فصل 11

سیستمهای نروفازی

در این فصل به بررسی مکانیزم شبکه‌های عصبی ترکیب‌شده با فازی یا اصطلاحاً *نروفازی* می‌پردازیم که ساختاری متداول برای حل مسائل مختلف است. همان‌طور که از نام این ساختار نیز مشخص است، این شبکه‌ها سعی در استفاده از نقاط قوت *شبکه‌های عصبی مصنوعی*[1] و سیستمهای فازی در ترکیب آنها با یکدیگر دارند.

اگر به دانش به صورت قوانین زبانی دسترسی داشته باشیم، می‌توانیم از آن برای ایجاد یک *واسط فازی*[2] استفاده کنیم. همچنین اگر داده‌های مناسبی داشته باشیم که توسط فرآیند شبیه‌سازی یادگیری، برای آموزش یک سیستم استفاده شوند، آنگاه می‌توان این داده‌ها را به شبکه‌های عصبی مصنوعی اعمال نمود.

برای ساختن یک FIS باید مجموعه‌های فازی، اپراتورهای فازی و پایگاه دانش خود را مشخص کنیم. همچنین برای ساختن یک شبکه عصبی، باید معماری و الگوریتم یادگیری آن توسط کاربر تعیین گردد. یک تحلیل ساده نشان می‌دهد که هرکدام از این دو سیستم به تنهایی ضعفهایی دارند که ترکیب آنها با یکدیگر، باعث رفع این ضعفها خواهد شد. چرا که قابلیت یادگیری، یک ارتقاء برای FIS محسوب می‌شود و فهم ساختار جملات طبیعی زبان یک مزیت برای ANN خواهد بود.

در این فصل، ابتدا مرور کوتاهی بر اصول *تشخیص آماری الگو*[3] و انواع شبکه‌های عصبی خواهیم داشت. سپس دو حالت کلی را برای سیستمهای نروفازی مورد بررسی قرار می‌دهیم که حالت اول به نروفازی *همکارگونه*[4] و سیستمهای نروفازی *همزمان*[5] و حالت دوم به نروفازی‌های *فیوزشده*[6] می‌پردازد. در نهایت، این فصل را با ارائه یک نمونه عملی از کاربرد سیستمهای نروفازی به پایان می‌رسانیم.

---

[1] Artificial Neural Network (ANN)
[2] Fuzzy Interface Systems (FIS)
[3] Statistical pattern recognition
[4] Cooperative
[5] Concurrent
[6] Fused

11-1: اصول تشخیص آماری الگو و تخمین توابع

هدف از این بخش، معرفی مفاهیم مورد استفاده در این فصل در حد تعاریف ساده و ابتدایی می‌باشد. از جمله این مفاهیم می‌توان از مجموعه داده‌های آموزش، داده‌های تست، سیستمهای یادگیری، بردار ویژگی و تفکیک‌کننده‌ها نام برد. خوانندگانی که با این مفاهیم آشنایی دارند می‌توانند با مرور سریع این قسمت به مطالعه بخشهای بعد بپردازند.

فرض کنید می‌خواهیم دو نوع ماهی مثلاً ماهی سفید و ماهی قزل‌آلا را به صورت خودکار تشخیص دهیم. در این صورت باید تعدادی ویژگی قابل فهم برای ماشین را از این ماهیها انتخاب کرده و ماشین را از این خصوصیات (مثلاً خصوصیاتی چون قد ماهی، میزان روشنایی پوست ماهی، ...) آگاه کنیم. خصوصیات مطرح‌شده باید امکان تشخیص و تفکیک نوع خاصی از ماهی را با ضریب اطمینان مناسبی دارا باشند.

اگر فرض کنیم که ماهی سفید طولی حداقل برابر با 0.5 متر و ماهی قزل‌آلا طولی حداکثر برابر با 0.3 متر داشته باشد، آنگاه طول ماهی ویژگی ایده‌آلی است که می‌تواند به راحتی با یک شرط ساده "if ➔ then" نوع ماهی را تعیین کند.

متاسفانه در اکثر مسائل دنیای واقعی، ویژگی ساده‌ای که قدرت تفکیک بالایی نیز داشته باشد، وجود ندارد و باید برای تشخیص دو نوع الگوی متفاوت (در اینجا ماهی سفید یک الگو و ماهی قزل‌آلا الگویی دیگر محسوب می‌شود) تعداد ویژگیهای بیشتری را به صورت *بردار ویژگیها*[1] انتخاب نمود.

مسئله مهم دیگری نیز مطرح است. چگونه متوسط طول را برای ماهیهای قزل‌آلا و ماهیهای سفید به دست آوریم؟ قطعاً راهکار ما در اینجا به یک نمونه‌برداری آماری منجر خواهد شد. بدین ترتیب علت کاربرد روشهای آمارگیری در تشخیص الگو معلوم می‌شود.

در تشخیص آماری الگو، ابتدا تعدادی نمونه از هر الگو انتخاب شده، سپس بردار ویژگیها از نمونه‌های انتخاب‌شده استخراج می‌شود. بردارهای استخراج‌شده به دو دسته تقسیم می‌شوند.

---

[1] Features vector

دسته اول که شامل قسمت عمده‌ای از بردارها می‌شود را مجموعه آموزش می‌نامند که از این دسته برای آموزش *تفکیک‌کننده*[1] خود استفاده می‌کنیم. تفکیک‌کننده‌ها دارای انواع متفاوت و الگوریتمهای گوناگونی هستند که به عنوان یک مورد شاخص از آنان، می‌توان به شبکه‌های عصبی مصنوعی اشاره نمود.

دسته دوم که به نام مجموعه تست خوانده می‌شود، وظیفه ارزیابی و بررسی صحت عملکرد تفکیک‌کننده را به عهده دارد و به عنوان ورودی به تفکیک‌کننده وارد می‌شود تا خروجی تفکیک‌کننده با خروجی واقعی (آنچه که انتظار داریم تا در خروجی دیده شود) مقایسه گردد. بنابراین، صحت عملکرد تفکیک‌کننده در مقایسه خروجیها با یکدیگر تعیین خواهد شد. ذکر این نکته ضروری است که تعاریف مربوطه در مباحث تخمین[2] نیز، برای مجموعه تست و آموزش به همین صورت مطرح می‌شوند.

حال با دانستن مفاهیم اساسی درباره مجموعه آموزش و تست به عنوان ورودیهای یک شبکه عصبی، به توضیح در مورد عملکرد این تفکیک‌کننده‌ها و نیز معرفی مختصری از انواع آنها می‌پردازیم.

11-2: شبکه‌های عصبی مصنوعی

شبکه‌های عصبی مصنوعی با اقتباس از شبکه‌های عصبی مغز انسانها طراحی شده‌اند. هرکدام از ساختارهای متفاوت این شبکه‌ها برای یک کاربرد خاص مناسب است. بنابراین هر شبکه الزاماً دارای کارایی مناسبی در تمامی کاربردها نخواهد بود.

این شبکه‌ها به دلیل دارا بودن خصوصیات مهمی چون توانایی کار با داده‌های غیرمطمئن، عدم نیاز به حافظه فراوان و نیز تاثیرپذیری ناچیز در برابر خطاهای موجود در داده ورودی، از تفکیک‌کننده‌ها و تخمین‌زننده‌های محبوب به شمار می‌روند.

---

[1] Classifier
[2] Approximation

در این بخش اساس کار نرونها به عنوان عنصر اصلی اکثر شبکه‌های عصبی بیان می‌شود و توضیحاتی در مورد شبکه‌های پرسپترون به عنوان پرکاربردترین شبکه عصبی مصنوعی ارائه خواهیم کرد.

یک شبکه عصبی به طور سنتی شبیه یک مغز کامپیوتری مینیاتوری تصور می‌شود که شامل سلولهای مستقلی است که در ارتباط با یکدیگر می‌توانند مسائل بزرگ و پیچیده‌ای را حل کنند که حل آنها برای یک سلول مجزا، مشکل یا غیرممکن است.

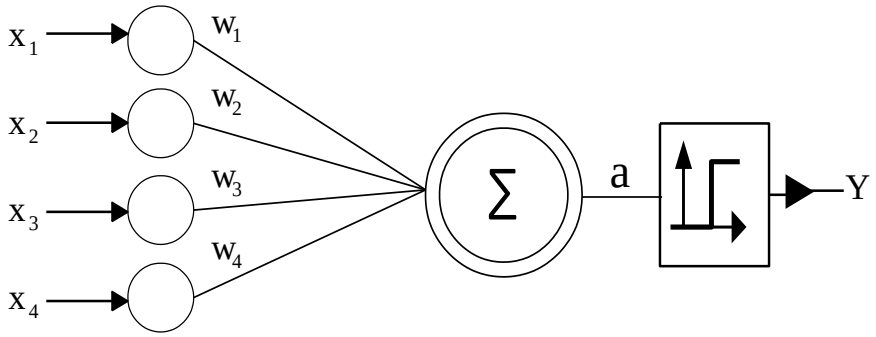
یک شبکه عصبی از نرونها یا گره‌ها ساخته می‌شود که در حکم واحدهای پردازشی ساده هستند. اساساً گره از دید برنامه‌نویسی یک کلاس محسوب می‌شود که وظیفه انجام کار یا رسیدن به هدف خاصی را بر عهده دارد. این وظیفه یا هدف براساس موردی است که برنامه‌نویس تعیین می‌کند و از طریق کدها تعریف می‌شود.

11-2-1: مدل یک نرون

در این مدل، اجزاء اصلی سازنده نرون در یک دید کلی عبارتند از:

- سیناپس
- بدنه سلول
- خروجی

مدل کلی یک نرون در شکل11-1 به نمایش درآمده است:



شکل11-1: مدل کلی یک نرون

همانطورکه در شکل 11-1 نیز مشخص است، سیناپسها با وزنهای مختلف (w_i) مدل شده‌اند تا بیانگر ارزش عملی ورودیهای مختلف باشند.

بدنه سلول که با جمع‌کننده (Adder) نشان داده شده است، دارای عملکرد ساده‌ای مطابق فرمول زیر است:

$$a = w_1 * x_1 + ... + w_n * x_n$$

تابع Hard limiter نقش صادرکننده پالس الکتریکی یا عنصر تصمیم‌گیرنده را در یک شبکه عصبی ایفا می‌کند که در ادبیات این حوزه به تابع فعال‌ساز[1] معروف است. ورودی این تابع از خروجی جمع‌کننده خواهد آمد.

خروجی سلول توسط تابع فعال‌ساز (با توجه به نوع تابع استفاده‌شده) تعیین می‌شود. عملکرد کلی سلول عصبی مطابق فرمول زیر خواهد بود:

$$Y = 1 \quad if \quad a \geq T$$
$$Y = 0 \quad if \quad a < T$$

بدین معنا که در صورت بزرگتر بودن خروجی از یک مقدار آستانه[2]، سلول عصبی فعال می‌شود و دارای مقدار خروجی یک خواهد بود و در غیر این صورت، مقدار خروجی برابر با صفر می‌باشد.

درحالت کلی، وزنها ممکن است بجای اینکه مستقیماً روی خود ورودیها به کار بروند، روی پردازش یافته آنها اعمال بشوند.

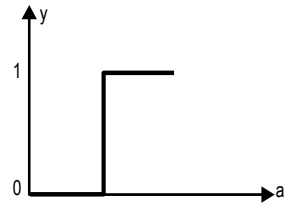
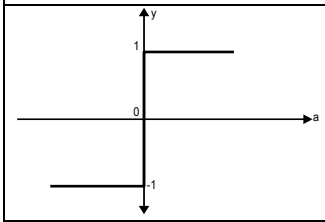
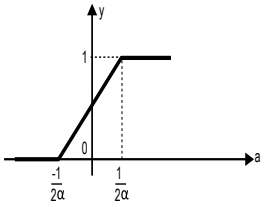
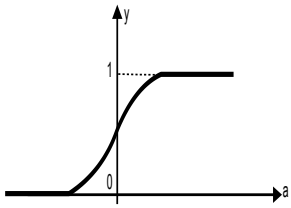
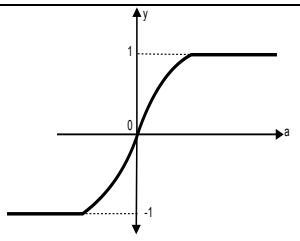
همچنین ممکن است در یک تابع فعال‌ساز، روی مجموعه وزنها محدودیتهای خاصی اعمال شود (مثلاً جمع آنها حتماً برابر با یک باشد) تا ایجاد رقابت کند. تعدادی از توابع فعال‌ساز متداول به همراه شکل و فرمول نظیر آنها در جدول 11-1 آورده شده‌اند.

همان‌طورکه در جدول نیز مشخص است، تابع فعال‌ساز می‌تواند یک تابع پله‌ای (دارای نقطه انفصال) یا متصل و مشتق‌پذیر باشد. پیوستگی و مشتق‌پذیری یک تابع فعال‌ساز برای ملاحظات محاسباتی دارای مزایای متعددی می‌باشد.

---

[1] Activation function
[2] Threshold

جدول 11-1: توابع فعال‌ساز متداول

نام تابع فعال‌ساز	فرمول مشخصه تابع فعال‌ساز	شکل تابع فعال‌ساز
پله‌ای تک‌قطبی	$y=\varphi(a)=\begin{cases}1 & if \quad a\geq 0\\ 0 & if \quad a<0\end{cases}$	
پله‌ای دوقطبی	$y=\varphi(a)=\begin{cases}+1 & if \quad a\geq 0\\ -1 & if \quad a<0\end{cases}$	
خطی	$y=\varphi(a)=\begin{cases}0 & if \quad a\leq -\frac{1}{2\alpha}\\ \alpha a+\frac{1}{2} & if \quad \lvert a\rvert <\frac{1}{2\alpha}\\ 1 & if \quad a\geq \frac{1}{2\alpha}\end{cases}$	
سیگموید تک‌قطبی (لگاریتمی، نمایی)	$\varphi(a)=\frac{1}{1+e^{-a}}=\frac{1}{2}\left(tanh(a/2)-1\right)$	
سیگموید دوقطبی (لگاریتمی، نمایی)	$\varphi(a)=tanh(\beta a)$ ضریب β میزان شیب منحنی است	

برای هر نوع خاص از شبکه‌های عصبی مصنوعی، باید سیاستها و قواعدی را برای افزایش کارایی در عملکرد شبکه تعیین کنیم. این قواعد شامل موارد زیر است:

- **قاعده‌ای برای انتشار سیگنالها در شبکه:** قاعده انتشار در یک شبکه متشکل از نرونها، حاصل‌ضرب مقدار هر ورودی در سیناپس (وزن) نظیر آن است.
- **قواعدی برای ترکیب سیگنالهای ورودی**
- **قاعده‌ای برای تجمیع یک سیگنال خروجی:** معمولاً سیگنالهای ورودی توسط جمع‌کننده با یکدیگر جمع می‌شوند، اما می‌توان بجای جمع از تابع "و منطقی"یا تابع "یای منطقی" یا هر تابع دیگری برحسب نیاز استفاده کرد. این خروجی لزوماً خروجی نهایی برنامه نیست، بلکه خروجی بخش خاصی از کدها می‌باشد. اگر به این خروجی از دید تابع نگاه کنیم، خروجی یک گره شبکه در واقع بیانگر پارامتر بازگشتی از تابع است که معمولاً به صورت یک مقدار عددی بیان می‌شود.
- **یک قاعده یادگیری برای تغییر وزنها:** در فرایند یادگیری شبکه، براساس خطایی که در پاسخ شبکه به داده‌های آموزش (نسبت به نتایجی که انتظار داریم وجود داشته باشد) دیده می‌شود، شبکه واکنش داده و وزنها را تغییر می‌دهد. نوع به‌روزرسانی وزنها منجر به آموزش شبکه و بهتر شدن پاسخ تا رسیدن به میزان خطای قابل‌قبول خواهد شد.

در ادامه مطالب، شبکه پرسپترون را به عنوان رایج‌ترین و پرکاربردترین شبکه عصبی مصنوعی مورد بررسی قرار می‌دهیم.

11-2-2: شبکه عصبی پرسپترون

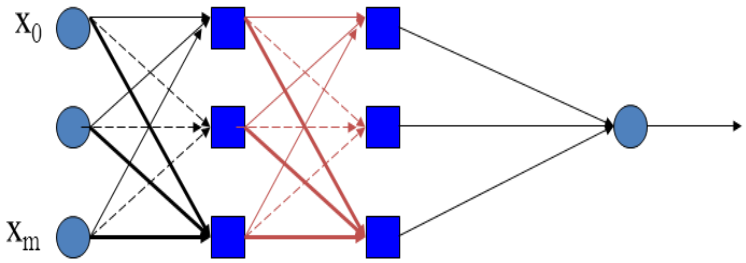
برای آشنایی با شبکه‌های پرسپترون، ابتدا چند مفهوم مورد استفاده در این شبکه‌ها را معرفی می‌کنیم:

- **لایه**[1]: گروهی از نرونها که نسبت به سیگنالهای ورودی و خروجی شبکه در موقعیت مشابهی قرار دارند را یک لایه می‌نامند.
- **شبکه‌های روبه‌جلو**[1]: در ساختار گراف این شبکه‌ها (برخلاف گراف شبکه‌های بازگشتی[2]) هیچ‌گونه حلقه یا دوری مشاهده نمی‌شود.

[1] Layer

- **شبکه‌های یادگیری با سرپرست**[3]: این نوع شبکه‌ها خروجی واقعی را به ازای تعدادی از داده‌های ورودی به دست می‌دهد. در واقع برای ما روشن است که اگر شبکه به درستی کار کند خروجی برای مجموعه آموزش باید چگونه باشد. در بعضی موارد که خروجی به وضوح مشخص نشده است یا در کاربردهایی که خوشه‌بندی و تقسیم‌بندی به خود شبکه (برای رسیدن به یک حالت بهینه) واگذار می‌شود، شبکه از نوع *فراگیری بدون سرپرست*[4] می‌باشد.

با این توضیحات، شبکه‌های پرسپترون از نوع شبکه‌های روبه‌جلو و نیز یادگیری با سرپرست محسوب می‌شوند و به طور کلی دارای معماری شبیه به شکل 2-11 هستند:



شکل 2-11: معماری یک شبکه پرسپترون

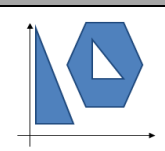
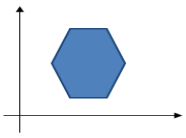
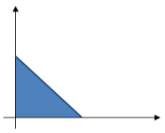
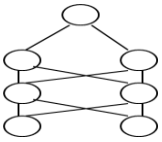
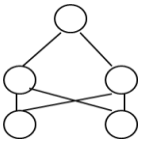
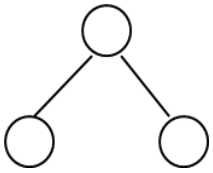
پرسپترون تک‌لایه عملاً یک صفحه تفکیک‌کننده طبقات را ایجاد می‌کند. تئوری شبکه پرسپترون چندلایه با استفاده از این مطلب بیان می‌شود که هر لایه باعث ایجاد یک خط می‌گردد و لایه‌های بعدی با ترکیب خطوط، فضا را طبقه‌بندی می‌کنند. در واقع، یک شبکه پرسپترون سه‌لایه قادر است شکل پیچیده‌ای را در فضا ایجاد کند.

---

[1] Feed forward
[2] Recursive
[3] Supervised
[4] Unsupervised

قضیه کولموگروف[1] بیان می‌کند که سه لایه برای شبکه‌های عصبی کافی است و هر شبکه عصبی با تعداد لایه‌های بیشتر را می‌توان با یک شبکه سه‌لایه‌ای مدل کرد. ویژگی شبکه پرسپترون با توجه به تعداد لایه‌ها در جدول 11-2 آمده است.

جدول11-2: نواحی مشخص‌شده توسط شبکه‌های عصبی یک، دو و سه لایه‌ای

ناحیه دلخواه	ناحیه محدب بسته	خطوط
		
		
شبکه سه‌لایه و بالاتر	شبکه دولایه	شبکه تک‌لایه

11-2-2-1: الگوریتم یادگیری یک شبکه MLP[2]

در الگوریتم یادگیری برای یک شبکه پرسپترون چندلایه بدین نحو عمل می‌کنیم که ابتدا، آستانه و ضرایب اولیه وزنی را مقادیری تصادفی و کوچک انتخاب کرده، ورودیها و خروجیهای مطلوب را به شبکه مطابق زیر، عرضه می‌نماییم:

$$T_p = t_0, t_1, \ldots, t_{m-1} \quad \text{و} \quad X_p = x_0, x_1, \ldots, x_{n-1}$$

n بیانگر تعداد عناصر بردار ورودی و m بیانگر تعداد عناصر بردار خروجی است. برای اعمال بایاس به شبکه، ضریب وزنی را به صورت $w_0 = -\theta$ و $x_0 = 1$ قرار می‌دهیم.

[1] Kolmogrov
[2] Multi Layer Percepteron (MLP)

تابع خروجی را می‌توان برای تمام لایه‌ها یکسان یا متفاوت در نظر گرفت. روشن است که تابع خروجی، نباید یک تابع ثابت باشد، چون در این صورت نمی‌تواند نشان‌دهنده رفتار لایه‌های قبلی شبکه باشد.

به عنوان متداول‌ترین تابع خروجی، از تابع سیگموئید $f(net)=\frac{1}{1-e^{-knet}}$ یا تابع خطی (مثالهایی از تابع مشتق‌پذیر) استفاده می‌شود.

مرحله اول: تغذیه شبکه با ورودیها انجام می‌شود که الگوریتم این مرحله برای یک تابع سیگموئید مطابق زیر است:

$$z_0^{(l-1)}(k)\equiv 1$$

برای تمام نمونه‌های آموزش از k=1 تا K (تعداد نمونه‌ها برابر با K است)

از لایه l=1 تا L (تعداد لایه‌ها برابر با L است)

و هر نود i در لایه l، محاسبات زیر را انجام دهد:

$$z_i^{(l)}(k)= f\left(u_i^{(l)}(k)\right)=1\Big/\left(1+exp\left[-u_i^{(l)}(k)\right]\right)$$

$$u_i^{(l)}(k)=\sum^{N} w_{ij}^{(l)}(t)\, z_j^{l-1}(k)$$

مرحله دوم: *انتشار به عقب*[1] برای خطا نام دارد که در آن به صورت زیر عمل می‌شود:

برای تمام نمونه های آموزش از k=1 تا K

از لایه l=1 تا L

و هر نود i در لایه l، محاسبات زیر را انجام دهد:

$$\delta_i^{(l)}(k)=\begin{cases} f'\left(u_i^{(t)}(k)\right).\sum_{m=1}^{N(l+1)} \delta_m^{(l+1)}(k).w_m^{(l+1)}(t) & l<L, \\ f'\left(u_i^{(t)}(k).\left[d_i(k)-z_i^{(L)}(k)\right]\right) & l=L \end{cases}$$

[1] Back Propagation (BP)

مرحله سوم: در آخرین مرحله، وزنها مطابق الگوریتم زیر به‌روزرسانی می‌شوند:

برای تمام نمونه های آموزش از k=1 تا K

از لایه l=1 تا L

و هر نود i در لایه l، محاسبات زیر را انجام دهد:

$$-\frac{\partial E}{\partial w_{ij}^{(l)}(t)} = \sum_{k=1}^{K} \delta_i^{(l)}(k)\, z_j^{(l-1)}(k)$$

$$w_{ij}^{(l)}(t+1) = w_{ij}^{(l)}(t) - \mu \frac{\partial E}{\partial w_{ij}^{(l)}(t)} + \mu\left(w_{ij}^{(l)}(t) - w_{ij}^{(l)}(t-1)\right)$$

انتشار خطا به عقب از دیدگاه ماتریسی نیز قابل بررسی است که در آن:

$$Z(l), l = 0,1, \ldots, L : N(l)' K$$

اگر $l = 0$ آنگاه Z بیانگر بردار ویژگیها است.

اگر $l > 0$ آنگاه iامین عنصر خروجی نرون در kامین نمونه آموزش در لایه ℓ ام است.

($N(l)' K$ بیانگر ابعاد ماتریس است.)

ماتریس وزنها به صورت زیر بیان می‌شود:

$$W(l), l = 1, 2, \ldots, L : N(l)'\left(N(l-1)+1\right)$$

هر عنصر (i,j)در $W(l)$با $W_{ij}^{(l)}$ نمایش داده می‌شود. شاخص ستونها در این ماتریس، از صفر شروع خواهد شد تا $W_{i0}^{(l)}$ نشان‌دهنده بایاس ورودی iامین نرون در لایه l باشد.

تابع شبکه $f(\cdot)$ که بر روی هر عنصر شبکه اعمال می‌شود، در شکل ماتریسی به صورت زیر نوشته خواهد شد:

$$U(l+1) = w(l).\begin{bmatrix} 1 & \ldots & 1 \\ & Z(l) & \end{bmatrix}, \quad and \quad Z(l) = f\left(U(l)\right)$$

دلتای خطا برای D (ماتریس خروجیهای مورد انتظار) با استفاده از مشتق تابع شبکه، مطابق فرمول زیر حاصل خواهد شد:

$$d(L) = f'\left(U(L)\right).*\left(D - Z(L)\right) : N(L)' K$$

باید دقت داشت که علامت * بیانگر ضرب درایه‌های نظیر به نظیر دو ماتریس در یکدیگر است که نباید آنرا با ضرب معمولی دو ماتریس اشتباه گرفت.

$$d(l) = f'(U(l)).*\left[W(l+1)(:,2:N(l+1)+1)\right]^T d(l+1):N(l)\ K$$

که $W(l+1)(:,2:N(l+1)+1)$ انتهایی‌ترین $N(l+1)$ ستون از ماتریس $W(l+1)$ و $[\ ]^T$ بیانگر ترانهاده یک ماتریس است. همچنین خواهیم داشت:

$$\Delta W(l,t) = \eta.\delta(l)\begin{bmatrix}1\\ :\quad Z^T(l)\\ 1\end{bmatrix} + \mu.\Delta W(l,t-1) + \varepsilon;$$

$$W(l,t+1) = W(l,t) + \Delta W(l,t)$$

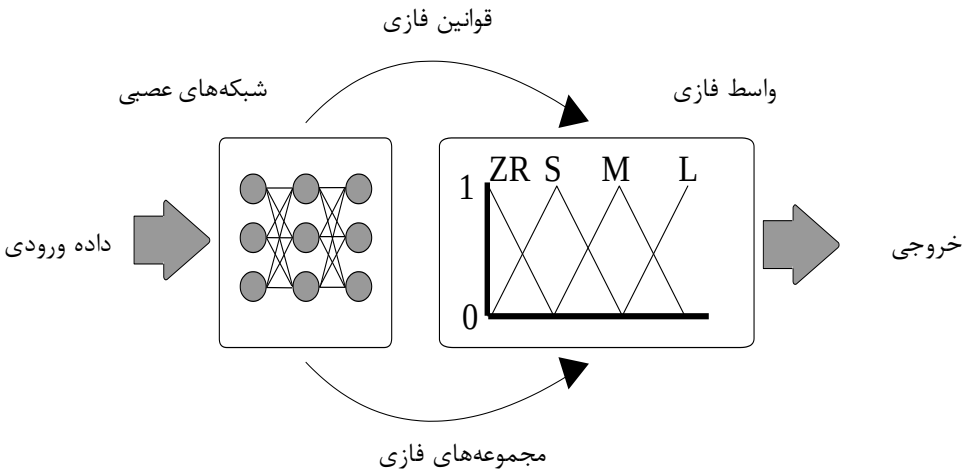
ε بیانگر نویزهای موجود است.

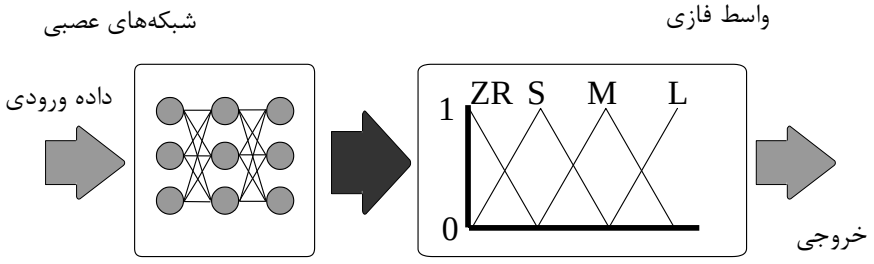
شبکه‌های عصبی دیگری نیز وجود دارند که ممکن است در کاربردهای خاصی بهتر و موثرتر از پرسپترونها عمل کنند. به عنوان نمونه می‌توان از شبکه‌هایRBF یا Hopfield نام برد که استفاده از آنها به نوع مسئله، داده‌ها و دقت یا سرعت مورد نیاز بستگی دارد.

11-3: سیستمهای نروفازی همکارگونه و همزمان

در ساده‌ترین حالت، یک سیستم همکارگونه به صورت یک شبکه عصبی به عنوان یک مکانیسم یادگیرنده در نظر گرفته می‌شود که به عنوان واحد پیش‌پردازش جهت تعیین عضویت توابع یا قوانین یک FIS عمل می‌کند.

عضویت در گروه‌ها معمولاً به صورت یک الگوریتم خوشه‌بندی (فازی یا غیرفازی) بوده، توابع عضویت نیز عمدتاً با تخمین‌زدن توسط یک شبکه عصبی از روی داده‌های آموزشی حاصل می‌شوند. شکل11-3 بیانگر توضیحات داده‌شده می‌باشد.





شکل11-3: سیستمهای نروفازی همکارگونه و همزمان

در حالت همزمان، شبکه‌های عصبی مرتباً FIS را در تعیین پارامترها یاری می‌کنند، به‌ویژه اگر متغیرهای ورودی کنترل‌کننده فازی مستقیماً قابل‌اندازه‌گیری نباشند. در بعضی حالتها ممکن است خروجی FIS به‌طور مستقیم بر فرایند قابل اعمال نباشد که در این صورت، شبکه عصبی به عنوان واحد پس‌پردازنده برای خروجیهای FIS عمل می‌کند.

11-4: سیستمهای نروفازی فیوزشده

نروفازی‌های فیوزشده، ساختمان داده‌ها و نمایش دانش را به اشتراک می‌گذارند و در معماری خود از انواع الگوریتمهای یادگیری شبکه‌های عصبی برای تعیین پارامترهای FIS استفاده می‌کنند.

یک راه رایج برای بکارگیری الگوریتمهای یادگیری در یک سیستم فازی، ارائه یک معماری شبیه به معماری شبکه‌های عصبی است. البته الگوریتمهای رایج یادگیری شبکه‌های عصبی (مثل روش gradient descent استفاده‌شده در شبکه‌های پرسپترون برای رسیدن به کمترین نرخ خطا) نمی‌توانند مستقیماً در سیستمهایی که توابع قابل تشخیصی در فرایند استنتاجی آنها وجود ندارد، استفاده شوند. این مشکل را با تغییر شبکه‌ها برحسب نیاز حل می‌کنیم. مثلاً می‌توان سیستم استنتاج را به سیستمی با توابع قابل‌تشخیص تعبیرکرد، یا اینکه از شبکه‌های عصبی استاندارد استفاده نکنیم. در ادامه به بررسی چند نمونه جالب از این نوع سیستمها می‌پردازیم.

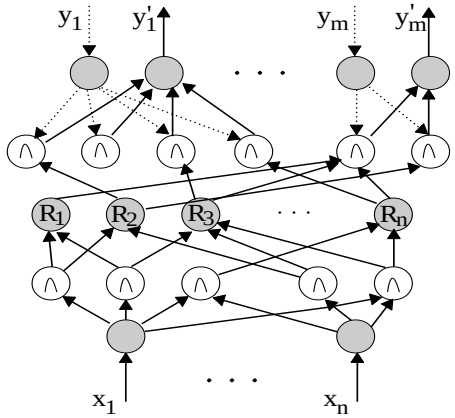
11-4-1: شبکه کنترل یادگیری تطبیق‌پذیر فازی: FALCON[1]

این شبکه شامل یک معماری پنج لایه است که در شکل 11-4 نشان داده شده است. در این معماری دو نود زبانی برای هر متغیر خروجی در نظر گرفته می‌شود که یکی برای داده‌های آموزش (آنچه که انتظار داریم از شبکه به دست آید) و دیگری برای خروجی حاصل از شبکه است. در این شبکه، هر نود می‌تواند خود نمایانگر یک تابع عضویت ساده باشد یا با نود چند لایه دیگر ترکیب شود و یک تابع عضویت پیچیده‌تر را نمایش دهد.

لایه مخفی اول مسئول فازی‌کردن هر یک از متغیرهای ورودی است. لایه مخفی دوم پیش‌شرطهای قوانینی را تعریف می‌کند که در لایه سوم به‌کار می‌رود. FALCON از یک الگوریتم یادگیری ترکیبی سود می‌برد که در آن، از یادگیری بدون سرپرست برای تعیین

[1] Fuzzy Adaptive Learning COntrol Network (FALCON)

توابع عضویت اولیه و قوانین استفاده می‌شود. سپس با استفاده از یک الگوریتم یادگیری برای پیداکردن پارامترهای بهینه تابع عضویت تلاش می‌کند تا به خروجیهای مطلوب برسد.



شکل11-4: معماری FALCON

11-4-2: سیستم استنتاج نروفازی تطبیق‌پذیر: ANFIS[1]

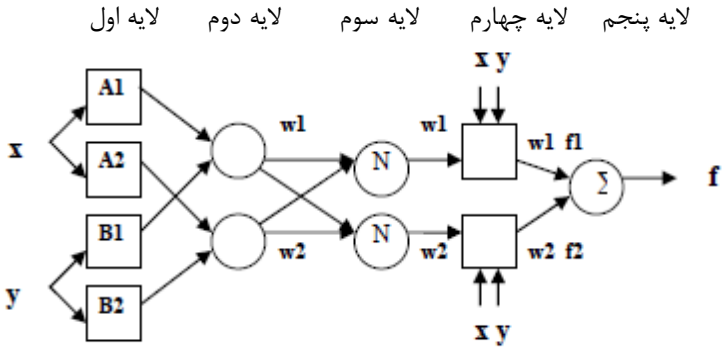
ANFIS در واقع اجرای سیستم FIS ساگنو است و مشتمل بر پنج لایه است که در شکل 11-5 به نمایش درآمده‌اند. لایه اول برای فازی‌کردن متغیرهای ورودی به کار می‌رود و عملگرهای نرم-T توسط لایه دوم برای محاسبه قانون بخش قبل اعمال می‌شوند. لایه سوم مسئول نرمالیزه‌کردن شدت قوانین است که این قوانین در لایه چهارم با توجه به پارامترهای مشخص‌شده آنها ایجاد می‌شوند. لایه خروجی که پنجمین و آخرین لایه است، حاصل‌جمع ورودیها را به خروجی اعلام می‌کند.

ANFIS از یک الگوریتم انتشار به عقب برای یادگیری پارامترها با توجه به میزان تابع عضویت آنها استفاده می‌کند و از طریق محاسبه کمترین میانه، پارامترهای نتیجه‌بخش را می‌یابد. بنابراین یادگیری در این شبکه، شامل دو مرحله خواهد بود که در مرحله اول الگوهای ورودی دوباره انتشار پیدا می‌کنند و پارامترهای نتیجه‌بخش با یک روال تکراری

[1] Adaptive Neuro Fuzzy Inference Systems (ANFIS)

محاسبه کمترین میانه حاصل می‌شوند. در این مرحله پارامترهای اساسی در طول حلقه، ثابت فرض می‌شوند.

در مرحله دوم، الگوها مجدداً منتشر می‌شوند که در این تکرار[1] از روش انتشار به عقب برای بهینه‌سازی پارامترهای اساسی استفاده می‌شود. باید دقت داشت که در این مرحله، پارامترهای نتیجه‌بخش به عنوان مقدار ثابت در نظر گرفته می‌شوند.



شکل11-5: معماری ANFIS

11-4-3: کنترل هوشمند براساس استنتاج تقریبی تعمیم‌یافته: GARIC[2]

این سیستم یک کنترلر نروفازی را با استفاده از دو ماژول شبکه‌های عصبی اجرا می‌کند. این دو ماژول شامل شبکه انتخاب عملکرد یا ASN[3] و شبکه محاسبه حالت یا ASE[4] می‌باشد.

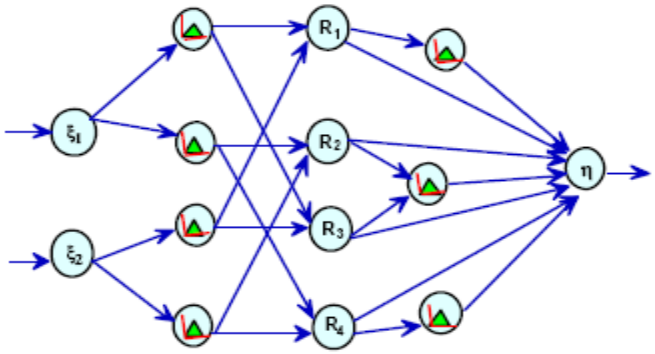
قسمت ASE، یک منتقد تطبیق‌پذیر است که کیفیت اعمال ASN را محاسبه می‌کند و قسمت ASN یک شبکه روبه‌جلو پنج‌لایه است. شکل11-6 نشان‌دهنده ساختار قسمت ASN یک GARIC است. دقت شود که اتصالات بین لایه‌های آن دارای وزن نیستند.

---

[1] Epoch
[2] Generalized Approximate Reasoning based Intelligent Control (GARIC)
[3] Action Select Network (ASN)
[4] Action State Evaluation (ASE)

لایه اول متغیرهای زبانی تمام متغیرهای ورودی را ذخیره می‌کند. هر واحد ورودی فقط با واحدهایی از لایه اول اتصال دارد که با متغیرهای زبانی آن مرتبط است. لایه دوم شامل نودهای قوانین فازی است و درجه ارزشمندی یک قانون را با استفاده از عملیات softmin مشخص می‌کند. لایه سوم مقادیر زبانی متغیرهای خروجی کنترل را نمایش می‌دهد. نتایج قوانین بسته به قدرت قانون اولویت توسط نود لایه قانون محاسبه می‌شود.

GARIC از روش میانه- ماکزیمم محلی برای محاسبه قوانین خروجی استفاده می‌کند. این روش نیاز به یک مقدار بی‌دوام و موقت خروجی از هر قانون دارد. بنابراین، نتیجه‌گیری نهایی باید قبل از جمع‌شدن در آخرین مقدار خروجی کنترلر، غیرفازی شود. GARIC از یک یادگیری مرکب شامل gradient descent و reinforcement برای تنظیم پارامترهای نودها استفاده می‌کند.



شکل11-6: ساختار ASN در شبکه GARIC

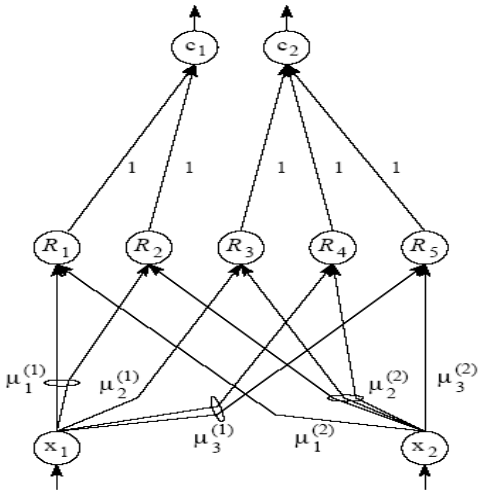
11-4-4: کنترل نروفازی: NEFCON [1]

NEFCON براساس پیاده‌سازی با مدل ممدانی طراحی شده است. تمامی اتصالات در NEFCON توسط مجموعه‌های فازی و قوانینی که مقدم آنها یکسان است، وزن‌دهی شده‌اند. این وزنها که به وزنهای اشتراکی موسومند، در شکل 11-7 با کشیدن بیضی به دور آنها مشخص شده‌اند. این مسئله، یکپارچگی براساس قوانین را تضمین می‌کند.

[1] NEuro Fuzzy CONtrol (NEFCON)

واحد ورودی وظیفه یک واسط فازی‌کننده را عهده‌دار می‌باشد. واسط منطق، توسط توابع انتشاری ارائه می‌شود و واحد خروجی (برعکس واحد ورودی) یک واسط غیرفازی‌کننده می‌باشد. فرایند یادگیری در NEFCON به صورت ترکیبی از reinforcement و انتشار به عقب است. اگر هیچ دانش اولیه‌ای درباره سیستم وجود نداشته باشد، آنگاه می‌توان از NEFCON برای یادگیری اولیه قوانین پایه و حتی برای بهینه‌سازی دستی تعیین این قوانین استفاده کرد. NEFCON در دو حالت متفاوت کار می‌کند:

1. از NEFRROX برای تخمین توابع استفاده می‌شود.
2. از NEFCLASS به عنوان تفکیک‌کننده استفاده می‌شود.



شکل11-7: معماری شبکه NEFCON

11-4-5: واسط فازی و شبکه‌های عصبی در نرم‌افزارهای FINEST[1]

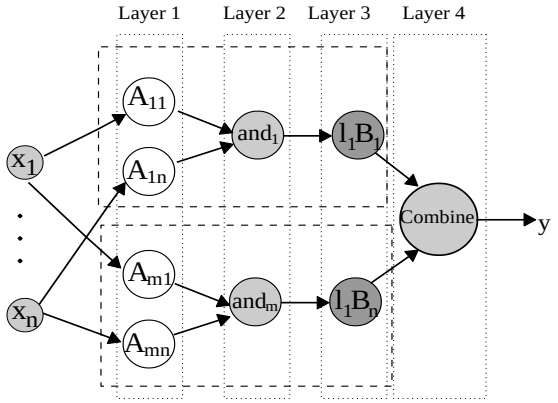
FINEST توانایی انجام فرآیندهای تنظیمی را دارا می‌باشد. این فرآیندها شامل تنظیم عبارات فازی، ترکیب توابع و نیز تنظیم یک تابع ضمنی است. در این مجموعه نرم‌افزارها،

[1] Fuzzy Inference and Neural Network in Fuzzy Inference Software (FINEST)

می‌توان از چهار روش مختلف برای بهبود استنتاج وضع مقدم عمومی استفاده کرد. این روشها عبارتند از:

1. اپراتورهای تجمعی با طبیعت هماهنگ و لغوپذیر
2. تابع ضمنی پارامتردارشده
3. تابع ترکیب‌کننده با قابلیت کاهش میزان فازی‌بودن
4. استنتاج زنجیره‌ای رو به عقب براساس وضع مقدم عمومی

FINEST از الگوریتم انتشار به عقب برای تنظیم مناسب پارامترها استفاده می‌کند. شکل11-8 معماری FINEST و فرایند محاسبات استنتاج فازی را نشان می‌دهد. FINEST برای تنظیم هر پارامتری که در نودهای نمایش‌دهنده محاسبات داده‌های فازی ظاهر شود، یک ساختار ایجاد می‌کند. این امر در صورتی انجام می‌شود که تابع مبدل مشتق به همراه پارامترهای مناسب ارائه شده باشد.

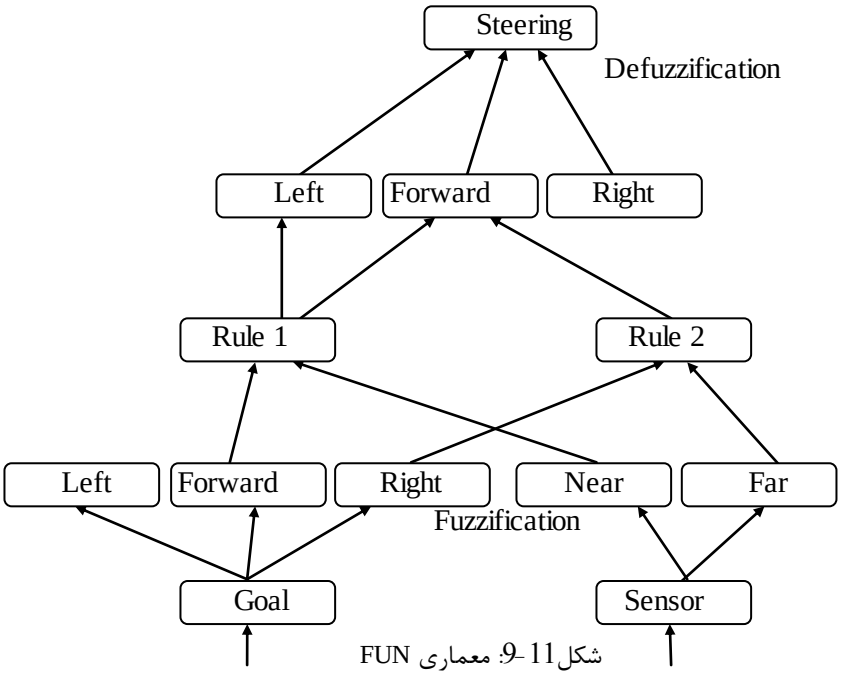


شکل11-8: معماری FINEST

11-4-6: شبکه‌های فازی: FUN[1]

در FUN نرونها در لایه اول مخفی شامل توابع عضویت بوده، وظیفه فازی‌کردن متغیرهای ورودی را برعهده دارند. در لایه مخفی دوم، تمامی ترکیبهای and فازی محاسبه می‌شود. تابع عضویت متغیرهای خروجی در لایه سوم ذخیره شده است و توابع فعال‌ساز نرونها، یک تابع OR فازی می‌باشد. یک نرون خروجی، عملیات غیرفازی‌کردن را در پایان کار انجام می‌دهد (شکل 11-9).

شبکه به کمک قوانین فازی پایه و توابع عضویت نظیربه‌نظیر مقداردهی اولیه می‌شود. سپس با استفاده از یک تکنیک یادگیری آماری، توابع عضویت و ساختار اتصالات درونی شبکه به‌طور تصادفی تغییر می‌یابد. فرایند یادگیری از یک تابع هزینه مشتق شده است که این تابع بعد از اصلاحات تصادفی ارزیابی می‌شود تا اگر نتایج اصلاحات، کارایی را بهبود بخشیده باشد، این اصلاحات برای مراحل بعدی حفظ بشود.



شکل 11-9: معماری FUN

[1] FUzzy Nets (FUN)

11-4-7: شبکه‌های عصبی فازی استنتاجی: EFuNN[1]

تمامی نودها در EFuNN در حین فرایند یادگیری ساخته می‌شوند. لایه دوم، داده را از لایه ورودی دریافت می‌نماید و میزان درجه عضویت فازی هر یک از متغیرهای ورودی به توابع عضویت فازی ازپیش‌تعریف‌شده را محاسبه می‌کند. لایه سوم، شامل نودهای قوانین فازی است که نمایش‌دهنده طرح اولیه ورودی- خروجی داده‌ها به عنوان ترکیبی از فضای مرتبه بالای ورودی و فضاهای خروجی می‌باشد.

در هر گره قانون دو بردار وزنهای اتصال که برای تکنیکهای یادگیری ترکیبی تنظیم شده‌اند، تعریف می‌شود. لایه چهارم محاسبه می‌کند که کدام تابع عضویت خروجی از چه درجه‌ای با کدام داده ورودی همخوانی دارد. درنهایت، لایه پنجم وظیفه غیرفازی‌کردن و محاسبه متغیرهای دقیق خروجی را برعهده دارد.

نسخه‌های متعددی از EFuNN در کاربردهای مختلف مورد استفاده قرار می‌گیرد که یکی از این نسخه‌های اصلاح‌شده با نام dmEFuNN شناخته می‌شود.

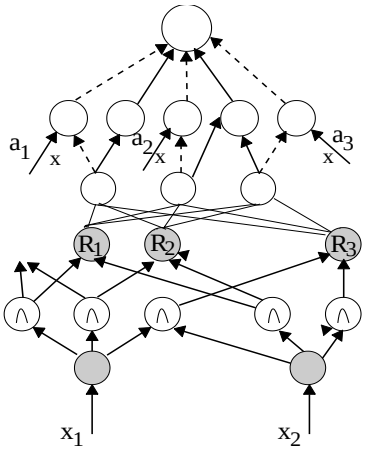
11-4-8: شبکه عصبی فازی استنتاجی خودساخته: SONFIN [2]

همان‌طور که در شکل 11-10مشاهده می‌شود، SONFIN یک واسط فازی ساگنو را اجرا می‌کند. در ساختار شناسایی قسمت پیش‌شرط، فضای ورودی براساس یک الگوریتم خوشه‌بندی به طریقی کاملاً انعطاف پذیر بخش‌بندی می‌شود. در ساختار شناسایی قسمت تالی، مقدار یگانه‌ای به هر قانون به صورت مقداردهی اولیه توسط روش خوشه‌بندی داده می‌شود. سپس تعدادی عبارت معنادار (متغیرهای ورودی) که از طریق یک فرایند اندازه‌گیری میزان همبستگی مبتنی بر روابط and حاصل می‌شوند، به هر قانون در قسمت تالی اضافه می‌شود. باید توجه داشت که این کار به صورت افزایشی و برای پیشرفت یادگیری انجام می‌شود.

---

[1] Evolving Fuzzy Neural Network (EFuNN)
[2] Self cOnstructing Neural Fuzzy Inference Network (SONFIN)

برای شناسایی پارامترها، پارامترهای تالی توسط یک الگوریتم بازگشتی حداقل مربعات یا حداقل میانه مربعات به صورت بهینه تنظیم می‌شوند و پارامترهای پیش‌شرط نیز توسط یک الگوریتم انتشار به عقب تنظیم خواهند شد.



شکل11-10: معماریSONFIN

11-5: طراحی سیستمهای نروفازی

استنتاج در مدلهای نروفازی می‌تواند با روشهای به‌کاررفته در واسطهای فازی ممدانی یا ساگنو منطبق شود. فرایند جستجو در مورد چگونگی نوع بهینه، تعداد نودها و ارتباطات بین‌لایه‌ای تصمیم می‌گیرد.

لایه فازی‌کننده و لایه قوانین مقدم، شبیه به سایر مدلهای نروفازی می‌باشد. قسمت تالی قوانین براساس سیستم استنتاج و نوع مسئله مشخص می‌شود که براساس مکانیزم جستجوی تکاملی کار می‌کند. برای هر پارامتر یادگیری، یک جستجوی سراسری مکانیزم استنتاج در محیطی سریعتر، باتوجه به تصمیم مکانیزم استنتاج، پارامترهای یادگیری و نوع مسئله انجام می‌شود. ویژگیهای محیطی مطرح‌شده، باعث می‌شود تا یک جستجوی سراسری در قوانین فازی (برای هر مکانیزم استنتاج) سریعتر انجام شود. در حالت مشابه، همین توضیحات در مورد معماری شبکه قابل بیان است.

حالت اکتشافی موجود در توابع انطباقی متفاوت، با تکیه بر دانش اولیه خواهد بود. مثلاً اگر دانش اولیه به معماری توجه بیشتری (نسبت به مکانیزم استنتاج) داشته باشد، بهتر است که اجرای معماری در سطوح بالاتر انجام شود.

در این فصل انواع ساختارهای نروفازی را به صورت فهرست‌وار و با بیان اجمالی ویژگیهای آنها، معرفی کردیم. بدون‌شک بررسی هرکدام از این ساختارها، خود نیازمند یک فصل یا کتاب مجزا می‌باشد. در پایان، یک سیستم کاربردی در نروفازی را که به عنوان بخشی از یک پروژه کارشناسی ارشد توسط مولف از آن دفاع شده است، بررسی می‌کنیم.

11-6: اعراب‌گذاری متون فارسی با شبکه‌های نروفازی

اعراب‌گذاری در جملات فارسی (خصوصاً در جملات ادبی و علمی) نقش مهمی را در ادای معنا و مفهوم جملات بازی می‌کند. یک مشکل مهم، عدم اِعمال اعراب‌گذاری جملات هنگام ورود آنها به سیستم است. بنابراین، باید اعراب‌گذاری را به شکلی به سیستم اعمال نمود. بدیهی است که واردکردن کلیه کلمات به همراه اعراب آنها، کاری پیچیده و زمان‌بر است و نیز کارایی سیستم را برای بررسی صحت اعرابها پایین می‌آورد.

مسئله خود را این‌گونه تعریف می‌کنیم: «طراحی برنامه‌ای هوشمند که متن ساده‌ای را دریافت کند و پس از اعراب‌گذاری، در اختیار میانجی‌های هوشمند قرار دهد.»

برنامه طراحی‌شده باید به دو نکته مهم توجه نماید: کلمات فارسی براساس نقش خود در جمله، ممکن است در حرف پایانی کسره بگیرند. همچنین باید دقت داشت که یک کلمه ممکن است اعرابهای متفاوتی داشته باشد. مثلاً کلمه «رود» می‌تواند به عنوان یک حالت از فعل رفتن (رَوَد) یا جریان آب معنی شود. این امر، کاملاً به مفهوم جمله و محل قرارگرفتن کلمه در آن وابسته است.

برای تشخیص نوع کلمه، می‌توان از زنجیره مارکوف به همراه روشهای آماری و نیز شبکه‌های فازی استفاده کرد که در ادامه به بررسی روش فازی خواهیم پرداخت.

به منظور استفاده از اندازه‌های امکان برای تشخیص نوع کلمات، از یک شبکه فازی به صورت گرافی جهت‌دار با لبه‌های وزن‌دار استفاده می‌کنیم.

G(V,E) یک گراف جهت‌دار فازی است که در آن V مجموعه محدودی از نودهای فازی و E مجموعه محدودی از لبه‌های فازی است. با دو نود فازی موجود در دو طرف هر لبه، اعضای آن مشخص می‌شوند.

Vمجموعه‌ای از $W\times K$است که Wمجموعه محدود واژگان تشکیل‌دهنده جمله و K مجموعه‌ای از حالتهای متفاوت و متوالی برای کلمات Wمی‌باشد. نودهای فازی در شبکه فازی به صورت زیر تعریف می‌شوند:

$$V=\left\{\left((w,k),\mu_v(w,k)\right)|(w,k)\in W\times K\right\}$$

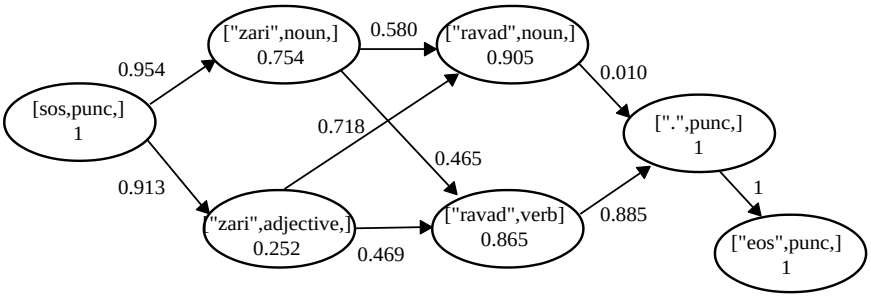
E زیرمجموعه‌ای از ضرب کارتزین V×V است. لبه‌های فازی در شبکه فازی G به صورت زیر تعریف می‌شوند:

$$E=\left\{\left((v_i,v_j),\mu_E(v_i,v_j)\right)|(v_i,v_j)\in V\times V\right\}$$

به عنوان مثال در کلمه «رود» برای امکان فعل یا اسم بودن آن می‌توان نوشت :

$$V=\left\{(("ravad",verb),0.753),(("ravad",noun),0.905)\right\}$$

شبکه فازی کوچک زیر با چند کلمه و نماد ساده (شکل11-11) ساخته شده است.



شکل11-11: مثالی از یک شبکه فازی کوچک

یک مسیر با نودهای مشخص‌شده $v_n=(w_n,k_n),...,v_2(w_2,k_2),v_1(w_1,k_1)$ که با لبه‌های (v_i,v_{i-1}) با هم مرتبط هستند، بیان می‌شود. برای هر V_i خواهیم داشت:

$$\mu_v(v_i)>0,\mu_E=(v_i,v_{i+1})>0$$

نودهای V_1و V_n به ترتیب نود مبدا و مقصد می‌باشند که به عنوان علامت‌گذاریهای مخصوص (sos,eos) در نظر گرفته شده‌اند. قوت هر مسیر $P = v_1, v_2, ..., v_n$ از رابطه زیر به دست می‌آید:

$$\mu_E(v_1, v_2) \wedge \mu_E(v_2, v_3) \wedge ... \wedge \mu_E(v_{n-1}, v_n) \wedge \mu_v(v_2) \wedge ... \wedge \mu_v(v_u)$$

که $\wedge$ یک عملگر فازی برای نرم مثلثاتی است و $\mu_v(v_i), \mu_E(v_i, v_{i+1})$ درجه عضویت بافتاری و لغوی فازی را نشان می‌دهند.

11-6-1: تابع عضویت لغوی فازی

برای رسیدن از اندازه‌های احتمال به اندازه‌های امکان، می‌توان از شبکه‌های عصبی استفاده کرد. اما با توجه به لزوم اختصاص یک نود به هر کلمه، نیازمند شبکه عصبی با هزاران نود خواهیم بود. بنابراین از روش دیگری استفاده می‌کنیم:

اگر در متن آموزشی تعداد تکرار کلمه w در نوع k را با $C(w,k)$ نمایش دهیم، داریم:

$$\Pr(k/w) = \frac{C(w,k)}{\sum_s C(w,s)}$$

که s مجموعه k های امکان پذیر است. با توجه به اینکه ممکن است در بعضی حالات $C(w,k)$ برابر با صفر باشد، یک عدد مثبت بین صفر و یک را به نام α در نظر می‌گیریم و رابطه فوق را به صورت زیر تصحیح می‌کنیم:

$$\overline{\Pr}(k/w) = \frac{\alpha + C(w,k)}{\sum_s \alpha + C(w,s)}$$

حال می‌توانیم تابع عضویت نودها را به وسیله رابطه زیر که اندازه‌های احتمال را به اندازه‌های امکان تبدیل می‌کند، بدست آوریم:

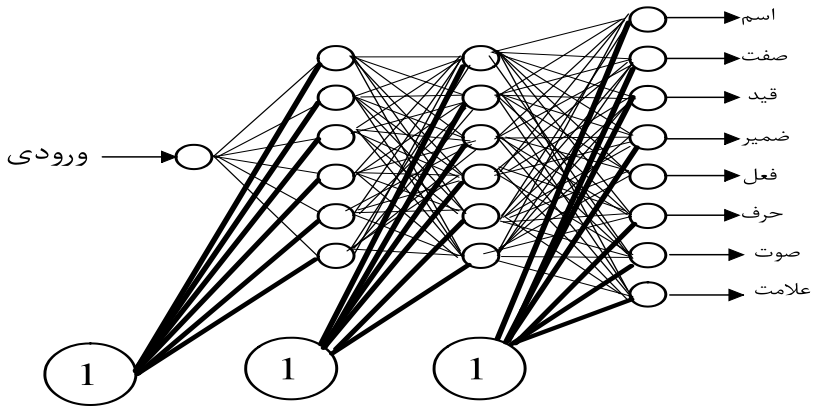
$$\mu_v(w,k) = 1 - \sqrt[q]{1 - \overline{Pr}(k/w)}$$

که درآن، q یک پارامتر کنترلی است و در مقادیر نزدیک به 0.8 نتیجه مناسبی را نشان داده است.

11-6-2: تابع عضویت بافتاری فازی

برای به‌دست آوردن تابع عضویت بافتاری فازی (اوزان شبکه فازی) از شبکه عصبی پرسپترون چندلایه استفاده می‌کنیم و فقط تابع عضویت قرارگرفتن هشت نوع مختلف بعد از یک نوع از این انواع هشت‌گانه را به دست می‌آوریم.

همان‌طور که در شکل 11-12 مشاهده می‌شود، این شبکه یک ورودی و هشت خروجی دارد که هر خروجی نماینده یک نوع است. برای یافتن تابع عضویت قرار گرفتن هشت نوع مختلف بعد از یک نوع (مثلاً اسم) در متن آموزش حرکت می‌کنیم و با برخورد به انواع «اسم + یک نوع از انواع هشت‌گانه» آنرا در ورودی و خروجی قرار داده، شبکه را با الگوریتم انتشار به عقب آموزش می‌دهیم.

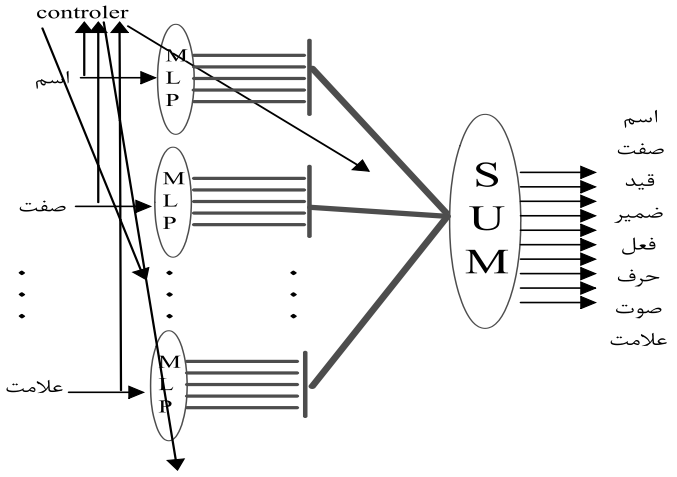


شکل 11-12: تابع عضویت بافتاری فازی

مثلاً اگر ترتیب «اسم + صفت» رخ دهد، در ورودی مقدار یک و در خروجی بردار [01000000] را قرار داده، شبکه را آموزش می‌دهیم. پس از آنکه تمامی ترتیبهای مختلف را به شبکه آموزش دادیم، می‌توانیم در ورودی یک قرار دهیم تا در خروجی، امکان قرار گرفتن هر یک از انواع بعد از اسم را دریافت کنیم.

این نکته شایان ذکر است که برای به دست آوردن انواع ترتیبهای مختلف که بیانگر تابع عضویت بافتاری فازی باشند، نیاز به هشت شبکه عصبی منفرد داریم. می‌توان با یک

معماری خاص، این هشت شبکه را در کنار یکدیگر قرار داد و آنها را به صورت یک شبکه عصبی چندگانه مطابق شکل 11-13 نمایش داد.



شکل 11-13: شبکه عصبی چندگانه بیانگر تابع عضویت بافتاری فازی

در این شبکه عصبی چندگانه، واحد کنترل‌کننده (طبق بردار ورودی) در هر لحظه فقط یکی از شبکه‌های منفرد MLP را فعال می‌کند. این واحد هم هنگام آموزش و هم هنگام استفاده از آن، بدین شکل عمل می‌کند.

برای آموزش شبکه عصبی پرسپترون چندلایه با ظاهرشدن هر ترتیب از انواع، شبکه را با الگوریتم انتشار به عقب اصلاح‌یافته آموزش می‌دهیم. به عنوان مثال اگر در ترتیب انواع «قید + صفت» را داشته باشیم، در ورودی بردار [00100000] و در خروجی بردار [01000000] را قرارداده، شبکه را تا رسیدن به تغییرات اندک در خروجی آموزش خواهیم داد. این عمل را برای ترتیبهای مختلف ظاهرشده در متن آموزش انجام می‌دهیم. برای استفاده از شبکه و یافتن تابع عضویت بافتاری فازی، با قراردادن یک در ورودی مربوط به هر نوع تابع عضویت قرارگرفتن انواع هشت‌گانه بعد از نوع مورد نظر را می‌توانیم در خروجی دریافت نماییم. پس از تشخیص نوع واژگان تشکیل‌دهنده جمله، حاصل این مرحله را برای کسره‌گذاری به بخش مربوط به تشخیص کسره می‌سپاریم تا در آنجا با سیستمهای قاعده- پایه مبتنی بر نظریه احتمالات و فازی، کسره‌گذاری شود.