نما
مجله الکترونیکی پژوهشگاه اطلاعات و مدارک علمی ایران
شماره اول دوره ششم
تاریخ انتشار این مقاله: ٢٦ مهر ١٣٨٥

معرفي شبكه هاي عصبي مصنوعي

سيد رضا شاه اميري
Email : reza@rsh.ir
دانشجوي كارشناسي ارشد مهندسي كامپيوتر – نرم افزار
دانشگاه آزاد اسلامي واحد نجف آباد

Web site : www.RSH.ir

چكيده :

در اين نوشتار به معرفي شبكه هاي عصبي مصنوعي[١] و ساختارهاي آنها به صورت خلاصه ميپردازيم. در ابتدا نرونهاي شبكه هاي عصبي طبيعي معرفي شده و طرز كار آنها نشان داده شده است. سپس مدل مصنوعي اين نرونها و ساختار آنها ، مدل رياضي آنها ، شبكه هاي عصبي مصنوعي و نحوه آموزش و بكار گيري اين شبكه ها به همراه روش يادگيري گراديان كاهنده نشان داده شده است. تمركز بيشتر بر نوعي از اين شبكه ها بنام شبكه هاي عصبي مصنوعي پرسپترون چند لايه ميباشد. ابزاهايي نيز براي پياده سازي اين شبكه ها نام برده شده است.

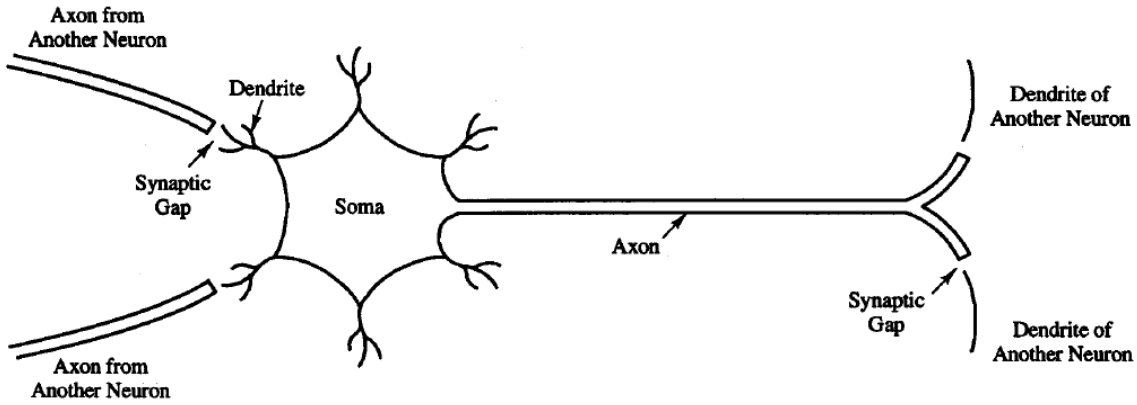
كليدواژه‌ها: شبكه هاي عصبي مصنوعي ، شبكه هاي عصبي، *Artificial Neural networks Multi layered Perceptron, Gradient descent*

مقدمه :

در ساليان اخير شاهد حركتي مستمر از تحقيقات صرفا تئوري به تحقيقات كاربردي بخصوص در زمينه پردازش اطلاعات ، براي مسائلي كه براي آنها راه حلي موجود نيست و يا براحتي قابل حل نيستند بوده ايم. با عنايت به اين امر ، علاقه اي فزاينده اي در توسعه تئوريكي سيستمهاي ديناميكي هوشمند مدل آزاد [٢]-كه مبتني بر داده هاي تجربي ميباشند – ايجاد شده است. *ANN* ها جزء اين دسته از سيستمهاي مكانيكي قرار دارند كه با پردازش روي داده هاي تجربي ، دانش يا قانون نهفته در وراي داده ها را به ساختار شبكه منتقل ميكنند.به همين خاطر به اين سيستمها هوشمند گفته ميشود ، زيرا بر اساس محاسبات روي داده هاي عددي يا مثالها ، قوانين كلي را فرا ميگيرند.اين سيستمها در مدلسازي ساختار نرو-سيناپتيكي[٣] مغز بشر ميكوشند.

البته اين سخن كه" *ANN* ها در مدلسازي مغز بشر ميكوشند" اغراق آميز ميباشد.دانشمندان هرچه بيشتر در مورد مغز بشر تحقيق ميكنند و مي آموزند ، بيشتر در مي يابند كه مغز بشر دست نيافتني است.در حقيقت در مورد مغز و ساختار سيستم عصبي انسان اطلاعات زيادي بدست آمده است. ولي پياده سازي ساختاري با پيچيدگي مغز انسان بر اساس اطلاعاتي و تكنولوژي كه امروزه وجود دارد غير ممكن ميباشد.

ما ميتوانيم يک نرون عصبي انسان و عملكرد آنرا را توسط مدل هاي رياضي ، مدلسازي كنيم.شكل١ ساختار يک نرون طبيعي را نشان ميدهد.



شكل۱ - ساختار نرون طبيعي انسان

هر نرون طبيعي از سه قسمت اصلي تشكيل شده است :

۱. بدنه سلول *(Soma)*
۲. دندريت *(Dendrite)*
۳. اكسون *(Axon)*

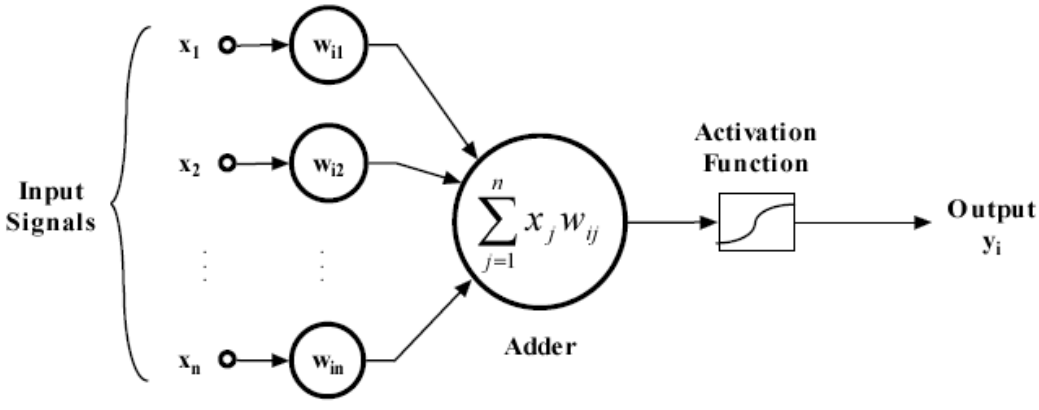
دندريتها به عنوان مناطق دريافت سيگنالهاي الكتريكي ، شبكه هايي تشكيل يافته از فيبرهاي سلولي هستند كه داراي سطح نامنظم و شاخه هاي انشعابي بيشمار ميباشند .دندريتها سيگنالهاي الكتريكي را به هسته سلول منتقل ميكنند. بدنه سلول انرژي لازم را براي فعاليت نرون فراهم كرده و برروي سيگنالهاي دريافتي عمل ميكند ، كه با يک عمل ساده جمع و مقايسه با يک سطح آستانه مدل ميگردد. اكسون بر خلاف دندريتها از سطحي هموارتر و تعداد شاخه هاي كمتري برخوردار ميباشد. اكسون طول بيشتري دارد و سيگنالهاي الكتروشيميايي دريافتي از هسته سلول را به نرونهاي ديگر منتقل ميكند.محل تلاقي يک اكسون از يک سلول به دندريتهاي سلولهاي ديگر را سيناپس ميگويند.توسط سيناپسها ارتباطات مابين نرونها برقرار ميشود.به فضاي مابين اكسون و دندريتها فضاي سيناپسي گويند.

در حقيقت دندريتها به عنوان ورودي نرون و اكسون به عنوان خروجي و فضاي سيناپسي محل اتصال ايندو ميباشد. زمانيكه سيگنال عصبي از اكسون به نرونها و يا عناصر ديگر بدن مثل ماهيچه ها ميرسد ، باعث تحريک آنها ميشود.

نرونها از هريک از اتصالات ورودي خود يک ولتاژ كم دريافت ميكند (توسط سيگنال عصبي ورودي) و آنها را با هم جمع ميزند. اگر اين حاصل جمع به يک مقدار آستانه رسيد اصطلاحا نرون آتش ميكند و روي اكسون خود يک ولتاژ خروجي ارسال ميكند كه اين ولتاژ به دندريتهايي كه به اين اكسون متصلند رسيده و باعث يكسري فعل و انفعالهاي شيميايي در اتصالات سيناپسي ميشود و ميتواند باعث آتش كردن نرونهاي ديگر شود. تمامي فعاليتهاي مغزي انسان توسط همين اتش كردنها انجام ميشود.

حافظه كوتاه مدت انسان جرقه هاي لحظه اي الكتريكي ميباشند و حافظه بلند مدت به صورت تغييرات الكتروشيميايي در اتصالات سيناپسي ذخيره ميشود كه عمدتا منجر به تغيير يونها ميشود.

همانگونه كه گفته شد ما ميتوانيم توسط مفاهيم رياضي يک نرون طبيعي را مدل كنيم.شكل ۲ يک نرون عصبي مصنوعي را نشان ميدهد.



شكل ۲ ساختار يک نرون مصنوعي

سيگنالهاي ورودي X_1 تا X_n معادل سيگنالهاي عصبي ورودي و وزنهاي تا معادل مقادير اتصالات سيناپسي ورودیهاي نرون ميباشند كه جمعا ورودي نرون را تشكيل داده است.

تابع جمع كننده $\sum_{j=1}^{n} x_i w_{ij}$ فشرده تمامي عمليات هسته سلول را انجام ميدهد.در مورد تابع فعال سازي[٤] سازي[٥]) صحبت خواهد شد.

خروجي نرون توسط تابع زير مشخص ميشود:

$$y_i = ActivationFunction(\sum_{j=1}^{n} x_i w_{ij})$$

به خصوصيات زير در مورد مغز انسان توجه كنيد :

محاسبات كاملا به صورت توزيع شده وموازي انجام ميشود.

"يادگيري" جايگزين برنامه ريزي از قبل ميشود.

در مغز انسان يک ساختار ALU مشخص وجود ندارد. ALU ، حافظه و كنترل همگي در يک ساختار درهم تنيده شبكه اي از تعداد بسيار زيادي نرون توزيع و پخش شده است.

مغز انسان توسط يک پروسه يادگيري مي آموزد كه در پاسخ به يک ورودي ، چه خروجي را توليد كرده و ارسال كند. اين فرآيند يادگيري در حقيقت توسط تنظيم اتصالات سيناپسي در نرونهاي طبيعي و معادل آنها در ANN ها ،يعني تنظيم وزنهاي W_{ij} نرونهاي مصنوعي انجام ميشود. در حقيقت در طراحي يک نرون مصنوعي فقط كافيست وزنهاي را مشخص كنيم تا شبكه عصبي بتواند خروجي مورد نظر را از ورودي خاص توليد كند. متدهاي مختلف يادگيري وجود دارد كه ميتواند بر اساس زوج مرتبهاي <خروجي، ورودي> مقدار وزنها را بدست آورد.

حال بر اساس مطالب گفته شده ميتوان به تعريف ANN ها پرداخت.

تعريف شبكه هاي عصبي مصنوعي

يک ساختار شبكه اي از تعدادي عناصر مرتبط به هم به نام نرون كه هر نرون داراي ورودیها و خروجيهايي است و يک عمل نسبتا ساده و محلي[٦] را انجام ميدهد.شبكه هاي عصبي عموما عملكرد خود را طي يک پروسه يادگيري[٧] فرا ميگيرد.

شبكه هاي عصبي كاربردهاي عمده اي در تشخيص الگو ، گروه بندي ، پيش بيني يا برون يابي و ... دارا ميباشد.

تاريخچه

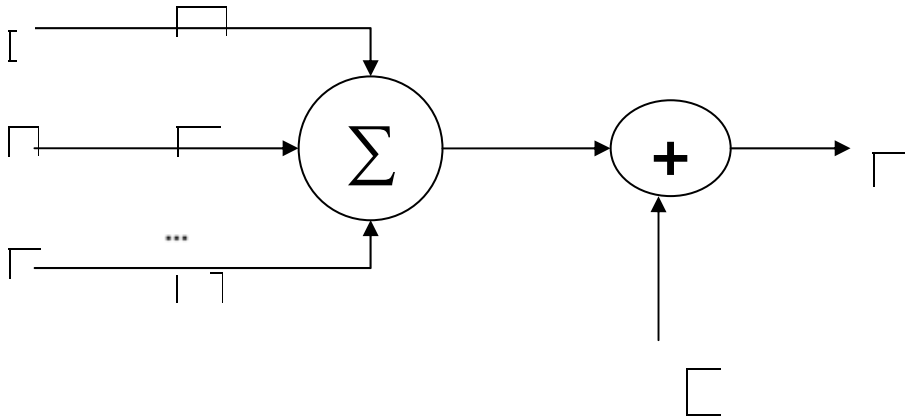
در سال ١٩٤٩ مدل *MP* شبكه هاي عصبي مصنوعي توسط مک كالوخ و پيت مطرح شد كه يک مدل خطي ساده بود . سپس پرسپترون الگوريتمهاي يادگيري را ارائه نمود. سال ١٩٦٩ آغاز افول موقت شبكه هاي عصبي شد. زيرا عدم توانايي شبكه هاي عصبي در حل مسائل غير خطي آشكار شد. *ANN* هاي آن زمان فقط قادر به حل مسائلي بودند كه ميتوانستيم پاسخهاي آن مسئله را توسط يک خط در محور مختصات از هم جدا كنيم. در ١٩٨٢ هاپفيلد با معرفي شبكه هاي چند لايه و الگوريتمهاي يادگيري داراي *feedback* ، راه حلي براي حل موارد غير خطي ارائه كرد.در اين زمان بود كه شبكه هاي بازگشتي ، خودسازمانده ، *RBF* ، *Autoregressive* و روش يادگيري هيبيان مطرح شد.از نيمه دهه نود ، نسل سوم *ANN* ها مطرح شدند كه عبارت بودند از :

- تعيين محدوديتهاي تئوري و عملي شبكه
- عموميت و حدود آن
- ANNتركيب و الگوريتمهاي ژنتيكي و منطق فازي

در نهايت امروزه استفاده عملي و پياده سازي تجاري و سخت افزاري *ANN* ها ممكن شده است .

معرفي مدل نرون ساده خطي

بر اساس ايده گرفته شده از نرونهاي طبيعي ، ميتوان مدل نرون مصنوعي را رائه كرد كه بتواند ورودیها را باهم تركيب كرده و يک خروجي از آنها به وجود آورد. در ساده ترين حالت نرون ورودیهاي وزن دار را با هم جمع ميكند. شكل ٣ يک نرون ساده خطي را نشان ميدهد.

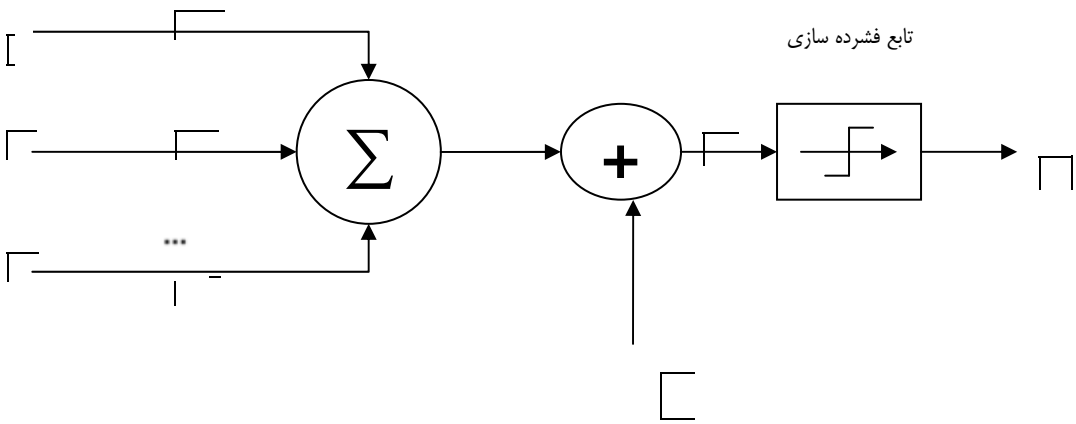


شکل۳ : نرون ساده خطي

$$\underline{i} = \begin{bmatrix} i_1 \\ i_2 \\ \vdots \\ i_d \end{bmatrix} \qquad \underline{w} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}$$

ميتوان شبکه عصبي را به صورت يک سيستم محرک / پاسخ در نظر گرفت. مهم اينست که بدانيم تعليم شبکه يعني تنظيم پارامترهاي آن به نحويکه رفتار محرک / پاسخ طبق خواسته باشد. در حين تعليم در واقع مجهولات ما وزنها(*w*) ميباشند. وزنها در غالب حافظه ها عمل ميکنند و نحوه توليد پاسخ را مشخص ميکنند. درصورتيکه ورودیهاي و خروجيهاي متناظر موجود باشد (به اين زوج ورودي و خروجي مجموعه تعليم[۸] گفته ميشود) ، ميتوان وزنها را بر اساس فرمول $w^T = oi^{-1}$ بدست آورد.

ميتوانيم براي اينکه خروجي خاصي توليد شود از يک تابع به نام تابع فشرده سازي[۹] استفاده کنيم.اين مسئله در شکل٤ نشان داده شده است.



شکل ٤ : مدل نرون خطي به همراه تابع فشرده سازي

تابع فشرده سازي(فعال سازي) رنج وسيعي از مقادير ورودي را به مقدار خاصي نگاشت ميکند.به عنوان مثال

میتوانیم هرمقدار خروجي را به مقادیر باینري ۰ و ۱ نگاشت کنیم. انواع مختلفي از توابع فشرده سازي در *ANN* ها استفاده میشود. ولي بیشترین استفاده را تابع فشرده سازي سیگموید[۱۰] دارد.

$$O_i = \frac{1}{1+e^{-g_i}}$$

ادعا میشود که فرکانس آتش نرون طبیعي به صورت تابعي شبیه سیگموید میباشد. البته دلایل دیگري براي استفاده از سیگموید وجود دارد. از جمله :

فشرده سازي میکند. (رنج ورودي و خروجي *[۰،۱]* میباشد) *(-∞,+∞)*

تقریبا خطي ، افزایشي و مشتق پذیر است.

در فرم بسته قابل نمایش است.

مشتق گیري از آن ساده است.

کهاي تعیین پارامترهاي نرون خطي

کلي دو روش براي تعیین پارامترهاي نرون خطي یا به عبارت دیگر مشخص کردن مقادیر وزنها وجود دارد:

تعیین مستقیم

روالهاي تکرار شونده

۱ - تعیین مستقیم

فرض کنیم $\underline{i}\underline{w} = \underline{b}$ که b بردار پاسخهاي دلخواه میباشد.

بنابر این :

$$H = \{<i_1, b_1>, <i_2, b_2>, \Lambda, <i_d, b_d>\}$$

بدین صورت میتوان مقادیر w را بدست آورد.

$$\underline{i}\underline{w}^T = \underline{b} \Rightarrow \underline{w}^T = \underline{i}^T \underline{b}$$

۲ – روالهاي تکرار شونده

نام دیگر این روش "گرادیان کاهنده[۱۱]" میباشد در ادامه به توضیح این روش مي پردازیم :

قبل از هر چیز بردار گرادیان را تعریف میکنیم:

$$\frac{dF(\underline{x})}{d\underline{x}} = \begin{bmatrix} \frac{dF(\underline{x})}{dx_1} \\ \frac{dF(\underline{x})}{dx_2} \\ \vdots \\ \frac{dF(\underline{x})}{dx_d} \end{bmatrix} = \nabla_x \underline{F}$$

بردار گرادیان در واقع جهت تغییرات ماکزیمم در تابع f را نشان میدهد.

در روش کاهش گرادیان ، ابتدا مقادیر تصادفي براي وزنها انتخاب میشود و بر اساس این خروجیهایي از شبکه بدست مي آید.در یک روال تکرار شونده $\nabla_{\underline{w}} F$ محاسبه شده و در جهت آن حرکت خواهد شد.بدین معني که را طور تغییر میدهیم که اختلاف خروجي واقعي و خروجي مطلوب کمینه شود. حرکت در جهت متناظر با حرکت در جهت کاهش آن با شیب ماکزیمم میباشد.

خطاي شبکه به ازاء وزنهاي $\underline{w}_n$ برابر است با : $e_w = \underline{i}\underline{w}_n - b$

e_w در واقع همان تابعي است که باید کمینه شود.پس :

$$\underline{w}^{n+1} = \underline{w}^n - \alpha_n \frac{\partial e_{\underline{w}}}{\partial \underline{w}} = \underline{w}^n - \alpha_n \underline{i}$$

خطاي خروجي برابر است با اختلاف مابین خورجي شبکه (۰) و خروجي مطلوب (t).خطاي یک ورودي برابر است با :

و خطاي كلي برابر است با :

$$e^p = \frac{1}{2}(t^p - o^p)^2$$

خطاي ميانگين مربع[۱۲] برابر با :

$$E = \sum_{p=1}^{n} e^p$$

$$E = \frac{1}{n}\sum_{p=1}^{n}(t^p - o^p)$$

قاعده كلي ،بدست آوردن مشتق خطا نسبت به w ميباشد تا بتوانيم تغييرات خطا نسبت به w را به صفر نزديک كنيم.يعني ما بدنبال پيدا كردن $\frac{\partial(e^p)}{\partial \underline{w}^k}$ ميباشيم كه k مرتبه تكرار اين روال است .به عبارتي ديگر :

$$\frac{\partial(e^p)}{\partial \underline{w}^k} = \frac{\partial e}{\partial o^p}\cdot\frac{\partial o^p}{\partial g^p}\cdot\frac{\partial g^p}{\partial w}$$

در نتيجه تصحيح وزن jام در مرحله kام از تكرار الگوريتم تعليم به صورت زير ميباشد:

$$w_{K+1}^j = w_k^j - \alpha[(o^p - t^p)\frac{\partial F(g^p)}{\partial g^p} i_k^p]$$

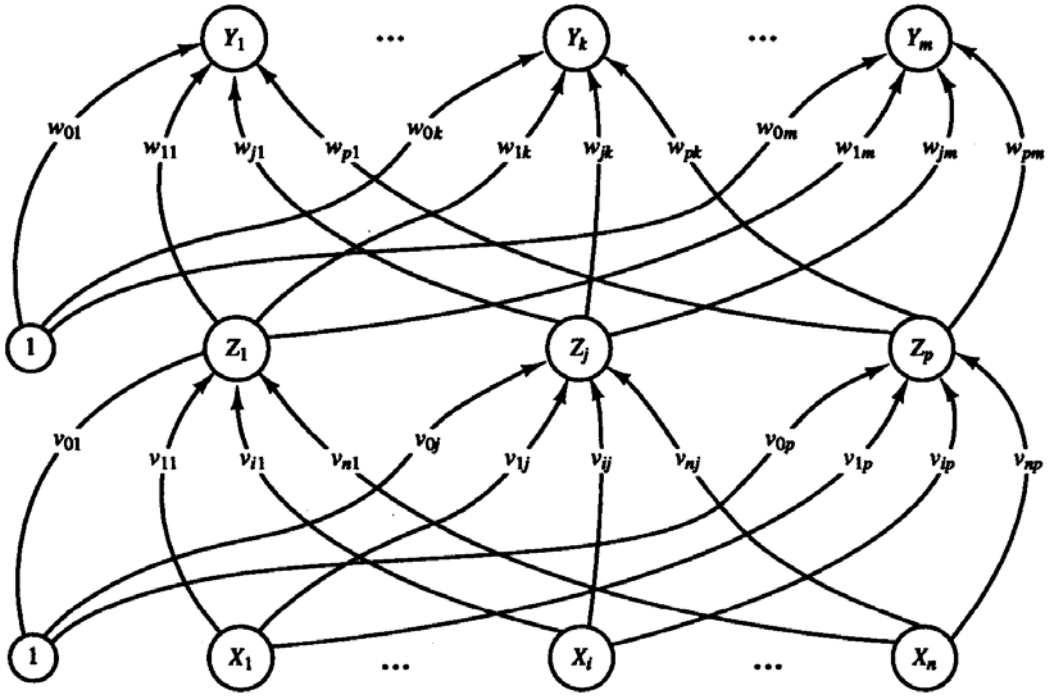
كه $\frac{\partial F(g^p)}{\partial g^p}$ ، مشتق تابع فشرده سازي (اكتيواسيون) و α، نرخ يادگيري ميباشد كه عموما مابين ۰ و ۱ انتخاب ميشود. بدين صورت ميتوان وزنهاي هر نرون را بدست آورد و روند تعليم شبكه را به اتمام رساند. پس از تعليم ، شبكه آماده توليد خروجي از ورودي متناظر ميباشد.

شبكه هاي پرسپترون[۱۳] چند لايه

شبكه هايي كه از يک نرون ساخته شده اند داراي محدوديتهايي ميباشند. اين شبكه ها توانايي پياده سازي توابع غير خطي را ندارند. به عنوان مثال توسط اين شبكه ها نميتوان تابع XOR را پياده سازي نمود. براي حل اين مشكل شبكه هاي ديگري پيشنهاد شد كه ما به يكي از آنان كه بيشترين استفاده را دارد مي پردازيم. دليل ديگر معرفي اين نوع شبكه ، استفاده از آن در ابن تحقيق مي باشد.

مدل عمومي شبكه هاي پرسپترون ، شبكه جلو رونده[۱٤] با روال تعليم انتشار به عقب[۱٥] است. شبكه هاي جلو رونده شبكه هايي هستند كه ورودیهاي لايه اول نرونهاي آن به لايه هاي بعدي متصل بوده و در هر سطح اين مسئله صادق بوده تا به لايه خروجي برسد. روال انتشار به عقب بدين معني است كه پس از مشخص شدن خروجي شبكه ، ابتدا وزنهاي لايه آخر تصحيح شده و بعد به ترتيب اوزان لايه هاي قبلي تصحيح ميشوند.در اين موارد بيشتر توضيح داده ميشود.

شبكه هاي پرسپترون از يک لايه ورودي ، تعدادي لايه پنهان و يک لايه خروجي تشكيل شده است.در شكل ٥ يک شبكه پرسپترون با يک لايه پنهان نشان داده شده است.



شکل ٥ : نمونه اي از یک شبکه پرسپترون

در این شکل لایه X لایه ورودي ، لایه Z لایه پنهان و لایه Y نیز لایه خروجي میباشد.

در این شبکه ها شرایط زیر وجود دارد :

نرونهاي هر لایه تنها به نرونهاي لایه بعدي متصل میباشند.

هر نرون به تمامي نرونهاي لایه بعد متصل است.

نرونهاي لایه ورودي عملي را انجام نمیدهند و اوزان آنها ثابت و برابر یک میباشد. این نرونها فاقد تابع فشرده سازي میباشند.

انتشار عملگر رو به جلو است. تمامي نرونها به غیر از لایه ورودي جمع کننده بوده و هر نرون میتواند تابع فشرده سازي مستقلي داشته باشد.

هر نرون میتواند داراي بایاس مستقل باشد.

تعداد لایه هاي پنهان مشخص نمیباشد.

ریتم یادگیري شبکه هاي پرسپترون (انتشار به عقب)

ن بخش به نحوه یادگیري شبکه پرسپترون چند لایه که در شکل ٥ نشان داده شده است مي پردازیم.

گوریتم یادگیري از علائم زیر استفاده میکنم:

ردار ورودي که براي یادگیري شبکه از آن استفاده میشود.

ردار خروجیهاي مطلوب

δ_k : درصدي از خطا که براي تصحیح وزن w_{jk} استفاده میشود و براساس خطاي واحد خروجي میباشد . این خطا در پنهان قبل از لایه خروجي انتشار به عقب مي یابد.

δ_j : درصدي از خطا که براي تصحیح وزن v_{ij} استفاده میشودو بر اساس انتشار به عقب اطلاعات خطا از لایه خروجي به و

پنهان Z_j میباشد.

خ یادگیري.

X_i : واحد ورودي iام.

v_{0j} : بایاس واحد پنهان jام.

Z_j : واحد پنهان jام.

ورودي شبكه به Z_j توسط z_in_j نشان داده ميشود:

$$z_in_j = v_{oj} + \sum_i x_i v_{ij}$$

سيگنال خروجي Z_j توسط z_j مشخص ميشود:

$$z_j = f(z_in_j)$$

w_{0k} : بايَاس واحد خروجي kام.

امk: واحد خروجي

ورودي Y_k توسط y_in_k مشخص ميشود:

$$y_in_k = w_{0k} + \sum_j z_j w_{jk}$$

سيگنال خروجي Y_k توسط y_k مشخص ميشود :

$$y_k = f(y_in_k)$$

براي آموزش شبكه توسط انتشار به عقب ، سه مرحله بايد انجام شود : انتشار به جلو الگوهاي يادگيري ورودي ، انتشار به عقب خطاي بدست آمده و تنظيم اوزان.

در حين انتشار به جلو ، هر واحد ورودي X_i سيگنال ورودي خودرا در هركدام از واحدهاي پنهان توزيع ميكند.سپس هركدام از واحدهاي پنهان اكتيواسيون خودرا محاسبه كرده و سيگنال حاصل را به هركدام از واحدهاي خروجي ارسال ميكند.در نهايت هركدام از واحدهاي خروجي Y_k با محاسبه اكتيواسيون خود ، را به عنوان پاسخ شبكه به الگوي وردي ، ارائه ميكند.

درمرحله يادگيري ،در هركدام از واحدهاي خروجي ، y_k توليد شده را با مقايسه كرده تا خطاي مربوطه را پيدا كنيم. بر اساس اين خطا ، فاكتور $\delta_k (k = 1,...,m)$ محاسبه ميشود.از براي توزيع به عقب خطا در تمامي واحدهاي لايه قبل استفاده ميشود. همچنين از آن براي بروزرساني اوزان ميان واحدهاي لايه پنهان و لايه خروجي نيز استفاده ميشود. به همين صورت نيز δ_j براي تک تک واحدهاي پنهان Z_j محاسبه ميشود. البته انتشار به عقب خطا در لايه ورودي لازم نيست، ولي δ_j براي بروزرساني اوزان ميان لايه ورودي و لايه پنهان استفاده خواهدشد.

در نهايت زمانيكه تمامي δ مشخص شدند ، تمامي اوزان به صورت هم زمان تصحيح خواهند شد. تصحيح وزن (از لايه پنهان به لايه خروجي) بر اساس δ_k و z_j خواهد بود و تصحيح وزن (از لايه ورودي به لايه پنهان) بر اساس و ميباشد.

حال بر اساس مطالب گفته شده الگوريتم يادگيري را شرح ميدهيم :

مرحله ۰ : مقدار دهي اوليه اوزان بر اساس مقادير تصادفي كوچک

مرحله ۱ : تازمانيكه شرط پايان برقرار نيست مراحل ۲ تا ۹ را تكرار كنيد.(در مورد شرط پايان صحبت خواهد شد)

مرحله ۲ : به ازاء هر زوج تعليم (ورودي و خروجي متناظر در مجموعه تعليم) مراحل ۳ تا ۸ را تكرار كنيد :

مرحله ۳ : هر واحد ورودي

X_i *,i=1,...,n*

سيگنال ورودي x_i را دريافت كرده و آنرا در تمامي واحدهاي لايه بعد از خويش توزيع ميكند.

: هر واحد پنهان

1,...,p

حاصل جمع سيگنالهاي ورودي وزن دار خويش را محاسبه ميكند :

سپس تابع فشرده سازي مربوطه را بر حاصل اعمال ميكند :

$$z_in_j = v_{oj} + \sum_i x_i v_{ij}$$

و سيگنال حاصل را به تمامي واحدهاي لايه بعدي ارسال ميكند.

$$z_j = f(z_in_j)$$

مرحله ٥ : هر واحد خروجي

حاصل جمع سيگنالهاي ورودي وزن دار خويش را بدست مي آورد:

Y_k , *k=1,...,m*

$$y_in_k = w_{0k} + \sum z_j w_{jk}$$

و تابع فشرده سازي مربوط به خود را براي محاسبه خروجي به حاصل اعمال ميكند:

$$y_k = f(y_in_k)$$

<u>انتشار به عقب خطا :</u>

مرحله ٦ : هر واحد خروجي

Y_k , *k=1,...,m*

δ_k را بر اساس خروجي مطلوب و خروجي بدست آمده محاسبه ميكند :

$$\delta_k = (t_k - y_k) f'(y_in_k)$$

و مقداري را كه بعدا براي تصحيح وزن w_{jk} و بايَاس w_{0k} نياز است ، بدست مي آورد :

$$\Delta w_{jk} = \alpha \delta_k z_j$$
$$\Delta w_{0k} = \alpha \delta_k$$

و δ_k را به واحدهاي لايه قبل از خود ارسال ميكند.

مرحله ٧ : هر كدام از واحدهاي پنهان

1,...,p

حاصل جمع دلتاي ورودي خود را كه از واحدهاي لايه بعد از خود به آن رسيده است ، محاسبه ميكند:

$$= \sum_{k=1}^{m} \delta_k w_{jk}$$

و حاصل را در مشتق تابع فشرده سازي ضرب كرده تا ترم اطلاعات خطاي بدست آيد:

$$_in_j f'(z_in_j)$$

سپس مقداري را كه بعدا براي تصحيح وزن v_{ij} و بايَاس v_{0j} نياز است ، بدست مي آورد :

$$\chi \delta_j x_i$$

$$\alpha \delta_j$$

<u>تصحيح اوزان و بايَاسها :</u>

: هر كدام از واحدهاي خروجي وزنها (*j=0,...,p*) و بايَاس خودرا به صورت زير تصحيح ميكنند :

$$w) = w_{jk}(old) + \Delta w_{jk}$$

هر كدام از لايه هاي پنهان نيز وزنها (*i=0,...,n*) و بايَاس مربوطه را به صورت زير تصحيح ميكند :

$$v_{ij}(new) = v_{ij}(old) + \Delta v_{ij}$$

مرحله ٩ : شرط پايان برقرار است. پايان الگوريتم يادگيري.

البته در زمان پياده سازي بايد آرايه هاي مجزايي براي دلتاهاي واحدهاي خروجي (مرحله ٦ ،) و واحدهاي پنهان (مرحله ٧ ، δ_j) تشكيل شود.

يک دوره[16] ، چرخه اي كامل حول مجموعه يادگيري است.تعداد زيادي چرخه براي آموزش يک شبكه عصبي انتشار به عقب لازم است. معمولا چرخه آموزش را تا زماني كه ميانگين كل خطا به يک مقدار حداقل مطلوب يا صفر برسد ادامه مي دهند (شرط پايان) . در برخي مواقع مقدار خطاي ميانگين در طي چند دوره تغييري نميكند . در اين صورت نيز يادگيري شبكه عصبي پايان مي يابد.

عموما مقدار دهي اوليه اوزان و بايَاسها به صورت تصادفي با مقادير كوچک انجام ميشود. ثابت شده است كه انتخاب هر مقدار اوليه براي وزنها و بايَاسها به سمت مقدار صحيح آنها همگرا است. بدين معني كه هر مقداري براي آنها انتخاب كنيم ، پارامترهاي شبكه تنظيم خواهند شد. تنها تفاوتي كه وجود دارد ، در صورت فاصله زياد مقادير اوليه وزنها و بايَاسها با مقادير صحيح آنها ، براي رسيدن به مقادير مطلوب تعداد دوره ها زياد خواهد بود . *Nguyen-Widrow[1990]* روشي را براي مقدار دهي اوليه اوزان و بايَاسها پيشنهاد داده اند كه باعث ميشود وزنها با فاصله كمي نسبت به مقدار صحيح آنها انتخاب شوند . توضيح اين روش در حوصله اين مقوله نيست و علاقه مندان ميتوانند با مراجعه به منابع شبكه هاي عصبي مصنوعي اين روش را بياموزند.

در زمينه تعيين تعداد الگوهاي يادگيري لازم براي تعليم شبكه روشي وجود دارد. اگر *P* تعداد الگوهاي يادگيري ، *W* تعداد وزنهاي شبكه و حداكثر خطاي مجاز شبكه *e* باشد ، بدين منظور كه شبكه به صورت صحيح عمل كند و بتواند عمل دسته بندي را با دقت *1-e* انجام دهد ، بايد رابطه زير برقرار باشد :

$$\frac{W}{P} = e \Rightarrow P = \frac{W}{e}$$

به عنوان مثال در صورتيكه *e=0.1* و *W=80* باشد ، ما به *(P=800)* ٨٠٠ الگوي يادگيري نيازمنديم تا مطمئن شويم كه شبكه عصبي مصنوعي طراحي شده در ٩٠٪ مواقع به صورت صحيح عمل ميكند. اثبات اين مسئله توسط *Baum* و *Haussler* در سال ١٩٨٩ انجام شد.

در شبكه هاي پرسپترون چند لايه ، تعداد لايه هاي پنهان ميتواند هر تعداد باشد. البته در بيشتر كاربردها يک لايه پنهان كفايت ميكند . در برخي مواقع نيز دولايه پنهان يادگيري شبكه را ساده تر ميكند. در حالتي كه تعداد لايه هاي پنهان بيش از يک لايه باشد ، بايد الگوريتم ذكر شده در بالا را براي تمامي لايه ها تعميم داد.

روشي عملي براي تخمين تعداد واحدها (نرونها) در هر لايه پنهان در دست نيست. بدين منظور بايد از روشهاي سعي و خطا استفاده كرد تا به مقدار ميانگين خطاي كل مطلوب رسيد.

براي مدلسازي شبكه هاي عصبي مصنوعي ابزاهاي زيادي وجود دارد . از اين جمله:

SNN (Stuttgart Neural Network Simulator)

Matlab standard NN Toolbox

Netlab toolbox for Matlab

به عنوان نمونه ميتوانيد از *Netlab* كه يک تول باكس مجاني براي نرم افزار *Matlab* ميباشد استفاده كنيد. براي استفاده از آن ميتوانيد به آدرس زير رجوع كنيد:

http://www.ncrg.aston.ac.uk/netlab/index.php

پانوشتها

[1] Artificial Neural Network
[2] Model Free
[3] Neuro-synaptic
[4] Activation function
[5] Squashing
[6] Local
[7] Training
[8] Training set

[9] Squashing
[10] Sigmoid
[11] Gradient descent
[12] Mean Square Error(MSE)
[13] Perseptron
[14] Feed forward
[15] Back propagation training
[16] Epoch

فهرست منابع

1 – Fauset,L.(1994). **Fundamentals of Neural Networks : Architectures , Algorithms and Applications** . Prentice Hall

2 – Schalkoff ,R,J.(1997). **Artificial Neural Networks** . McGraw-Hill

3 – Krose,B ,Smagt,P(1196) .**An Introduction to Neural Networks** . University of Amesterdam

۴ – منهاج ، محمد باقر (۱۳۸۱). **مباني شبكه هاي عصبي** .دانشگاه صنعتي امير كبير