری بدون ناظر بدون فایده و غیرممکن باشد. با طرح این سوال که "چگونه میتوانیم یک سیستم یادگیرنده را آموزش دهیم اگر ندانیم که این سیستم چه کاری قرار است انجام دهد؟" کاربرد اصلی این الگوریتم‌ها برای عمل خوشه‌بندی است. شبکه با این الگوریتم‌هامی‌آموزد که الگوهای ورودی را به تعداد متناهی از گروه‌ها تقسیم بندی کند. باید توجه داشت که در این حالت این فرد طراح یا معلم است که هدف و مقصد نهایی، یعنی چیزی که باید به آن رسید را مشخص می‌کند. به عبارت دیگر یادگیری بدون معلم مفهوم نادرستی است که عموماً به جای یادگیری بدون ناظر مورد استفاده قرار میگیرد.

۳-۵-۱ یادگیری تشدیدی[۱۵]

یادگیری تشدیدی یا تقویتی یکی از پرکاربردترین و جالبترین انواع یادگیری شبکه‌ها عصبی است. این شبکه‌هااصولاً به صورت برخط[۱۶] مورد استفاده قرار میگیرند و بیشترین کاربرد را در مسائل کنترلی دارند. در این حالت یادگیری در واقع از یک نگاشت ورودی-خروجی استفاده می‌کند. به عبارت دیگرp^iها وt^iهابطور مشخص و مجزا در دسترس نیستند. بلکه ورودی از طریق یک رابط بین دستگاه و محیط (یک سنسور) فراهم می‌شود. این کار از طریق یک پروسه سعی و خطا به صورتی انجام می‌پذیرد که

---

[۱۵] - Reinforcement learning
[۱۶] - online

یك شاخص اجرایی موسوم به سیگنال تشدید ماکزیمم شود. در واقع به جای فراهم نمودن جواب واقعی، به شبکه عددی که نشانگر میزان عملکرد شبکه است ارائه می‌شود. این بدین معنی است که اگر شبکه عصبی پارامترهایش را به گونه‌ای تغییر داد که منجر به یك حالت مساعد شد، آنگاه تمایل سیستم یادگیر جهت تولید آن عمل خاص تقویت یا تشدید می‌شود. در غیر این صورت تمایل شبکه عصبی جهت تولید آن عمل خاص تضعیف می‌شود.

فصل ۲

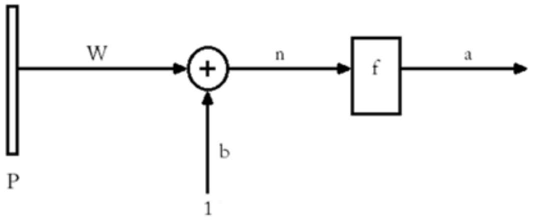
MLP شبکه عصبی

۱-۲ مدل ریاضی نرون

پس از آشنایی با نرون‌ها و شبکه عصبی بیولوژیکی نوبت به مدل کردن آنها می‌رسد، در این بخش مدل ساده ای از یك نرون واقعی را بررسی میکنیم، خواهیم دید که شبکه‌های عصبی مصنوعی[۱۷] از ترکیب و کنار هم گذاشتن تعداد زیادی از این نرون‌ها به وجود می‌آیند. به عبارت دیگر نرون کوچکترین واحد پردازشگر اطلاعات است که اساس عملکرد شبکه‌های عصبی را تشکیل می‌دهد.

- Artificial neural network[۱۷]

شکل ۲-۱ ساختار یك نرون تك ورودی را نشان می‌دهد. اسکالرهای P و aبه ترتیب ورودی و خروجیمی‌باشند.



شکل۲- ۱مدل نرون تك ورودی

میزان تاثر p روی a به وسیله مقدار اسکالر w تعیین می‌شود. ورودی دیگر که مقدار ثابت ۱ است در جمله بایاس b ضرب شده و سپس با wp جمع می‌شود. این حاصل جمع، ورودی خالص n برای تابع تحریك f خواهد بود. بدین ترتیب خروجی نرون با معادله ۲-۱ تعریف می‌شود:

$$a = f(wp + b) \quad (۲-۱)$$

در مقایسه این نرون تك ورودی با یك نرون بیولوژیکی، w معادل شدت سیناپس و مجموعه جمع کننده، تابع محرك معادل هسته سلول و a معادل سیگنال گذرنده از اکسون خواهد بود. نکته ای که باید به آن توجه شود اهمیت و تأثیر جمله بایاس b است. این جمله را میتوان مانند وزن W در نظر گرفت، با این تصور که میزان تأثیرورودی ثابت ۱ را روی نرون منعکس میسازد. تابع محرك f توسط طراح انتخاب می‌شود،بر اساس انتخاب f و نوع الگوریتم یادگیری، پارامترهای W و b تنظیم

می‌شوند. به عبارت دیگر یادگیری بدین معنی است که پارامترهای W و bطوری تنظیم شوند که رابطه ورودی و خروجی نرون با هدف خاصی مطابقت نماید.

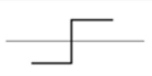
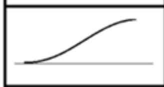
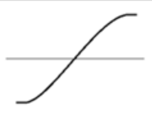
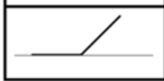
۲-۲ توابع محرك[18]

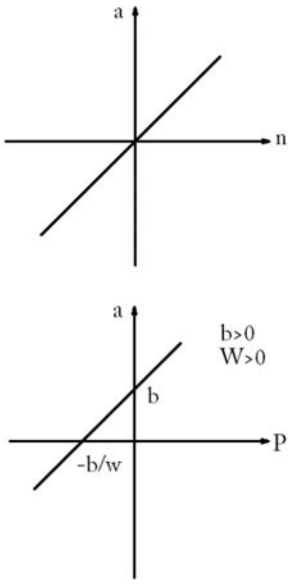
تابع محرك fمی‌تواند خطی یا غیرخطی باشد. یك تابع محرك بر اساس نیاز خاص حل یك مسئله –مسئله‌ای که قرار است به وسیله شبکه عصبی حل شود- انتخاب می‌شود. در عمل تعداد محدودی از توابع محرك مورد استفاده قرار می‌گیرند که مهمترین آنها در جدول ۲-۱لیست شده است. در اینجا به دو مورد از آنها که در این تحقیق مورد استفاده ما هستند اشاره می‌کنیم.

---

[18] - activation functions

جدول۲- ۱ انواعتوابعمحرك

	نام تابع	تعریف تابع	نمودار
1	آستانه ای دو مقداری	a=0, n<0 a=1, n≥0	
2	آستانه ای دو مقداری متقارن	a=-1, n<0 a= 1, n≥0	
3	خطی	a=n	
4	آستانه ای خطی متقارن	a=-1, n<-1 a= n, -1≤n≤1 a= 1, n>1	
5	آستانه ای خطی	a=0, n<0 a=n, 0≤n≤1 a=1, n>1	
6	زیگموئیدی	$a=\frac{1}{1+e^{n}}$	
7	تانژانت هیپربولیک	$a=\frac{e^{n}-e^{n}}{e^{n}+e^{n}}$	
8	خطی مثبت	a=0, n<0 a=n, n≥0	



شکل۲- ۲تابع محرك خطی و تاثیر جمله

بایاس

۱-۲-۲ تابع محرك خطی:

خروجی این تابع برابر ورودی آن است:

$$a = f(n) \tag{۲-۲}$$

اهمیت جمله بایاس b را در شکل ۲-۲می‌بینیم. اگر پاسخ نرون بر حسب ورودی رسم شده باشد، جمله بایاس b موجب جابجایی منحنی در فضای ورودی می‌گردد و به عبارتی موجب می‌گردد که نرون به زیر فضایی از فضای ورودی بایاس منتقل گردد، که خود انتخاب کلمه بایاس را برای ترم b توجیه می‌کند.

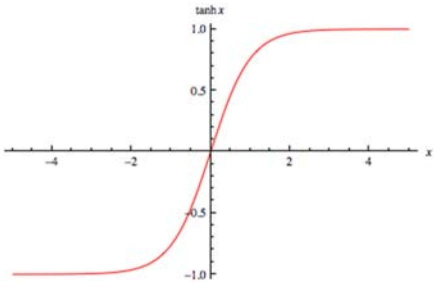
۲-۲-۲ تابع محرک تانژانت هیپربولیک:

این تابع با فرمول کلی زیر بیان می‌شود:

$$a = \tanh(n) = \frac{e^{cn} - e^{-cn}}{e^{cn} + e^{-cn}} \qquad (۲-۳)$$

شکل این تابع به ازاء $C = 1$ در تصویر ۲-۳ رسم شده است. مقدار C وسعت ناحیه خطی بودن تابع را تعیین می‌کند. مثلاً اگر C خیلی بزرگ باشد، شکل منحنی به تابع محرک آستانه ای دومقداری متقارن نزدیکتر می‌شود. دو مسئله مهم باعث محبوبیت این تابع تحریک در شبکه های عصبی چندلایه شده است، که یکی غیر خطی بودن و دیگری مشتق‌پذیر بودنآن است، مخصوصاً اینکه می‌توان مشتقات این تابع را بر حسب خود تابع نوشت.

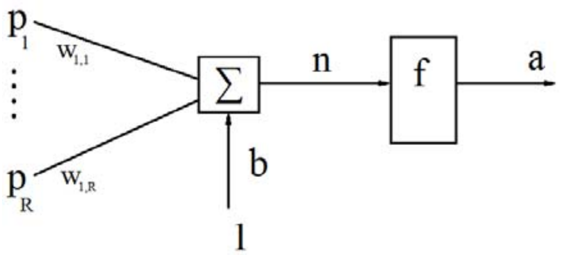
غیر خطی بودن تابع تحریک باعث می‌شود شبکه عصبی بتواند توابع غیرخطی را مدلسازی کند و مشتق‌پذیر بودن آن پیشنیاز لازم برای استفاده از الگوریتم پس‌انتشارخطا است که این مسئله را در بخش ۲-۱۱ به شکل دقیق‌تری مشاهده خواهید فرمود.



شکل۲- ۳تابع محرك تانژانت هیپربولیك

۲-۳ مدل نرون چند ورودی

عموماً یك نرون بیش از یك ورودی دارد. شکل ۲-٤ یك مدل نرون با R ورودی را ارائه میدهد. بردار ورودی با Pنمایش داده می‌شود. اسکالرهای p_i $(i=1,2,…,R)$ عناصر بردارP هستند. مجموعه سیناپس‌های $w_{1,j}$، عناصر ماتریس وزن را تشکیل می‌دهند. دراین حالتW یك بردارسطری با عناصر$w_{1,j}$ $(j=1,2,…,R)$ است. هر عنصر از بردار ورودیP در عنصر متناظر ازWضرب می‌شود. نرون، یك جمله بایاسbدارد که با حاصل ضرب ماتریس وزنWبا بردار ورودیP جمع می‌شود.



شکل۲- ٤مدل چند ورودی یك نرون

ورودی خالص n مطابق فرمول زیر محاسبه می‌شود:

$$n = \sum_{i=1}^{R} p_i w_i + b = WP + b$$ (۲- ٤)

که در آن:

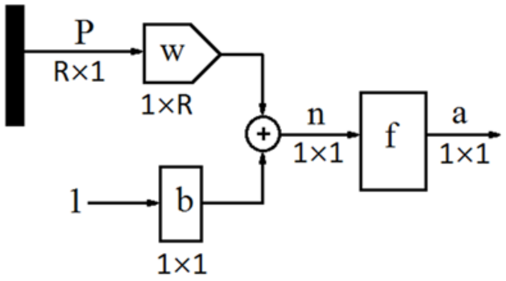
$$P = [p_1, p_2, \dots, p_R]^T, W = [w_{1,1}, \dots, w_{1,R}]$$ (۲- ٥)

در نهایت، خروجی نرون به صورت زیر خواهد بود:

$$a = f(WP + b)$$ (۲- ٦)

٤-۲ فرم خلاصه شده نرون چند ورودی

یك مدل خلاصه شده نرون چند ورودی را می‌توان به شکل ۲-٥ نشان داد.



شکل۲- ٥فرم ساده شده نرون با R ورودی

همانگونه که در شکل ۲- ٥ مشخص است، بردار ورودیP با یك ستون عمودی در سمت چپ نمایش داده می‌شود. ابعادP در زیر متغیر P با $R\times1$مشخص شده است. این نماد نشان‌دهنده آن است که بردار ورودیPدارای R عنصر است. بردارP در یك ماتریسW باR ستون، ضرب می‌شود. مقدار ثابت ۱ به عنوان یك ورودی به نرون اعمال شده و در جمله اسکالر بایاسb ضرب می‌شود. ورودی خالص n، به تابع تبدیل

f اعمال شده و خروجی مربوطه به وجود می‌آید. aعملاً نشان دهنده خروجی شبکه تك نرونی است ودر این حالت یك اسکالر با ابعاد ۱×۱ است.

۲-۵ ساختار شبکه‌های عصبی

باید توجه داشت که معمولاً نرونی با ورودی‌های زیاد نیز، به تنهایی برای حل مسائل فنی- مهندسی کفایت نمی‌کند. مثلاً برای مدلسازی نگاشت‌هایی که دارای دو خروجی هستند احتیاج به دو نرون داریم، که بطور موازی عمل کنند. در این حالت یك لایه خواهیم داشت که از اجتماع چند نرون تشکیل شده است. جهت بیان ساختار یك شبکه چندلایه، از نمایش عبارت کوتاهی زیر استفاده میکنیم:

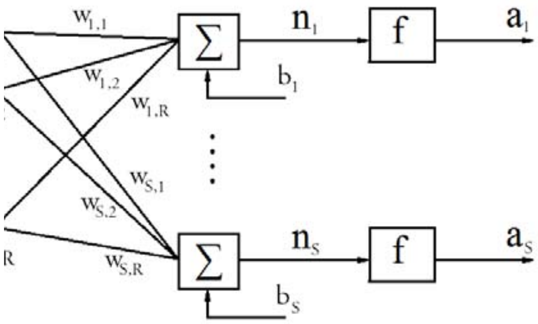
$(R - S^1 - S^2 - S^3)$

جایی که R تعداد ورودی‌ها وS^i و تعداد نرون‌ها در لایه iام باشند.

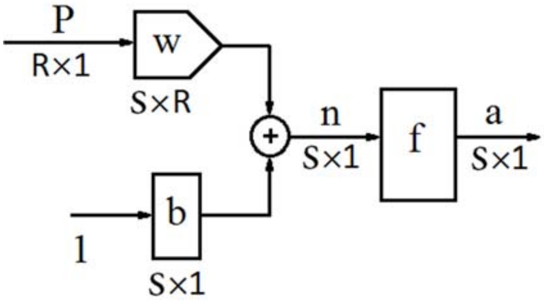
۲-٦ شبکه عصبی تك‌لایه

یك شبکه تك‌لایه با S نرون در شکل ۲- ٦ نشان داده شده است. شکل مذکور را می‌توان به فرم فشرده تصویر ۲- ۷ هم نمایش داد. ورودی شبکه با بردار p و خروجی آن با بردار a نشان داده شده است. باید توجه داشت که هر یك از ورودی‌ها به همه نرون‌ها متصل شده است. ماتریس W نیز در این حالت دارایS سطر و R ستون

می‌باشد. همان گونه که در شکل مشاهده می‌شود، لایه‌ها شامل ماتریس وزن، جمع کننده‌ها، بردار بایاس b (دارای S عنصر) و تابع تبدیل f هستند.



شکل۲- ٦شبکه تك‌لایه با S نرون



شکل ۲- ۷فرم فشرده یا ماتریسی شبکه تك‌لایه با S نرون

۲-۷ شبکه‌هایعصبی چندلایه[۱۹]

در شبکه‌هایتك‌لایه بردار ورودی P توسط نرون‌های لایه میانی، طبق رابطه۲-٦ به بردار خروجی مرتبط می‌شود. این شبکه شکل ساده ای از شبکه‌های پیشخور[۲۰] می‌باشد.

در این بخش، ایده شبکه پیشخور را به شبکه‌هایچندلایه تعمیم می‌دهیم. هر لایه، ماتریس وزن W، بردار بایاس b، بردار ورودی خالص n و بردار خروجی مختص خود را دارد. جهت تمایز متغیرهای فوق و این که کدام متغیر به کدام لایه تعلق دارد، نیاز داریم که علامت دیگری را هم به متغیرهای فوق تخصیص دهیم. از این رو از بالانویس

- multilayer networks[۱۹]

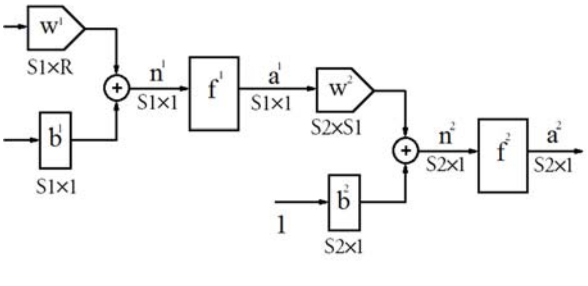
- Feedforward[۲۰]

برای مشخص نمودن لایه استفاده می‌کنیم، بنابراین ماتریس وزن را برای لایه اول با W^1مشخص می‌نماییم. با به خاطر سپردن این نماد، يك شبکه پیشخور دولایه را می‌توان به شکل ۲-۸ ترسیم نمود. همانگونه که از شکل پیداست، تعداد R ورودی، تعداد S^1نرون در لایه اول و تعداد S^2نرون در لایه دوم در شبکه چندلایه پیشخور موجود است.

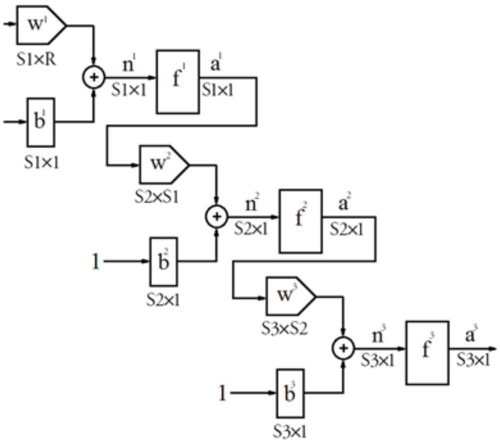
۲-۸مزیت شبکه‌های عصبی چندلایه

شبکه‌های عصبی چندلایه نسبت به شبکه‌های عصبی تك‌لایه دارای توانایی بیشتری هستند. شبکه‌های عصبی پیشخور دولایه با توابع زیگموئیدیدر لایه اول، قادرند هر تابعی را با دقت دلخواه تقریب بزنند، در حالیکه شبکه‌های عصبی تك‌لایه از چنین توانایی برخوردار نیستند. به نظر می‌رسد که تعداد درجات آزادی (مثلاً تعداد ورودی‌ها، خروجی‌ها و نرون‌های هر لایه) برای طراحی يك شبکه چندلایه پیشخور زیاد باشد. ولی باید توجه داشت که تعداد ورودی‌های شبکهR و تعداد خروجی‌های شبکهS^l، بر اساس مسئله خاص که شبکه قرار است حل کند، به دست می‌آیند. به عبارت دیگر این دو پارامتر جزء پارامترهای آزاد طراحی نیستند، بلکه انتخاب آنها بستگی دارد به مسئله‌ای که در حال بررسی است. مثلاً اگر می‌خواهیم سیستمی را که دارای چهار ورودی و سه خروجی است، توسط شبکه‌های عصبی تقریب بزنیم، باید يك شبکه عصبی انتخاب

کنیم که دارای چهار ورودی $(R = 4)$ و سه خروجی $(S^l = 3)$ باشد (به عبارتی تعداد نرونهای لایه خروجی برابر ۳ باشد). علاوه بر این، ویژگی‌های خروجی مطلوب شبکه نیز، در انتخاب نوع تابع محرك لایه خروجی تأثیر دارد. مثلاً اگر خروجی شبکه تنها ۱ یا ۱-باشد، بهتر است تابع محرك لایه خروجی، تابع متقارن آستانه‌ای دومقداره حدی باشد. بنابراین می‌توان گفت که ساختار شبکه‌های عصبی تك‌لایه پیشخور را، کاملاً از روی ویژگی‌های مسئله مورد بررسی، می‌توان مشخص نمود.



شکل ۲- ۸شبکه پیشخور دولایه



شکل۲- ۹فرم ساده شده ی شبکه پیشخور با دولایه میانی

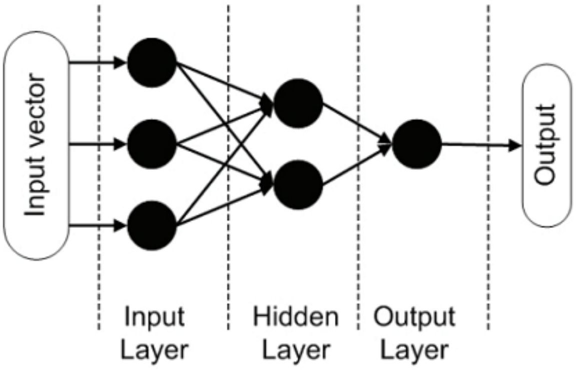
در مورد شبکه عصبی چندلایه، مطالب فوق صادق نیست. یعنی از روی ویژگی‌های مسئله مورد بررسی، نمی‌توان اطلاعاتی در مورد تعداد نرونهای لایه میانی کسب نمود.

درواقع تعداد اندکی از مسائل وجود دارند که برای آنها با توجه به مسئله خاص، امکان پیش بینی تعداد نرونهای لایه میانی وجود داشته باشد.

۲-۹ شبکه عصبی MLP[۲۱]

شبکه‌های عصبی پرسپترون تك‌لایه از این مشکل اساسی برخوردارند که تنها توانایی حل آن دسته از مسائل طبقه بندی را دارند که بطور خطی از هم مستقل‌اند، محققین از این امر آگاه بودند ولی به علت عدم ارائه قانون یادگیری که بتوان جهت تنظیم پارامترهای شبکه بکار برد، توپولوژی شبکه MLP ناقص بود. توسعه الگوریتمBP با فراهم آوردن روشی کارا از نظر محاسباتی، رنسانسی در شبکه‌های عصبی ایجاد کرد. شبکه‌های عصبیMLP در واقع نوعی از شبکه‌هایی عصبی چندلایه هستند که در آن نرون‌های هر لایه بطور کامل به نرون‌های لایه بعدی متصل شده باشد. در ادامه شکل یك شبکه عصبیMLP کهاز به هم پیوستن سه شبکه تك‌لایه درست شده را میبینید.خروجی‌های لایه اول، بردار ورودی لایه دوم را تشکیل می‌دهند و به همین ترتیب بردار خروجی لایه دوم، بردار ورودی‌های لایه سوم را می‌سازند و خروجی‌های لایه سوم، پاسخ واقعی شبکه را تشکیل می‌دهند. به عبارتی دیگر روند جریان سیگنالی در شبکه، در یك مسیر پیشخور صورت میگیرد.

- Multilayer perceptron[۲۱]



شکل ۲- ۱- شبکه MLP

در شبکهMLPعموماً دو نوع سیگنال استفاده می‌شود. یک نوع سیگنال‌هایی هستند که در مسیر رفت حرکت می‌کنند، و دسته دیگر سیگنال‌هایی هستند که در مسیر برگشت حرکت می‌کنند. به دسته اول سیگنال‌های تابعی و به دسته دوم سیگنال‌های خطا میگویند. دلیل این نامگذاری این است که سیگنال‌های دسته نخست، بر اساس تابعی از ورودی‌های هر نرون و پارامترهای شبکه متناظرش محاسبه می‌شوند ولیسیگنال‌های دسته دوم، با محاسبه شدن از روی سیگنال خطا و توزیع برگشتی، از لایه خروجی به لایه‌های دیگر شبکه منشعب شده و روی آنها تاثیر می‌گذارد. در این شبکه‌ها برای اینکه بتوانیم رابطه غیرخطی بین ورودی و خروجی را برقرار کنیم الزاماً باید از توابع محرک غیرخطی استفاده کنیم، متداول‌ترین توابع تحریک غیرخطی که در این شبکه‌ها استفاده می‌شوند، توابع زیگموئید و تانژانت هیپربولیک می‌باشند. مزیت این دو تابع تحریک

مشتق پذیری آنهاست که در ادامه خواهیم دید که عملکرد الگوریتم BP به آن وابسته است.

۱۰-۲الگوریتمپس‌انتشارخطا[۲۲]

الگوریتم پس‌انتشارخطا که در روش آموزش باناظر کارایی دارد، حالت کلی تر قانون دلتا[۲۳] است و از نوع الگوریتم‌های بهینه‌سازی گرادیان‌نزولی[۲۴]می‌باشد. این الگوریتم یك روش متداول برای آموزش شبکه‌های عصبی مصنوعی است که سعی می‌کند یك تابع هدف (که در اینجا خطای سیستم است) را مینیمم کند. این الگوریتم که در دسته الگوریتم‌های بهینه‌سازی چندمرحله‌ایطبقه‌بندی می‌شود، ابتدا در سال ۱۹۶۹ معرفی شد ولی کاربرد آن در شبکه‌های عصبی تا سال ۱۹۷٤ ناشناخته مانده بود، استفاده از این الگوریتم که به رنسانسی در زمینه شبکه‌های عصبی انجامید باعث معروفیت آن شد. برای استفاده از این الگوریتم احتیاج به بردار ورودیP و خروجی‌های مطلوب t داریم که دسته داده‌های آموزش را تشکیل می‌دهند. در این الگوریتم لازم است که توابع محرك شبکه، دیفرانسیل پذیر باشند. برای درك بهتر،روند الگوریتم پس‌انتشارخطا را میتوان به دو قسمت مجزا تقسیم کرد: انتشار و بروزرسانی وزن‌ها.

---

[۲۲] - Back propagation
[۲۳] - delta rule
[۲٤] - gradient descent

۲-۱۰-۱ انتشار[۲۵]

هر انتشار درای دو مرحله زیر است:

۱- انتشار مستقیم، که داده‌ها به شبکه داده می‌شوند تا خروجی‌های شبکه بدست آیند.

۲- پس‌انتشار، که خروجی شبکه با خروجی مطلوب داده‌هامقایسه می‌شوند و طی آن مقدار دلتا برای تمام نرون‌ها محاسبه می‌شود.

۲-۱۰-۲ بروزرسانی وزن‌ها[۲۶]

بروزرسانی وزن‌ها را نیز میتوان به دو مرحله تقسیم کرد:

۱- ضرب دلتای خروجی در ورودی تابع فعال ساز برای یافتن گرادیان وزن.

۲- تغییر مقدار وزن در خلاف جهت گرادیان بدست آمده به وسیله کسر ضریبی از گرادیان، از وزن فعلی.

این ضریب که بر سرعت و کیفیت یادگیری تأثیر بسزایی دارد به ضریب یادگیری[۲۷] معروف است. با تغییر ضریب یادگیری میتوان سرعت همگرا شدن شبکه را تنظیم کرد، لازم به ذکر است که انتخاب ضریب یادگیری نامناسب میتواند در همگرا شدن شبکه به مقادیر مطلوب اختلال ایجاد کند.

- Propagation[۲۵]
- Weight update[۲۶]
- learning rate[۲۷]

مراحل انتشار و بروزرسانی وزن‌ها را آنقدر تکرار میکنیم تا کارایی شبکه به مقدار قابل قبولی برسد.

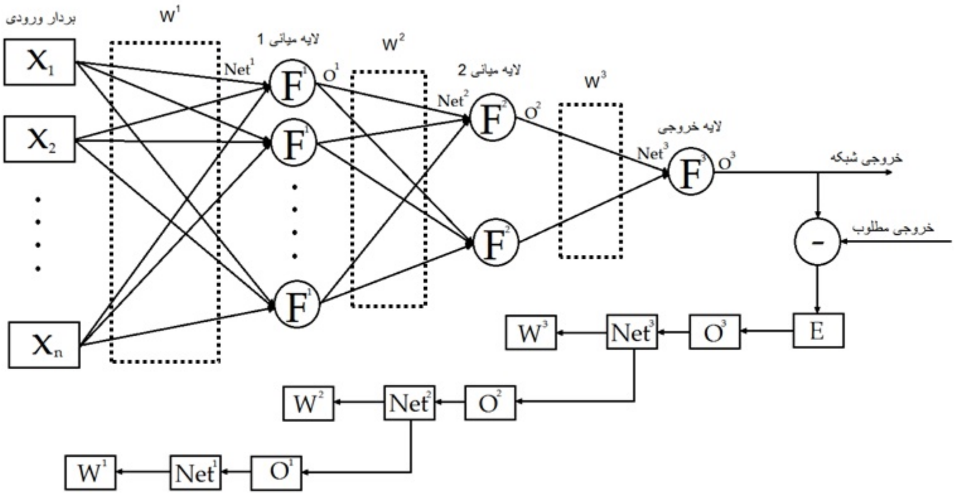
الگوریتم پس‌انتشارخطا را میتوان به دو روش به شبکه اعمال نمود، روش برخط[۲۸] و روش دسته‌ای[۲۹]، در روش برخط پس از هربار انتشار ورودی‌ها وزن‌های شبکه تنظیم می‌شوند در حالی که در روش دسته ای پس از تعداد زیادی انتشار یکبار وزن‌های شبکه را تنظیم میکنیم. روش دسته ای به حافظه بیشتری نیاز دارد در حالی که در روش برخط تعداد دفعات تنظیم وزن‌ها بیشتر است. ما در این تحقیق از روش برخط استفاده کرده ایم.

۲- ۱۱فرمول بندی الگوریتم پس‌انتشارخطا برای یك شبکهMLP

فرمول بندی این الگوریتم را برای یك شبکه MLP با ساختار $(R - S^1 - S^2 - S^3)$ در زیر مشاهده می‌فرمایید:

---

[۲۸] - on-line
[۲۹] - Batch learning



شکل ۲- ۱۲سیگنال انتشار و سیگنال پس‌انتشارخطا در یک شبکه MLP

خطای خروجی شبکه را میتوان به کمک فرمول زیر محاسبه کرد:

$$E(k) = \frac{1}{2}e^3(k) = \frac{1}{2}\left(d(k) - O^3(k)\right)^2 \quad (۲- ۷)$$

$$= \frac{1}{2}\left(d(k) - F\left(Net^3\,(k)\right)\right)^2$$

مقدار تغییر وزن‌های شبکه به کمک مشتقات خطا از رابطه زیر بدست می‌آید:

$$\Delta W^i = -\eta . \frac{\partial E}{\partial W^i} \quad (۲- ۸)$$

با پخش سیگنال پس‌انتشارخطا داریم:

$$\frac{\partial E}{\partial W^3} = \frac{\partial E}{\partial O^3} . \frac{\partial O^3}{\partial Net^3} . \frac{\partial Net^3}{\partial W^3} \quad (۲- ۹)$$

$$\frac{\partial E}{\partial W^2} = \frac{\partial E}{\partial O^3} . \frac{\partial O^3}{\partial Net^3} . \frac{\partial Net^3}{\partial O^2} . \frac{\partial O^2}{\partial Net^2} . \frac{\partial Net^2}{\partial W^2} \quad (۲- ۱۰)$$

$$\frac{\partial E}{\partial W^1} = \frac{\partial E}{\partial O^3} . \frac{\partial O^3}{\partial Net^3} . \frac{\partial Net^3}{\partial O^2} . \frac{\partial O^2}{\partial Net^2} . \frac{\partial Net^2}{\partial O^1} . \frac{\partial O^1}{\partial Net^1} . \frac{\partial Net^1}{\partial W^1} \quad (۲- ۱۱)$$

که به سادگی میتوان تغییرات وزن‌های شبکه را به شکل زیر محاسبه کرد:

$$\Delta W_3 = -\eta.\frac{\partial E}{\partial W^3} = -\eta.\frac{\partial E}{\partial O^3}.\frac{\partial O^3}{\partial Net^3}.\frac{\partial Net^3}{\partial W^3}$$

$$\Delta W_3 = -\eta.(-e).F'^3.O^2 = \eta.e.F'^3.O^2$$ (۲-۱۲)

$$\Delta W_2 = -\eta.\frac{\partial E}{\partial W^2} = -\eta.\frac{\partial E}{\partial O^3}.\frac{\partial O^3}{\partial Net^3}.\frac{\partial Net^3}{\partial O^2}.\frac{\partial O^2}{\partial Net^2}.\frac{\partial Net^2}{\partial W^2}$$

$$\Delta W_2 = -\eta.(-e).F'^3.W^3.F'^2.O^1 = \eta.e.F'^3.W^3.F'^2.O^1$$ (۲-۱۳)

$$\Delta W_1 = -\eta.\frac{\partial E}{\partial W^1} = -\eta.\frac{\partial E}{\partial O^3}.\frac{\partial O^3}{\partial Net^3}.\frac{\partial Net^3}{\partial O^2}.\frac{\partial O^2}{\partial Net^2}.\frac{\partial Net^2}{\partial O^1}.\frac{\partial O^1}{\partial Net^1}.\frac{\partial Net^1}{\partial W^1}$$

$$\Delta W_1 = -\eta.(-e).F'^3.W^3.F'^2.W^2.F'^1.X = \eta.e.F'^3.W^3.F'^2.W^2.F'^1.X$$ (۲-۱٤)

۲- ۱۲ محدودیت‌های الگوریتم پس‌انتشارخطا

الگوریتم پس‌انتشارخطا دارای چهار محدودیت اساسی است:

- سرعت همگرایی این الگوریتم پائین است.
- همگرایی آن تضمین شده نیست.
- احتمال همگرا شدن الگوریتم به مینیمم‌های محلی وجود دارد.
- در آموزش پس‌انتشارخطا نیاز است داده‌های ورودی نرمالیزه شوند.

۲-۱۳خلاصه

در این فصل با شبکه‌های عصبی آشنا شدیمو دیدیم که تمام شبکه‌ها از اجزای کوچکی به نام نرون ساخته می‌شوند، همچنین دیدیم که چطور میتوان یك نرون بیولوژیکی را

با فرمول‌های ریاضی مدلسازی کرد. پس از آن با کنار هم گذاشتن نرون‌هاشروع به طراحی شبکه‌های عصبی مصنوعی کردیم و راجع به شبکه عصبی MLP توضیحاتی دادیم، در انتها نیز نیم‌نگاهی داشتیم به الگوریتم پس‌انتشارخطا که یکی از متداول‌ترین الگوریتم‌های آموزش شبکه‌های عصبی باناظراست. در بخش بعدی بطور خاص به معرفی دو شبکه عصبی برگشتی می‌پردازیم، خواهید دید که این شبکه‌ها شباهت زیادی با شبکه‌هایMLP دارند. سپس الگوریتم پس‌انتشارخطا را برای هرکدام از این شبکه‌ها تعمیم می‌دهیم.

فصل ۳

شبکه‌های برگشتی المان و جردن

۳-۱ مقدمه

نظر به استفاده بهینه‌تر از شبکه‌های عصبی چند لایه در این فصل به بررسی و طراحی شبکه‌های عصبی که قابلیت بالایی در سیستم‌های پویا دارند پرداخته می‌شود. دراینگونهشبکه‌ها،شبکهازتعدادینرون‌هایدارایفیدبکتشکیلشدهاستکههمانندحافظهدرشبکهعملمی‌کنند. ابتدا به بحث و طراحی شبکه‌های برگشتی المن و جردن می‌پردازیم و سپس سعی در پیاده‌سازی الگوریتم‌های مربوطه آنها داریم.

۲-۳ شبکه عصبی المان

درسال ۱۹۹۰ اِلمن[۳۰] شبکهبافیدبکیرامعرفیکرد.

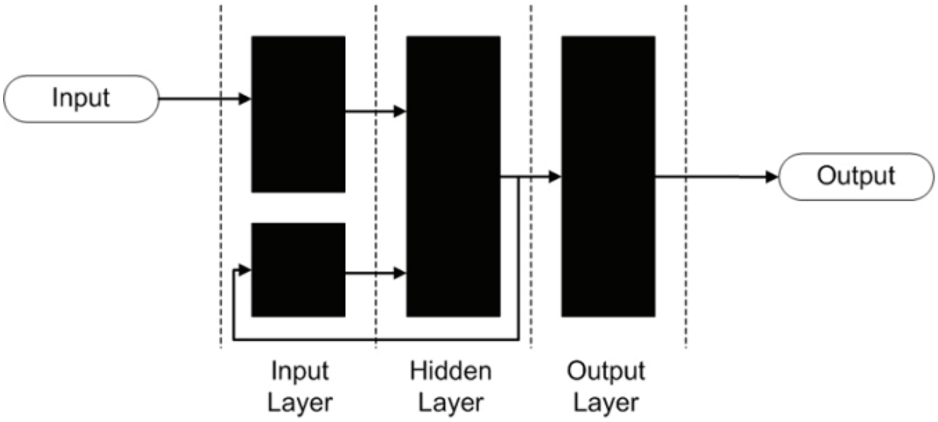
درشبکهٔاِلمنتعدادیازنرونهایمخفیوجوددارندکههمهیابرخیازخروجیهایلایهٔاولعلاوهبروارد شدنبهلایهٔدوم،بهاینلایهٔمخفینیزواردمیشوندوپسازآندوبارهبهلایهٔاولبازمیگردند،یعنی میتوانگفتکهاینشبکهداراینوعیحافظهمیباشدکهتأثیرمقادیرقبلیرادرلایهاینگاهمیداردو همراهباورودیهایجدیدبهشبکهواردمینماید. این مسئله باعث میشود شبکه المان تواناییهای بیشتری از شبکههای پرسپترون چندلایه پیدا کند.

۳-۳ فرمول بندی الگوریتم پس‌انتشارخطا برای شبکه عصبی المان

در این بخش الگوریتم پس‌انتشارخطا را به یك شبکه عصبی المان با یك لایه میانی و یك لایه خروجی تعمیم می‌دهیم، ورودی‌های شبکه را میتوان به شکل یك لایه در نظر گرفت،در این صورت مقدار O^0همان بردار ورودی خواهد بود. ساختار این شبکه را در شکل ۳-۱ مشاهده میکنید.

---

[۳۰] - JeffElman



شکل ۳-۱ ساختار شبکه المان

معادلات پیشخور برای این شبکه به صورت زیر است:

$$\begin{cases} net^1 = W^1.X(t) + W^c.X^c(t) + b^1 \\ Where: \\ X^c(t) = O^1(t-1)\,, O^1(0) = 0 \end{cases}$$

$$O^1 = F^1(net^1)$$

$$\begin{cases} net^2 = W^2.O^1(t) + b^2 \\ F^2(net^2) = O^2(t) = y(t) \end{cases}$$

که در آن W^i، b^i، net^i و O^i به ترتیب‌وزن، بایاس، خروجی خالص وخروجی لایه iام هستند و $X^c(t)$ ورودی لایه فیدبك در زمان t(که در واقع خروجی لایه پنهان در زمان $(t-1)$ است)می‌باشد.

معادلات لازم برای اصلاح وزن لایه‌های w^1و w^2 در این شبکه کاملاً مشابه با شبکه MLP بوده و به شکل زیر بدست می‌آیند.

$$\Delta W^2 = -\eta.\frac{\partial E}{\partial W^2} = -\eta.\frac{\partial E}{\partial O^2}.\frac{\partial O^2}{\partial Net^2}.\frac{\partial Net^2}{\partial W^2}$$

$$\Delta W^2 = -\eta.(-e).F'^2.O^1 = \eta.e.F'^2.O^1$$

$$\Delta W^1 = -\eta.\frac{\partial E}{\partial W^1} = -\eta.\frac{\partial E}{\partial O^2}.\frac{\partial O^2}{\partial Net^2}.\frac{\partial Net^2}{\partial O^1}.\frac{\partial O^1}{\partial Net^1}.\frac{\partial Net^1}{\partial W^1}$$

$$\Delta W^1 = -\eta.(-e).F'^2.W^2.F'^2.O^0 = \eta.e.F'^2.W^2.F'^1.O^0$$

همچنین جزئیات معادلات BP برای آموزش لایه برگشتی را در زیر مشاهده می‌فرمایید:

$$\begin{cases} \Delta W^c = -\eta\frac{\partial E}{\partial W^c} = -\eta.\frac{\partial E}{\partial O^2}.\frac{\partial O^2}{\partial net^2}.\frac{\partial net^2}{\partial O^1}.\frac{\partial O^1}{\partial net^1}.\frac{\partial net^1}{\partial W^c} \\ \qquad = \eta.\delta^1.X_c(t) \\ \qquad = \eta.\delta^1.O^1(t-1) \\ Where: \\ \qquad \delta^1 = e.F^{2\prime}.W^2.F^{1\prime} \end{cases}$$

۳- ٤ شبکه عصبی جردن

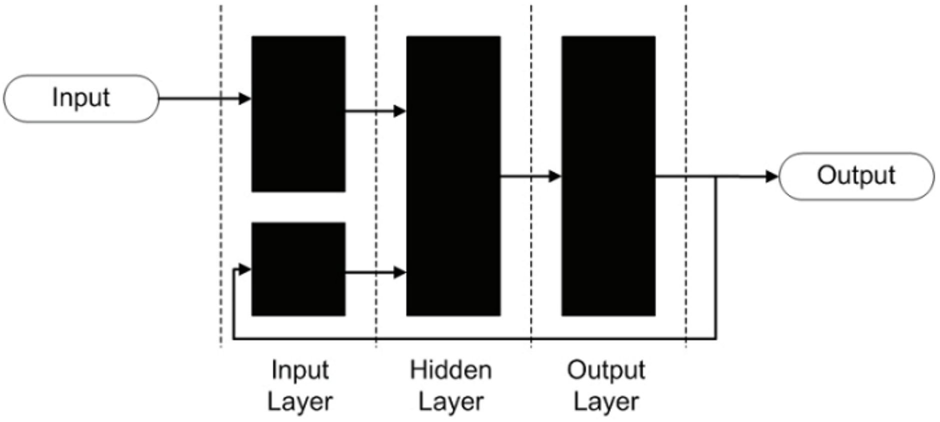
شبکهٔعصبیجردنکهدرسال ۱۸۸٦

ارائهگردید،ساختاریبسیارشبیهبهشبکهعصبیاِلمندارد،بااینتفاوتکهشبکه جردن فیدبك خود را از خروجی شبکه میگیرد و آن را همراه با ورودی به لایه اول اعمال می‌کند، در این شبکه به لایه فیدبك، لایه حالت هم میگویند، چرا که حالت قبلی سیستم را در خود نگه میدارد.

۳-۵ فرمول بندی الگوریتم پس‌انتشارخطا برای شبکه عصبی جردن

این بار الگوریتم پس‌انتشارخطا را به یك شبکه عصبی جردن با یك لایه میانی و یك لایه خروجی تعمیم می‌دهیم، مانند شبکه المان میتوان ورودی‌های شبکه را به شکل یك

لایه در نظر گرفت، که در این صورت مقدارO^0 همان بردار ورودی خواهد بود. خواهیم دید که معادلات این دو شبکه بسیار شبیه به هم هستند. ساختار این شبکه را در شکل ۳-۲ مشاهده میکنید.



شکل ۳-۲ ساختار شبکه جردن

معادلات پیشخور برای این شبکه به صورت زیر است:

$$\begin{cases} net^1 = W^1.X(t) + W^c.X^c(t) + b^1 \\ Where: \\ X^c(t) = O^2(t-1) = y(t-1)\,, y(0) = 0 \end{cases}$$

$$O^1 = F^1(net^1)$$

$$\begin{cases} net^2 = W^2.O^1(t) + b^2 \\ F^2(net^2) = O^2(t) = y(t) \end{cases}$$

که در آن W^i، b^i،net^i و O^i به ترتیبوزن، بایاس، خروجی خالص و خروجی لایه iام هستند و $X^c(t)$ ورودی لایه فیدبك در زمان t(که در واقع خروجی شبکه در زمان $(t-1)$ است)می‌باشد.

معادلات لازم برای اصلاح وزن لایه‌های W^1 و W^2 در این شبکه نیز کاملاً مشابه با شبکه MLP و به صورت زیر است:

$$\Delta W^2 = -\eta . \frac{\partial E}{\partial W^2} = -\eta . \frac{\partial E}{\partial O^2} . \frac{\partial O^2}{\partial Net^2} . \frac{\partial Net^2}{\partial W^2}$$

$$\Delta W^2 = -\eta . (-e) . F'^2 . O^1 = \eta . e . F'^2 . O^1$$

$$\Delta W^1 = -\eta . \frac{\partial E}{\partial W^1} = -\eta . \frac{\partial E}{\partial O^2} . \frac{\partial O^2}{\partial Net^2} . \frac{\partial Net^2}{\partial O^1} . \frac{\partial O^1}{\partial Net^1} . \frac{\partial Net^1}{\partial W^1}$$

$$\Delta W^1 = -\eta . (-e) . F'^2 . W^2 . F'^2 . O^0 = \eta . e . F'^2 . W^2 . F'^1 . O^0$$

معادلات لازم برای آموزش لایه فیدبک هم به جز تفاوت جزئی مربوط به ورودی لایه مشابه شبکه المان و به صورت زیر بدست می‌آیند.

$$\begin{cases} \Delta W^c = -\eta \dfrac{\partial E}{\partial W^c} = -\eta . \dfrac{\partial E}{\partial O^2} . \dfrac{\partial O^2}{\partial net^2} . \dfrac{\partial net^2}{\partial O^1} . \dfrac{\partial O^1}{\partial net^1} . \dfrac{\partial net^1}{\partial W^c} \\ \qquad = \eta . \delta^1 . X^c(t) \\ \qquad = \eta . \delta^1 . O^2(t-1) \\ Where: \\ \qquad \delta^1 = e . F^{2\prime} . W^2 . F^{1\prime} \end{cases}$$

فصل ٤

نتایج بدست آمده از شبیه سازی شبکه‌های عصبی

۴-۱ پیش بینی سری زمانی به کمك شبکه عصبی MLP

در این بخشکد یك شبکه عصبی MLPسه لایه با ساختار (۴-۹-۱) که در نرم افزار مطلب شبیه سازی شده را با هم بررسی میکنیم. شبکه را با ۷۰۰ داده ورودی با روش آموزش باناظر و به کمك الگوریتم پس‌انتشارخطاآموزش داده و سپس با ۳۰۰ داده دیگر، صحت عملکرد آن را می‌سنجیم، به دلیل محدودیت‌های الگوریتم پس‌انتشارخطاداده‌های ورودی به صورت نرمالیزه شده و در بازه[۱،۱-]قرار دارند. نرخ آموزش را نیز ۰/۰۱ در نظر گرفته ایم، نرخ آموزش بزرگ میتواند به واگرایی الگوریتم منتهی شود در حالی که نرخ یادگیری خیلی کم باعث کاهش سرعت همگرایی شبکه می‌شود. این ۷۰۰ داده‌آموزشی۱۰۰ بار به شبکه اعمال می‌شوند. تابع فعالسازی را برای لایه خروجی خطی، و برای لایه میانی به شکل تانژانت هیپربولیك (سیگموئیدی) در نظر گرفتهایم، این امر به دلیل طبیعت مسئله مورد بررسی است که یك تابع غیرخطیمی‌باشد. در ادامه کد مطلب این شبکه عصبی را با هم مشاهده میکنیم:

```
load'Input','Target'; % loading input & output samples file
y=Target;% x => inputvector 4*1000
x=Input;% y => output vector 1*1000

ns = 700;% num of train samples
nt = 300;% num of test samples

ni = 4;% number of inputlayer neurons
nh = 9;% number of hidden layer nerouns
no = 1;% number of output layer neurons

epoch =100;% nymber of epochs

msqe = [];% mean square error for training samples
t_msqe=[];% mean square error for testing samples
alpha =0. 01;% learning rate

Da = y(1:ns);% output samples for network training
Dt = y(ns+1:end);% output samples for network testing
```

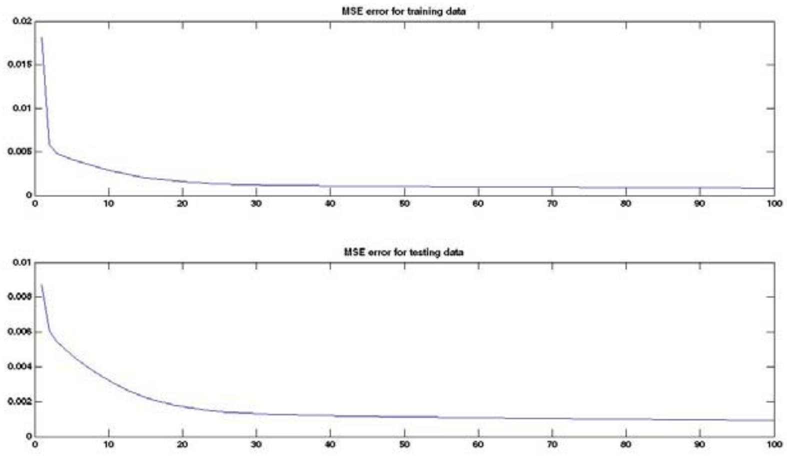
```
pa = x(:,1:ns);% input samples for network
training
pt = x(:,ns+1:end);% input samples for network
testing

d_f1 = zeros(nh,nh);% hidenlayer function
derivative
d_f2 = zeros(no,no);% output layer function
derivative

w1 = rand(nh,ni)*1-0. 5;% weight initialization
w2 = rand(no,nh)*1-0. 5;

b1= rand(nh,1)*1;% bias initialization
b2= rand(no,1)*1;

for epochs=1:epoch
error = [];% training error
for i=2:ns% loop to train the network
% feedforward
in = x(:,i);
n1 = w1*in+b1;
a1 = tansig(n1);% hidden layer output
n2 = w2*a1+b2;
a2 = purelin(n2);% network output
Y(i) = a2;
```

```
error = [error,Da(i)-a2];
e = Da(i)-a2;

% backpropagation
d_f2 = 1;
S2 = d_f2*e;

delta_w2 = (alpha)*S2*a1';
w2 = w2+delta_w2;% output layer wight update
delta_b2 = (alpha)*S2;
b2 = b2+delta_b2;% output layer bias update

d_f1 =diag(ones(size(a1))-a1. ^2);
S1 = d_f1*w2'*S2;

delta_w1 = (alpha)*S1*in';
w1 = w1 + delta_w1;% hidden layer wight update
delta_b1 = (alpha)*S1;
b1 = b1+delta_b1;% hidden layer bias update

end

msqe = [msqe,mse(error)];% msqe => mean square
error
t_error = [];% testing error
```

```
Yt = [];% network test output

for i=1:length(pt)% loop to test the network
in = pt(:,i);
n1 = w1*in+b1;
a1 = tansig(n1);% hidden layer output
n2 = w2*a1+b2;
a2 = purelin(n2);% network output
t_error = [t_error,Dt(i)-a2];
Yt = [Yt,a2];
end
% error ploting
t_msqe(epochs) = mse(t_error);
subplot(2,1,1)
plot(msqe)
title('MSE error for training data')
subplot(2,1,2)
plot(t_msqe)
title('MSE error for testing data')
drawnow

end
mse_error = mse(error)
t_msqe = mse(t_error)
```

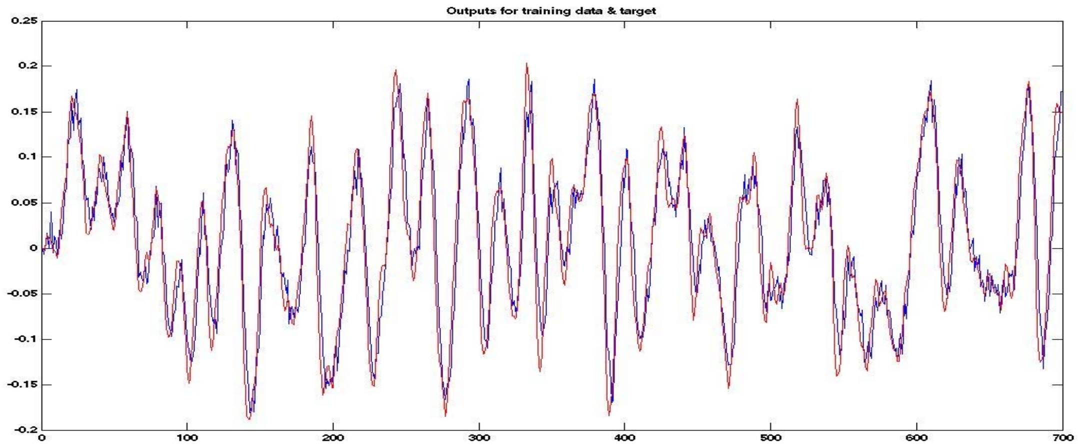
```
% output plotting
figure
plot(Y);hold on;plot(Da,'red');
title('Outputs for training data & target')
figure
plot(Yt);hold on;plot(Dt,'red');
title('Outputs for testing data & target')
% errors
Error_CALC(Dt,Yt);
```

نمودار ٤-١ همگرایی خروجی‌های شبکه به خروجی‌های مطلوب را به وسیله اندازه‌گیری خطای mseنشان می‌دهد.

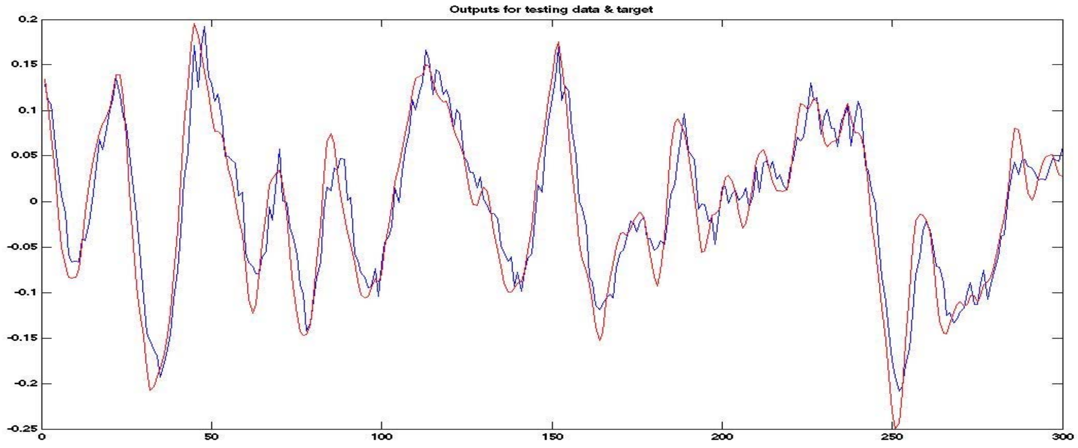


نمودار ٤-١ خطای mse خروجی شبکه برای داده‌های آموزش و تست

نتایج این شبیه سازی را درنمودارهای ٤-٢ و ٤-٣ ملاحظه می‌فرمایید. نمودار قرمز رنگ مربوط به سری زمانی و نمودار آبی رنگ مربوط به خروجی شبکه عصبی می‌شود. واضح است که خروجی شبکه،خروجی مطلوب را دنبال می‌کند ولی دارای مقداری پرتی در نقاط بحرانی بوده و ماکزیمم-مینیمم‌های محلی را به خوبی پوشش نداده است.



نمودار ٤-٢ خروجی داده‌های آموزش شبکه به رنگ آبی و خروجی مطلوب به رنگ قرمز



نمودار ٤-٣ خروجی داده‌های تست شبکه به رنگ آبی و خروجی مطلوب به رنگ قرمز

۲-٤پیش بینی سری زمانی به کمك شبکه عصبی المان

در این قسمتکد یك شبکه عصبی المان سه لایه با ساختار (٤-٩-١) را با هم بررسی میکنیم. شبکه را با ٧٠٠ داده ورودی با روش آموزش باناظر و به کمك الگوریتم پس‌انتشارخطا آموزش داده و سپس با ٣٠٠ داده دیگر، صحت عملکرد آن را می‌سنجیم، به دلیل محدودیت‌های الگوریتم پس‌انتشارخطاداده‌های ورودی به صورت نرمالیزه شده و در بازه[١،١-]قرار دارند. نرخ آموزش را نیز ٠/٠١ در نظر گرفته ایم. این ٧٠٠ داده‌آموزشی ١٠٠ بار به شبکه اعمال می‌شوند. طبق طراحی شبکه‌های المان، تابع فعالسازی را برای لایه خروجی خطی و برای لایه میانی به شکل تانژانت هیپربولیك (سیگموئیدی) در نظر گرفته ایم. در ادامه کد مطلب این شبکه عصبی را با هم مشاهده میکنیم.

```
load'Input','Target'; % loading input & output samples file
y=Target;% x => inputvector 4*1000
x=Input;% y => output vector 1*1000

ns = 700;% num of train samples
nt = 300;% num of test samples
```

```
ni = 4;% number of inputlayer neurons
nh = 9;% number of hidden layer nerouns
no = 1;% number of output layer neurons

epoch =100;% nymber of epochs

msqe = [];% mean square error for training
samples
t_msqe=[];% mean square error for testing
samples
alpha =0. 01;% learning rate

Da = y(1:ns);% output samples for network
training
Dt = y(ns+1:end);% output samples for network
testing

pa = x(:,1:ns);% input samples for network
training
pt = x(:,ns+1:end);% input samples for network
testing

d_f1 = zeros(nh,nh);% hidenlayer function
derivative
d_f2 = zeros(no,no);% output layer function
derivative
```

```
xr= zeros(nh,1);% hidden layer input

w1 = rand(nh,ni)*1-0. 5;% weight initialization
w2 = rand(no,nh)*1-0. 5;
wr = rand(nh,nh)*1-0. 5;

b1= rand(nh,1)*1;% bias initialization
b2= rand(no,1)*1;

for epochs=1:epoch
error = [];% training error
for i=2:ns% loop to train the network
% feedforward
in = x(:,i);
n1 = w1*in+wr*xr+b1;
a1 = tansig(n1);% hidden layer output
n2 = w2*a1+b2;
a2 = purelin(n2);% network output
Y(i) = a2;
error = [error,Da(i)-a2];
e = Da(i)-a2;
% backpropagation
d_f2 = 1;
S2 = d_f2*e;
```

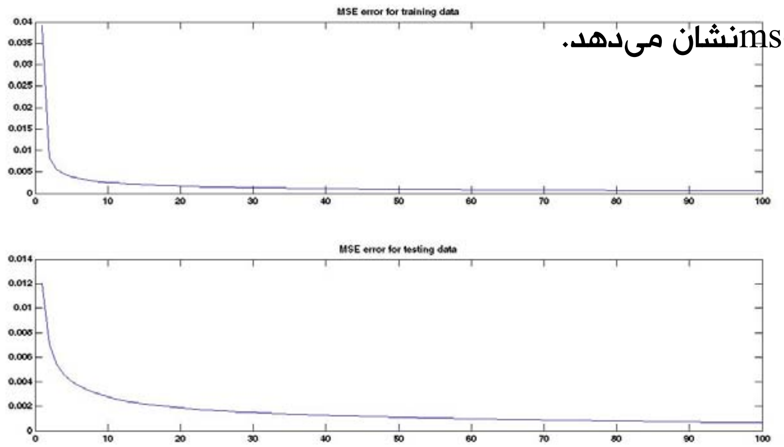
```
delta_w2 = (alpha)*S2*a1';
w2 = w2+delta_w2;% output layer wight update
delta_b2 = (alpha)*S2;
b2 = b2+delta_b2;% output layer bias update

d_f1 =diag(ones(size(a1))-a1. ^2);
S1 = d_f1*w2'*S2;

delta_w1 = (alpha)*S1*in';
w1 = w1 + delta_w1;% hidden layer wight update
delta_b1 = (alpha)*S1;
b1 = b1+delta_b1;% hidden layer bias update

delta_wr = (alpha)*S1*xr';
wr = wr+delta_wr;%feedback layer wight update

xr = a1;% new feedback input
end

msqe = [msqe,mse(error)];% msqe => mean square
error
t_error = [];% testing error
Yt = [];% network test output
xr_t = zeros(nh,1);

for i=1:length(pt)% loop to test the network
```

```
in = pt(:,i);
n1 = w1*in+b1+wr*xr_t;
a1 = tansig(n1);% hidden layer output
n2 = w2*a1+b2;
a2 = purelin(n2);% network output
t_error = [t_error,Dt(i)-a2];
Yt = [Yt,a2];
xr_t = a1;
end
% error ploting
t_msqe(epochs) = mse(t_error);
subplot(2,1,1)
plot(msqe)
title('MSE error for training data')
subplot(2,1,2)
plot(t_msqe)
title('MSE error for testing data')
drawnow

end
mse_error = mse(error)
t_msqe = mse(t_error)
% output plotting
figure
plot(Y);hold on;plot(Da,'red');
title('Outputs for training data & target')
```

```
figure
plot(Yt);hold on;plot(Dt,'red');
title('Outputs for testing data & target')
% errors
Error_CALC(Dt,Yt);
```

نمودار ٤-٤ همگرایی خروجی‌های شبکه به خروجی‌های مطلوب را به وسیله اندازه‌گیری

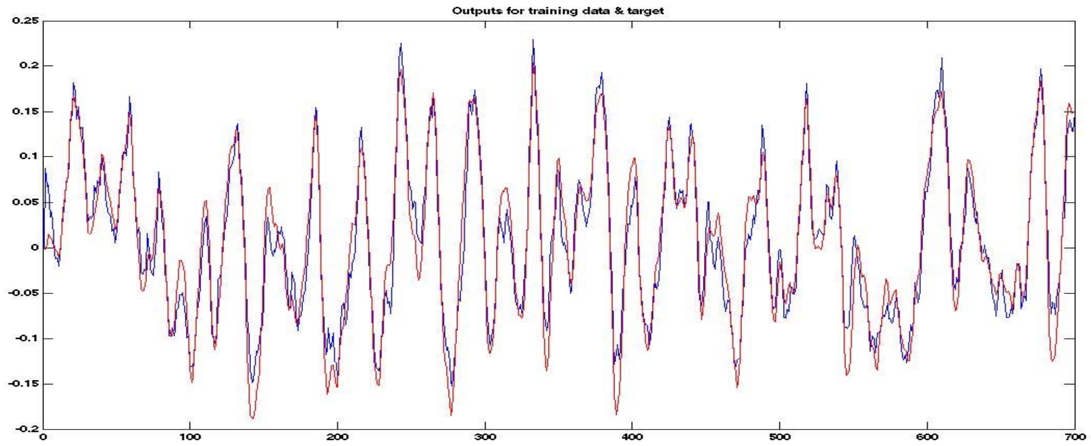
خطای mseنشان می‌دهد.



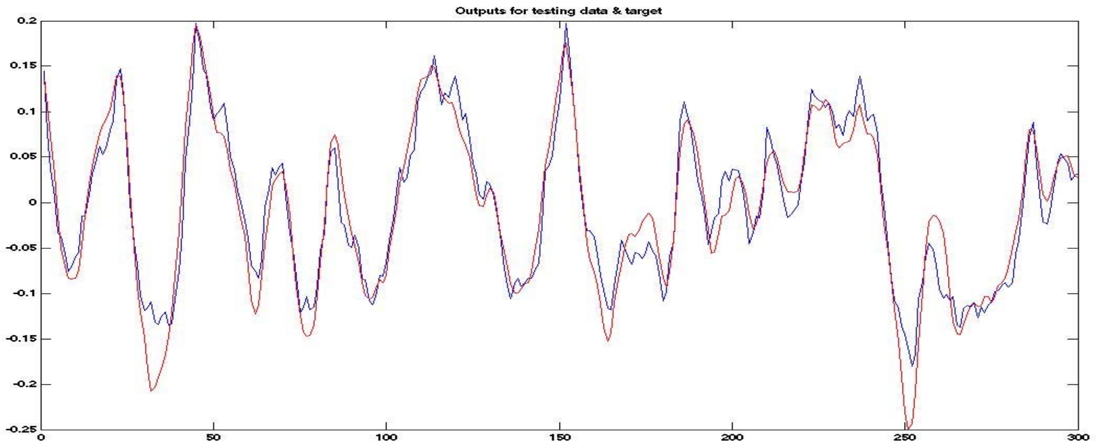
نمودار۴- ۴ خطای mse خروجی شبکه برای داده‌های آموزش و تست

نم موزش و

داده‌های تست می‌باشند.



نمودار ٤- ٥ خروجی داده‌های آموزش شبکه به رنگ آبی و خروجی مطلوب به رنگ قرمز



نمودار ٤- ٦ خروجی داده‌های تست شبکه به رنگ آبی و خروجی مطلوب به رنگ قرمز

۳-٤ پیش بینی سری زمانی به کمك شبکه عصبی جردن

در این قسمتکد یك شبکه عصبی جردن سه لایه با ساختار (٤-۹-۱) را با هم بررسی میکنیم. همانند دو شبکه قبلی، شبکه را با ۷۰۰ داده ورودی با روش آموزش باناظر و به کمك الگوریتم پس‌انتشارخطا آموزش داده ایم و سپس با ۳۰۰ داده دیگر، صحت عملکرد آن را می‌سنجیم. بردار ورودی نرمالیزه شده و درایه‌های آن در بازه[-۱،۱]قرار دارند. نرخ آموزش را نیز ۰/۰۱ در نظر گرفته ایم. این ۷۰۰ داده‌آموزشی ۱۰۰ بار به شبکه اعمال می‌شوند. طبق طراحی شبکه جردن، تابع فعالسازی را برای لایه خروجی خطی و برای لایه میانی به شکل تانژانت هیپربولیك (سیگموئیدی) در نظر گرفته ایم.

در ادامه کد مطلب این شبکه عصبی را بررسی میکنیم.

```
load'Input','Target'; % loading input & output samples file
y=Target;% x => inputvector 4*1000
x=Input;% y => output vector 1*1000

ns = 700;% num of train samples
nt = 300;% num of test samples

ni = 4;% number of inputlayer neurons
nh = 9;% number of hidden layer nerouns
no = 1;% number of output layer neurons
```

```
epoch =100;% nymber of epochs

msqe = [];% mean square error for training
samples
t_msqe=[];% mean square error for testing
samples
alpha =0. 01;% learning rate

Da = y(1:ns);% output samples for network
training
Dt = y(ns+1:end);% output samples for network
testing

pa = x(:,1:ns);% input samples for network
training
pt = x(:,ns+1:end);% input samples for network
testing



d_f1 = zeros(nh,nh);% hidenlayer function
derivative
d_f2 = zeros(no,no);% output layer function
derivative

xr= zeros(no,1);% hidden layer input
```

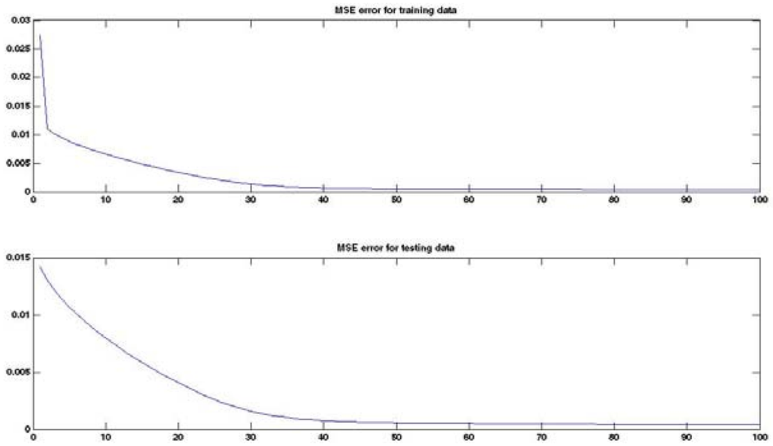
```
w1 = rand(nh,ni)*1-0. 5;% weight initialization
w2 = rand(no,nh)*1-0. 5;
wr = rand(nh,no)*1-0. 5;

b1= rand(nh,1)*1;% bias initialization
b2= rand(no,1)*1;

for epochs=1:epoch
error = [];% training error
for i=2:ns% loop to train the network
% feedforward
in = x(:,i);
n1 = w1*in+wr*xr+b1;
a1 = tansig(n1);% hidden layer output
n2 = w2*a1+b2;
a2 = purelin(n2);% network output
Y(i) = a2;
error = [error,Da(i)-a2];
e = Da(i)-a2;
% backpropagation
d_f2 = 1;
S2 = d_f2*e;

delta_w2 = (alpha)*S2*a1';
w2 = w2+delta_w2;% output layer wight update
```

```
delta_b2 = (alpha)*S2;
b2 = b2+delta_b2;% output layer bias update

d_f1 =diag(ones(size(a1))-a1. ^2);
S1 = d_f1*w2'*S2;

delta_w1 = (alpha)*S1*in';
w1 = w1 + delta_w1;% hidden layer wight update
delta_b1 = (alpha)*S1;
b1 = b1+delta_b1;% hidden layer bias update

delta_wr = (alpha)*S1*xr';
wr = wr+delta_wr;%feedback layer wight update

xr = a2;% new feedback input
end



msqe = [msqe,mse(error)];% msqe => mean square
error
t_error = [];% testing error
Yt = [];% network test output
xr_t = zeros(no,1);

for i=1:length(pt)% loop to test the network
in = pt(:,i);
```

```
n1 = w1*in+wr*xr_t+b1;
a1 = tansig(n1);% hidden layer output
n2 = w2*a1+b2;
a2 = purelin(n2);% network output
t_error = [t_error,Dt(i)-a2];
Yt = [Yt,a2];
xr_t = a2;
end
% error ploting
t_msqe(epochs) = mse(t_error);
subplot(2,1,1)
plot(msqe)
title('MSE error for training data')
subplot(2,1,2)
plot(t_msqe)
title('MSE error for testing data')
drawnow

end
mse_error = mse(error)
t_msqe = mse(t_error)
% output plotting
figure
plot(Y);hold on;plot(Da,'red');
title('Outputs for training data & target')
figure
```

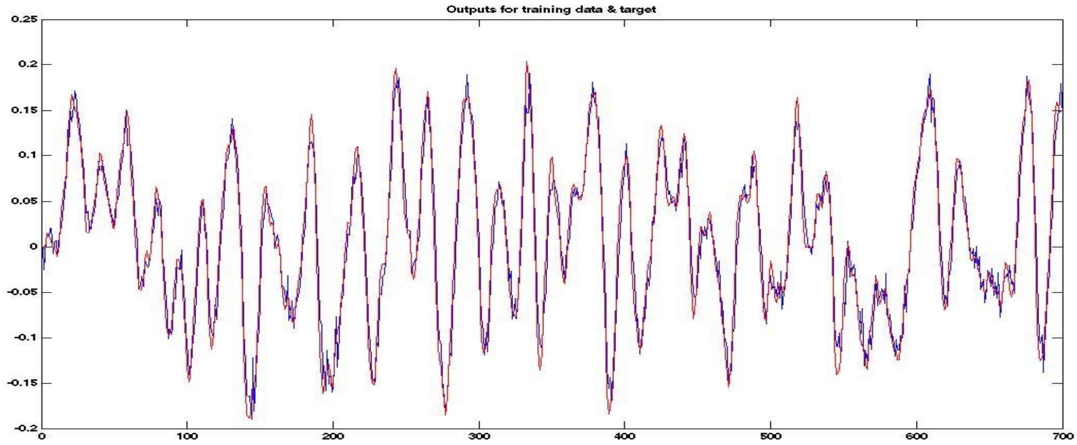
```
plot(Yt);hold on;plot(Dt,'red');
title('Outputs for testing data & target')
% errors
Error_CALC(Dt,Yt);
```

نمودار ٤-٧ همگرایی خروجی‌های شبکه به خروجی‌های مطلوب را به وسیله اندازه‌گیری خطای mseنشان می‌دهد.

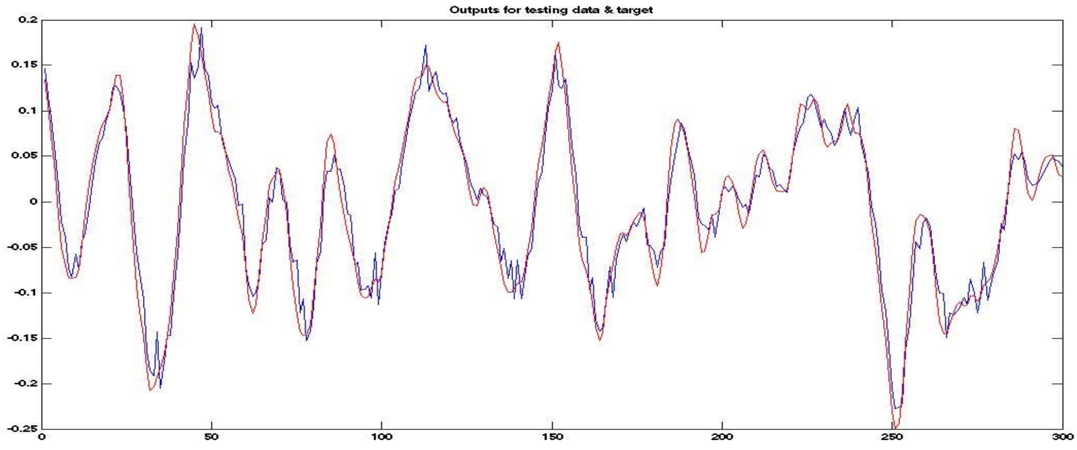


نمودار٤- ٧ خطای mse خروجی شبکه برای داده‌های آموزش و تست

نتایج این شبیه سازی را درنمودارهای ٤-٨ و ٤-٩ ملاحظه می‌فرمایید. نمودار قرمز رنگ مربوط به سری زمانی و نمودار آبی رنگ مربوط به خروجی شبکه عصبی می‌شود.



نمودار ٤- ٨ خروجی داده‌های آموزش شبکه به رنگ آبی و خروجی مطلوب به رنگ قرمز



نمودار ٤- ٩ خروجی داده‌های آموزش شبکه به رنگ آبی و خروجی مطلوب به رنگ قرمز

٤-٤ مقایسه نتایج بدست آمده از شبیه سازی شبکه‌ها

همانطور که پیش از این هم اشاره شد، در هنگام طراحی شبکه تعداد نرون‌های ورودی و خروجی وابسته به مسئله بوده و در اختیار طراح نیست، حال این سوال مطرح می‌شود که تعداد نرون‌های لایه میانی چند تا باید باشد و آیا عدد دقیقی به عنوان تعداد بهینه وجود دارد یا خیر؟ حقیقت این است که تعداد نرون‌های کم در لایه میانی باعث می‌شود شبکه نتواند تمام دانش موجود در مسئله را درك کند و تعداد بیش از حد نرون‌ها هم باعث افت کارایی شبکه می‌گردد، از این رو تعداد نرون‌های لایه میانی اصولاً در یك بازه بهینه قرار می‌گیرد که انتخاب هر عددی از این بازه رسیدن به جواب مطلوب را در پی خواهد داشت. لازم به ذکر است این بازه برای هر شبکه ای متفاوت است.

در این تحقیق در شبیه سازی تمام شبکه‌های مورد بررسی، از پارامترهای یکسانی برای طراحی استفاده کرده ایم، در این قسمت نتایج بدست آمده از شبیه سازی‌ها را در قالب اعداد و ارقام با یکدیگر مقایسه میکنیم، ارقام مورد استفاده در این قسمت پس از ۲۰ بار شبیه سازی هر شبکه بدست آمده اند و دارای ±۰٫۵٪ خطا می‌باشند.

٤-٤-١ انواع خطاهای اندازه‌گیری شده

برایمسائلپیش‌بینی،ازبرخیمعیارهایعملکردبراینشاندادنچگونگییادگیریارتباط‌هایدادهادرشبکهعصبیاستفادهمی‌شود.

ازآنجاکههریکازمعیارهایارزیابیعملکرد،جنبهخاصیراموردارزیابیقرارمی‌دهد،

دراینتحقیقاز5معیاراستفاده کرده

ایمکهسهموردا‌ولازخانوادهمحاسباتمیانگینخطایاستانداردهستند:

مربعمجذورمیانگینخطا[۳۱](RMSE)،مربعمیانگینخطایاستانداردنرمالشده[۳۲](NMSE) و

میانگین خطای بایاس (MBE)[۳۳]. موردبعدیFitnessاستکهدرصد شایستگی را نشان

میدهدودرارتباطباNMSEاست. $Fitness = (1 - NMSE) \times$

100مقدارFitnessبینصفرتاصداستومقدارصدنشاندهندهتطابقکاملدادههااست.

معیارآخر نیزدرموردخطایمطلقاست: میانگینقدرمطلقخطا[۳٤](MAE).

۲-٤-٤ نتایج شبیه سازی

نتایجمعیارهایارزیابیعملکردبرایهرسه شبکه درجدول٤-۱آوردهشدهاست.

جدول ٤-۱ مقایسه شبیه سازی شبکه‌های عصبی

شبکه	MBE	MAE	RMSE	Fitness	NMSE
MLP	۰،۰۰۰٤۳۸۲۷	۰،۰۲٤٥۱۲	۰،۰۳۰۲٥۱	٦۳،٥٤۸۳٪	۳٦،٤٥۱۷٪
المان	۰،۰۰۰۲٤۷۸	۰،۰۲۰۱٤۸	۰،۰۲٥۷۷	۷۱،۲۸۸٤٪	۲۸،۷۱۱٦٪
جردن	۰،۰۰۰۲۲۳۷۳	۰،۰۱٥۸٥٦	۰،۰۱۹٦٥۲	۷۸،۱۰٥٥٪	۲۱،۸۹٤٥٪

- Root Mean Square Error[۳۱]
- Normalized Mean Square Error[۳۲]
- Mean Bias Error[۳۳]
- Mean Absolute Error[۳٤]

فصل ۵

نتیجه گیری و پیشنهادات

همانطور که ملاحظه کردید شبکه‌های عصبی توانایی جالبی در پیش بینی سری زمانی دارند، برای کاهش خطا و همگرا شدن بیشتر به جواب‌های واقعی مسئله باید موارد زیر را در نظر داشت:

- افزایش تکرار در الگوریتم یادگیری – افزایش epoch

اگر دقت کرده باشید هنگامی که ما ۷۰۰ داده یادگیری خود را برای آموزش شبکه استفاده کردیم، این داده‌ها را ۱۰۰ بار به شبکه اعمال نمودیم تا شبکه فرصت کافی داشته باشد که دانش موجود در داده‌ها را برداشت کند، میتوان این تعداد تکرار را برای همگرایی بیشتر افزایش داد، فقط باید دقت داشت که اگر الگوریتم پس‌انتشارخطا در یك مینیمم محلی گیر بیفتد، روند همگرایی مختل شده و ثابت میگردد و خارج شدن از این وضعیت معمولاً طول میکشد. البته پیش از این نیز اشاره شد که افزایش بیش از حد تکرار باعث کاهش نویزپذیری سیستم می‌شود.

- افزایش حجم داده‌های ورودی

واضح است که ما آینده را بر اساس اتفاقات گذشته پیش بینی میکنیم، هرچه اطلاعات ما از گذشته کامل تر باشد پیش بینی ما دقیقتر خواهد بود، اصولاً هرچه یك شبکه عصبی با داده‌های ورودی بیشتری آموزش ببیند، باعث درك بهتر مفهوم و فرمول مسئله توسط شبکه می‌شود.

- تغییر در تعداد نرون‌های لایه میانی

تعداد نرون‌های لایه ورودی و لایه خروجی ارتباط مستقیم با خواص مسئله دارد، به همین علت نمیتوان تعداد آنها را تغییر داد ولی تعداد نرون‌های لایه میانی یك پارامتر طراحی است، بدین معنی که طراح شبکه انتخاب می‌کند که چندین نرون در لایه میانی موجود باشد، با توجه به مسئله مورد نظر می‌توان با آزمون و خطا تعداد نرون مناسبتری را انتخاب کرد.

- تغییر در تعداد لایه‌های میانی

ما در این تحقیق از یك لایه میانی استفاده کرده ایم، ولی میتوان تعداد لایه‌های میانی را نیز تغییر داد، این مورد نیز از پارامترهای طراحی یك شبکه عصبی است.

- تغییر الگوریتم یادگیری

الگوریتم پس‌انتشارخطااصولاًدرشبکه‌هایپیشخور بهتر عمل می‌کند، از این رو میتوان از الگوریتم‌های دیگری مثل الگوریتم ژنتیك یا الگوریتم POS برای آموزش شبکه‌ها استفاده کرد.

- تغییر توابع محرك

ما به علت ماهیت غیرخطی مسئله مطرح شده در این تحقیق از توابع تبدیل زیگموئیدی استفاده کردیم، باید توجه داشت که انتخاب تابع تحریك مناسب با مسئله بسیار مهم بوده و در رسیدن به پاسخ صحیح نقش دارد.

- تغییر نوع شبکه

این که از چه نوع شبکه ای استفاده کنیم تا بهترین پاسخ را برای مسئله خود بگیریم مسئله مهمی است، معمولاً برای هر مسئله‌ای میتوان از چندین نوع شبکه عصبی برای رسیدن به پاسخ بهره گرفت، برای مثال میتوان مسئله مطرح شده در همین تحقیق رابه کمك یکی از شبکه‌های عصبی المان- جردن، گاما، LSTM یا به کمك داینامیك سیناپس تحلیل کرد.

پیوست ۱

در این پیوست کد مطلب تولید کننده سری زمانی بررسی شده در این تحقیق را ملاحظه می‌فرمایید، بردار input ورودی شبکه عصبی و output خروجی مطلوب شبکه را نمایندگی می‌کنند.

```
u = 2*rand(1,1001)-1;
y(1) = 0;
y(2) = 0;
for i=2:1001
a = 1.753*y(i)-0.819*y(i-1)+0.012*u(i);
b = 0.011*u(i-1)/(1+y(i-1)^2);
y(i+1) = a+b;
end
Input =
[u(2:1001);u(1:1000);y(2:1001);y(1:1000)];
Target = y(3:1002);
save'Input','Target';
```

نتیجه هربار اجرای این برنامه یك سری زمانی آشوبناك جدید خواهد بود. در این تحقیق برای هر سه شبکه عصبی از سری زمانی یکسانی استفاده کرده‌ایم.پیوست ۲

در این پیوست کد مطلب تشخیص خطا که با عنوان تابع “Error_CLAC” در شبیه سازی‌ها مورد استفاده قرار گرفت را مشاهده میکنید. در این کد ٥ نوع خطایی که از آنها برای مقایسه شبکه‌های عصبی استفاده کردیم محاسبه می‌شود.

```
function Error_CALC(Y,YHAT)
```

```
NMSE = 100*((sqrt(sum((Y-
YHAT).^2)))/sqrt(sum((Y-mean(Y(:))).^2))) ;
FITNESS = 100-NMSE ;
RMSE = sqrt(sum((Y-YHAT).^2)/length(Y)) ;
MAE = (sum(abs(Y-YHAT))/length(Y)) ;
MBE = (sum((Y-YHAT))/length(Y)) ;

disp(['NMSE is : ' num2str(NMSE) '%']) ;
disp(['FitNess is : ' num2str(FITNESS) '%']) ;
disp(['RMSE is : ' num2str(RMSE)]) ;
disp(['MAE is : ' num2str(MAE)]) ;
disp(['MBE is : ' num2str(MBE)]) ;
end
```

پیوست ۳

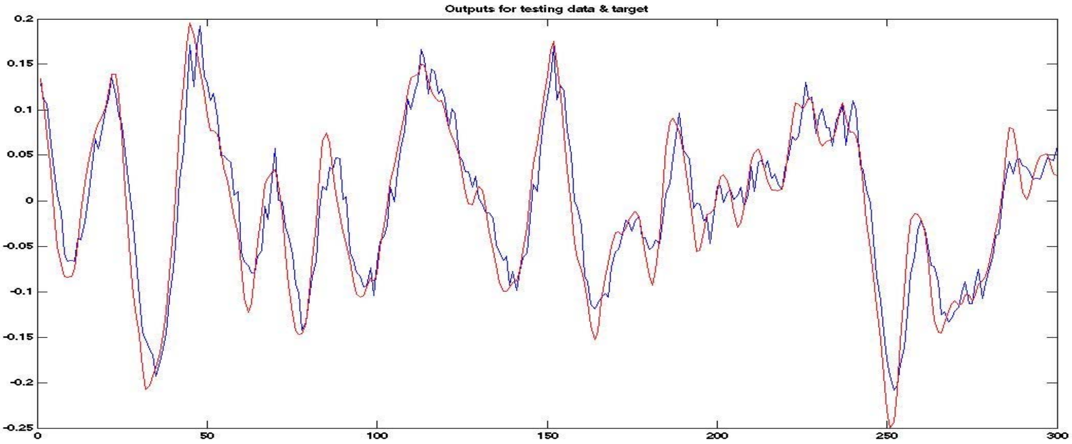
در این پیوست تغییر بعضی پارامترهای شبکه را برای شبکه MLPبررسی میکنیم، علت انتخاب این شبکه، سادگی درك آن نسبت نسبت به شبکه‌های عصبی برگشتی می‌باشد.

جدول پیوست ٣ – نتایج تغیر در پارامترهای طراحی شبکه MLP

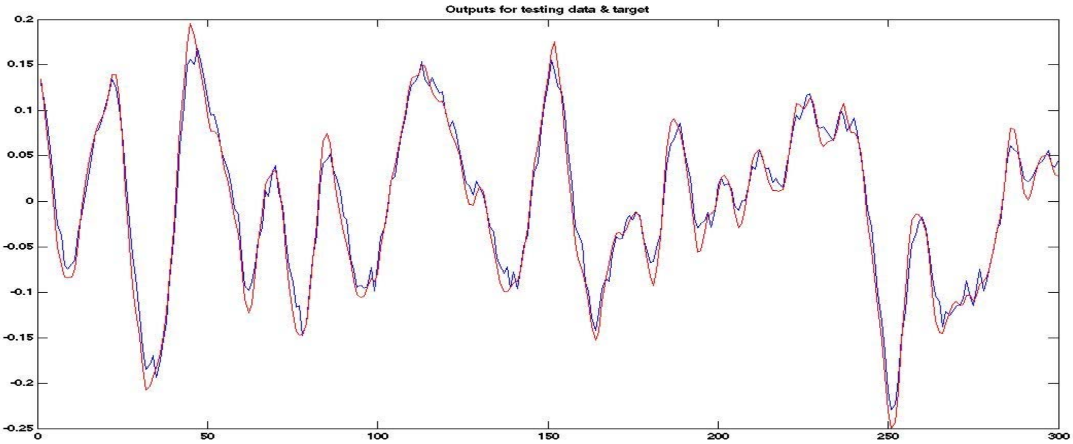
	N1	N2	N3	N4	N5	N6
Epochs[٣٥]	١٠٠	١٠٠	٢٠٠	٢٠٠	١٠٠	١٠٠
nh[٣٦]	٩	٥٠	٩	٥٠	٩	٩
eta[٣٧]	٠٫٠٠١	٠٫٠٠١	٠٫٠٠١	٠٫٠٠١	٠٫٠٠٠٥	٠٫٠٠١
f1[٣٩]	sig[٣٨]	sig	sig	sig	sig	Sig
f2[٤١]	lin[٤٠]	lin	lin	lin	lin	Sig
Fitness	٦٣٫٥٤٨٣٪	٨٢٫٥٦٦٧٪	٨٠٫٣٩٤٢٪	٩٠٫٢١٧٧٪	٧٧٫٠٢٠٩٪	٧٦٫٣٠٠٧٪
NMSE	٣٦٫٤٥١٧٪	١٧٫٤٣٣٣٪	١٩٫٦٠٥٨٪	٩٫٧٨٢٣٪	٢٢٫٩٧٩١٪	٢٣٫٦٩٩٣٪
RMSE	٠٫٠٠٣٠٢٥١	٠٫٠٠١٤٦٩٩	٠٫٠٠١٦٧٤٥	٠٫٠٠٠٨٤٧١٦	٠٫٠٠١٩٧٩٥	٠٫٠٠٢٠٢٤٤

٣٥- تعداد تکرار الگوریتم
٣٦- تعداد نرون‌های لایه میانی
٣٧- نرخ یادگیری
٣٨- sigmoid
٣٩- تابع فعالسازی لایه میانی
٤٠- linear
٤١- تابع فعالسازی لایه خروجی

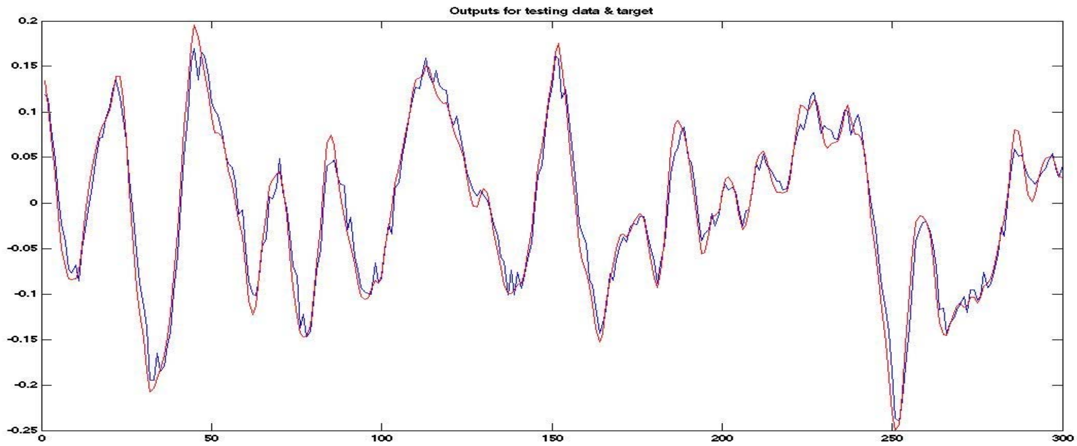
MAE	٠,٠٢٤٥١٢	٠,٠١١٨٢٦	٠,٠١٣٢٤٤	٠,٠٠٦٦٣٢	٠,٠١٦١١١	٠,٠١٦٣٧٧
MBE	٠,٠٠٤٣٨٢٧	٠,٠٠٤٢٣٥٢	٠,٠٠٢٤٥٦٨	٠,٠٠٢٩١٢	٠,٠٠١٨٣٥	٠,٠٠٢٠٦٨٥



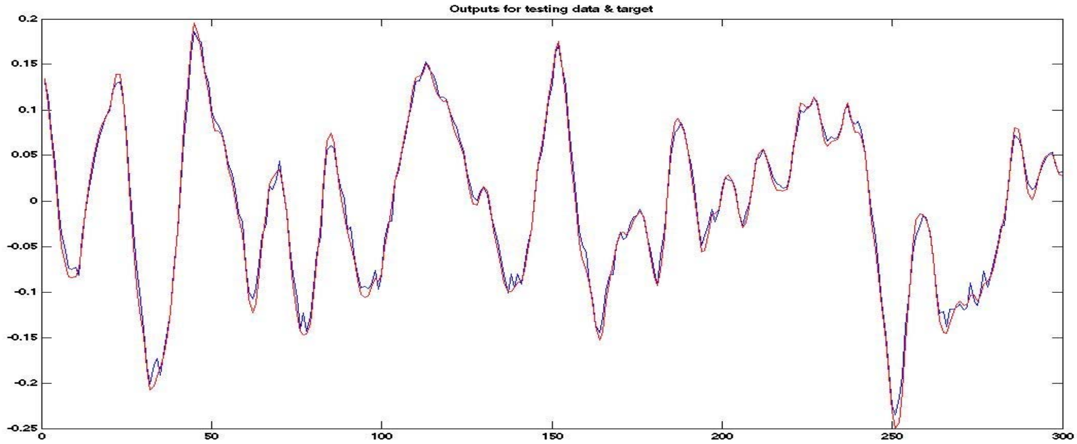
*- ١ نتیجه داده‌های تست شبکه N1



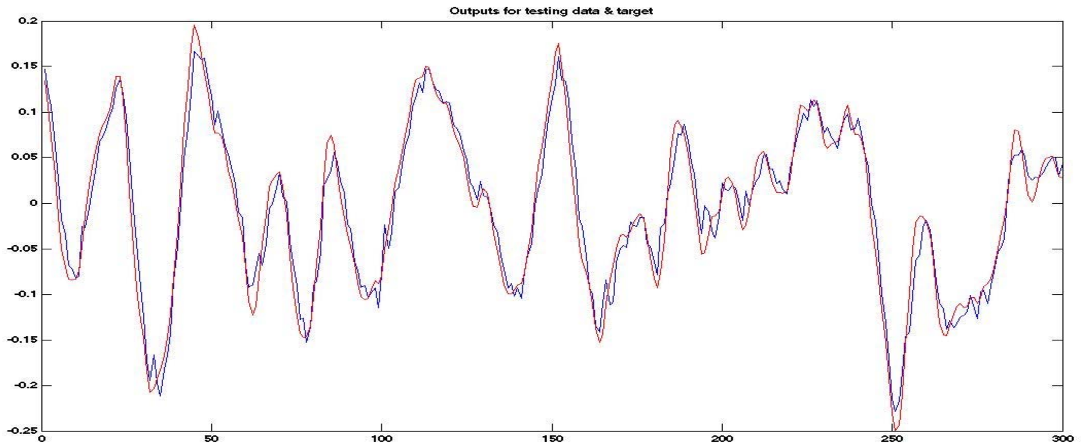
*- ٢ نتیجه داده‌های تست شبکهN2



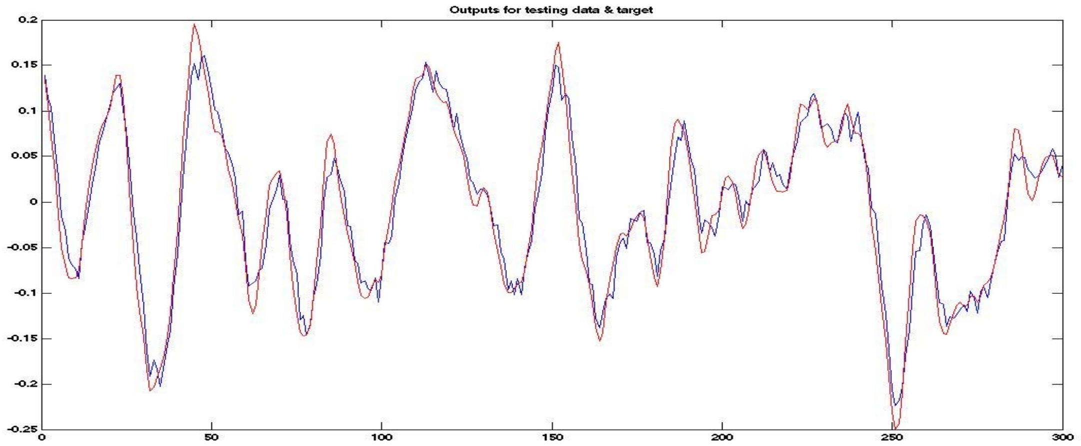
*-۳ نتیجه داده‌های تست شبکهN3



*- ٤ نتیجه داده‌های شبکهN4



*- ٥ نتیجه داده‌های شبکهN5



*- ٦ نتیجه داده‌های تست شبکهN6

مراجع

منابع فارسی

۱- منهاج محمد باقر، مبانی شبکه‌های عصبی، جلد اول، انتشارات دانشگاه صنعتی امیر کبیر، واحد تفرش، مرکز نشر پروفسور حسابی، ۱۳۸۷

۲- آر. بیل و تی. جکسون، «آشنایی با شبکه‌های عصبی مصنوعی» موسسه انتشارات علمی دانشگاه شریف، ۱۳۸۶

۳- شکیبا مریم، پیش بینی نرخ ترافیك ورودی به مسیریاب با استفاده از شبکه‌های عصبی با قابلیت وزن‌های دینامیك دار، دانشگاه صنعتی خواجه نصیر الدین طوسی، پایان نامه برای دریافت درجه کارشناسی ارشد، آبان ۱۳۸۶

۴- اسنل، ریچارد، عصب شناسی بالینی، ترجمه دکتر محمد اکبری، دکتر غلامرضا حسن زاده، دکتر علیرضا فتح اللهی و دکتر سیامك ملکی، نشر فردا، تهران، ۱۳۸۸

منابع انگلیسی

1- Mazzarello, Paolo (2010). Golgi: A Biography of the Founder of Modern Neuroscience. Translated by Aldo Badiani and Henry A. Buchtel. New York

2- Neural Networks as Cybernetic Systems 2nd and revised edition, Holk Cruse

3- Paul J. Werbos (1994). The Roots ofBackpropagation. From Ordered Derivatives to Neural Networks and Political Forecasting

4- Bertsekas, D. P. , Tsitsiklis, J. N. (1996). Neuro-dynamic programming. Athena Scientific