بسم الله الرحمن الرحيم

دانشکده مهندسی برق و رباتیک

گروه الکترونیک

پیاده‌سازی شبکه عصبی فازی روی FPGA و به‌کارگیری آن در طبقه‌بندی

دانشجو:

فرید فرج‌زاده اصل

استاد راهنما:

دکتر علی سلیمانی ایوری

استاد مشاور:

دکتر حسین خسروی

پایان نامه جهت اخذ درجه کارشناسی ارشد

بهمن ۱۳۹۱

شماره : ۱۰۵۰/آ.ت.ب
تاریخ : ۹۱/۱۱/۲۹
ویرایش : ------

بسمه تعالی

مدیریت تحصیلات تکمیلی

فرم شماره (۶)

فرم صورتجلسه دفاع پایان نامه تحصیلی دوره کارشناسی ارشد

با تأییدات خداوند متعال و با استعانت از حضرت ولی عصر (عج) جلسه دفاع از پایان نامه کارشناسی ارشد خانم / آقای :

فرید فرج زاده اصل رشته : **بـــرق** گرایش : **الکترونیک**

تحت عنوان : **پیاده سازی شبکه عصبی فازی روی FPGA و بکارگیری آن در طبقه بندی**

که در تاریخ **۱۳۹۱/۱۱/۲۹** با حضور هیأت محترم داوران در دانشگاه صنعتی شاهرود برگزار گردید به شرح زیر است :

قبول (با درجه : بسیار خوب امتیاز (۱۸/۸۸) ☑ دفاع مجدد ☐ مردود ☐

۱ـ عالی (۲۰ ـ ۱۹)
۲ـ بسیار خوب (۱۸/۹۹ ـ ۱۸)
۳ـ خوب (۱۷/۹۹ ـ ۱۶)
۴ـ قابل قبول (۱۵/۹۹ ـ ۱۴)
۵- نمره کمتر از ۱۴ غیر قابل قبول

عضو هیأت داوران	نام ونام خانوادگی	مرتبهٔ علمی	امضـــاء
۱ـ استـاد راهنمـا			
۲ـ استـاد مشـاور			
۳ـ نماینده شورای تحصیلات تکمیلی			
۴ـ استـاد ممتحـن			
۵ ـ استـاد ممتحـن			

رئیس دانشکده :

تقدیم به

روح پدرم

و

قلب پر مهر مادرم

و

تقدیم به خواهرانم و برادرانم

تشکر و قدردانی

سپاس و شکر خداوند متعال را، که در همه لحظات زندگی دستگیر و یاورم بوده است.
از استاد راهنمای بزرگوارم، آقای دکتر علی سلیمانی ایوری، که با راهنمایی‌های ارزنده خود، مرا در انجام این تحقیق یاری نمودند، تشکر و قدردانی می‌کنم. همچنین از آقای دکتر حسین خسروی که استاد مشاور بنده در این تحقیق بودند و راهنمایی‌های کارگشا و مؤثری برای انجام این تحقیق ارائه نمودند، سپاسگزاری می‌نمایم. از خانواده عزیزم که همواره باعث دلگرمی بنده بودند تشکر می‌نمایم.
در پایان، از همه دوستانم به خاطر حمایت‌های برادرانه‌شان تشکر می‌کنم.

تعهد نامه

اینجانب **فرید فرجزاده اصل** دانشجوی دوره کارشناسی ارشد رشته مهندسی برق - گرایش الکترونیک دیجیتال دانشکده مهندسی برق و رباتیک دانشگاه صنعتی شاهرود نویسنده پایان نامه با عنوان **"پیاده‌سازی شبکه عصبی فازی روی FPGA و به کارگیری آن در طبقه‌بندی"** تحت راهنمائی **دکتر علی سلیمانی ایوری** متعهد می‌شوم:

- تحقیقات در این پایان‌نامه توسط اینجانب انجام شده است و از صحت و اصالت برخوردار است.
- در استفاده از نتایج پژوهش‌های محققان دیگر به مرجع مورد استفاده استناد شده است.
- مطالب مندرج در پایان‌نامه تاکنون توسط خود یا فرد دیگری برای دریافت هیچ نوع مدرک یا امتیازی در هیچ جا ارائه نشده است.
- کلیه حقوق معنوی این اثر متعلق به دانشگاه صنعتی شاهرود می‌باشد و مقالات مستخرج با نام «دانشگاه صنعتی شاهرود» و یا «Shahrood University of Technology» به چاپ خواهد رسید.
- حقوق معنوی تمام افرادی که در به دست آمدن نتایج اصلی پایان‌نامه تأثیرگذار بوده‌اند در مقالات مستخرج از پایان‌نامه رعایت می‌گردد.
- در کلیه مراحل انجام این پایان‌نامه، در مواردی که از موجود زنده (یا بافتهای آنها) استفاده شده‌است، ضوابط و اصول اخلاقی رعایت شده‌است.
- در کلیه مراحل انجام این پایان‌نامه، در مواردی که به حوزه اطلاعات شخصی افراد دسترسی یافته یا استفاده شده‌است اصل رازداری، ضوابط و اصول اخلاق انسانی رعایت شده‌است.

تاریخ:

امضای دانشجو:

مالکیت نتایج و حق نشر

- کلیه حقوق معنوی این اثر و محصولات آن (مقالات مستخرج ، کتاب ، برنامه های رایانه ای ، نرم افزار ها و تجهیزات ساخته شده است) متعلق به دانشگاه صنعتی شاهرود می باشد. این مطلب باید به نحو مقتضی در تولیدات علمی مربوطه ذکر شود.
- استفاده از اطلاعات و نتایج موجود در پایان نامه بدون ذکر مرجع مجاز نمی‌باشد.

چکیده: شبکه‌های عصبی فازی Min-Max، روش‌های نسبتاً جدیدی در حوزه کاربردی طبقه‌بندی و خوشه یابی می‌باشند. این شبکه‌ها، ویژگی‌های برتری مانند الگوریتم آموزش برخط یکبارگذر دارند که آنها را برای پیاده‌سازی سخت‌افزاری و استفاده در کاربردهای بلادرنگ، ایده‌آل می‌سازد.

در این پایان‌نامه یک روش جدید برای طراحی شبکه‌های عصبی فازی بر پایه ابرجعبه‌های فازی Min-Max ارائه شده است. همانند شبکه FMNN، این شبکه از تجمیع ابرجعبه‌های فازی برای تعیین مرزهای کلاس‌ها استفاده می‌کند، با این تفاوت که تست همپوشانی و انقباض ابرجعبه‌ها از مراحل آموزش حذف شده است. که منجر به الگوریتم آموزش ساده‌تر و سرعت آموزش بیشتر شده است. شبکه ارائه شده از دو دسته ابرجعبه با ضریب انبساط متفاوت، به منظور استفاده از ابرجعبه‌های کوچک‌تر در مرز کلاس‌ها، استفاده می‌کند. شبکه با یک بار ارائه داده‌های آموزش، آموزش می‌بیند. نتایج حاصل از شبیه‌سازی‌های رایانه‌ای، بیانگر کارایی خوب این شبکه می‌باشد. روش پیشنهادی در اکثر موارد، نتایج بهتری را، نسبت به بهترین روش‌های قبلی، با تعداد نرون‌های کمتر بدست می‌دهد.

در پایان‌نامه حاضر، یک طرح سخت‌افزاری، برای پیاده سازی شبکه معرفی شده روی FPGA، ارائه شده است. در این طرح تابع عضویت ابرجعبه‌ها، به منظور صرفه‌جویی در ناحیه مصرفی FPGA، ساده شده است، به طوری که طرح حاصله، از ضرب کننده استفاده نمی‌کند. همچنین طرح سخت‌افزاری ارائه شده، به طور موثری از تکنیک خط‌لوله و موازی‌کاری، برای افزایش سرعت محاسبات، استفاده می‌کند.

کلمات کلیدی: طبقه‌بندی، شبکه عصبی فازی، ابرجعبه فازی Min-Max، FPGA.

مقالات مستخرج از پایان‌نامه:

فرید فرجزاده اصل، علی سلیمانی ایوری، حسین خسروی، " *روش جدیدی برای طبقه‌بندی الگو، بر پایه ابرجعبه‌های فازی Min-Max*"، یازدهمین کنفرانس سیستم‌های هوشمند ایران، دانشگاه خوارزمی، تهران، اسفند ۱۳۹۱.

فهرست مطالب

عنوان **صفحه**

فصل ۱- مقدمه ۲

۱-۱- پیشگفتار ۲

۱-۲- نوآوری پایان‌نامه ۵

۱-۲-۱- طراحی شبکه ۵

۱-۲-۲- پیاده سازی شبکه روی FPGA ۶

۱-۳- ساختار پایان‌نامه ۶

فصل ۲- مروری بر شبکه‌های عصبی مصنوعی و منطق فازی ۹

۲-۱- شبکه عصبی مصنوعی ۹

۲-۱-۱- کاربردهای شبکه‌های عصبی ۱۰

۲-۱-۲- ساختارهای شبکه‌های عصبی ۱۰

۲-۱-۳- روش‌های یادگیری ۱۰

۲-۱-۴- مزایا و محدودیت‌های شبکه‌های عصبی ۱۱

۲-۲- منطق فازی ۱۱

۲-۲-۱- تاریخچه ۱۲

۲-۲-۲- مجموعه فازی ۱۲

۲-۲-۳- مفاهیم مجموعه‌های فازی ۱۳

۲-۲-۴- عملیات بر روی مجموعه‌های فازی ۱۳

۲-۲-۵- اصل توسعه ۱۴

۲-۲-۶- اعداد فازی ۱۴

۲-۲-۷- متغیر زبانی ۱۵

۲-۲-۸- قواعد اگر-آنگاه فازی ۱۵

۲-۲-۹- تفسیر قواعد اگر-آنگاه فازی ۱۶

۲-۲-۱۰- استلزام فازی ۱۶

۲-۳- سیستم‌های فازی ۱۷

۲-۳-۱- پایگاه قواعد فازی ۱۷

۲-۳-۲- موتور استنتاج فازی ۱۸

۲-۳-۳- فازی‌ساز ها ۱۸

۲-۳-۴- غیرفازی‌سازها ۱۸

۲-۴- ترکیب شبکه‌عصبی با منطق فازی ۱۸

۲-۴-۱- شبکه‌های عصبی فازی ۱۹

۲-۴-۱-۱- شبکه عصبی ترکیبی ۱۹

۲-۴-۱-۲- شبکه عصبی فازی منظم ۲۰
۲-۴-۲- سیستم‌های استنتاج عصبی-فازی ۲۰
۲-۴-۲-۱- سیستم‌های عصبی-فازی مشارکتی ۲۰
۲-۴-۲-۲- سیستم‌های عصبی-فازی همزمان ۲۱
۲-۴-۲-۳- سیستم‌های عصبی-فازی یکپارچه ۲۲
۲-۴-۳- ANFIS ۲۳
۲-۴-۳-۱- آموزش ANFIS ۲۵
۲-۴-۴- NEFCLASS ۲۶
۲-۵- شبکه‌های عصبی فازی Min-Max ۲۸
۲-۵-۱- شبکه FMNN ۳۰
۲-۵-۲- آموزش FMNN ۳۳
۲-۵-۲-۱- انبساط ۳۳
۲-۵-۲-۲- تست همپوشانی ۳۴
۲-۵-۲-۳- انقباض ۳۵
۲-۵-۳- ساختار شبکه FMNN ۳۶
۲-۵-۴- بهبود FMNN ۳۸
۲-۵-۵- شبکه FMCN ۳۹
۲-۵-۵-۱- بخش CLN ۴۰
۲-۵-۵-۲- بخش OCN ۴۲
۲-۵-۵-۳- بخش CCN ۴۴
۲-۵-۵-۴- آموزش FMCN ۴۶
۲-۵-۵-۵- فاز تست ۴۷

فصل ۳- پیاده‌سازی سخت‌افزاری ۵۰

۳-۱- پیاده‌سازی سخت‌افزاری شبکه‌های عصبی ۵۰
۳-۱-۱- مشکلات پیش رو در طراحی HNN ۵۱
۳-۱-۲- پارامترهای ارزیابی HNN ۵۱
۳-۱-۳- طبقه‌بندی انواع HNN ۵۳
۳-۱-۴- رویکردهای سخت‌افزاری طراحی نرون ۵۳
۳-۱-۴-۱- نرون دیجیتال ۵۳
۳-۱-۴-۲- نرون آنالوگ ۵۴
۳-۱-۵- تراشه‌های HNN ۵۴
۳-۱-۵-۱- تراشه عصبی دیجیتال ۵۵
۳-۱-۵-۲- تراشه عصبی آنالوگ ۵۵
۳-۱-۵-۳- تراشه عصبی ترکیبی ۵۶
۳-۱-۵-۴- پیاده‌سازی‌ها بر پایه FPGA ۵۶
۳-۱-۶- کاربردها ۵۷

۳-۲- مقدمه کوتاهی در مورد FPGA ۵۷
۳-۲-۱- کاربرد FPGA ۵۹
۳-۲-۲- مزایا و معایب FPGA ۶۰
۳-۲-۲-۱- مزایای FPGA ۶۰
۳-۲-۲-۲- معایب FPGA ۶۱
۳-۲-۳- فرآیند پیاده‌سازی روی FPGA ۶۱
۳-۳- تکنیک‌های بهینه‌سازی طراحی FPGA ۶۳
۳-۳-۱- تکنیک‌های بهینه‌سازی سرعت ۶۳
۳-۳-۲- تکنیک‌های بهینه‌سازی ناحیه ۶۴
۳-۳-۳- تکنیک‌های بهینه‌سازی توان ۶۶
۳-۴- مروری بر کارهای صورت گرفته ۶۸

فصل ۴- روش جدید طبقه‌بندی برپایه ابرجعبه‌های فازی Min-Max ۷۲
۴-۱- ساختار شبکه پیشنهادی ۷۳
۴-۲- الگوریتم آموزش ۷۷
۴-۳- نتایج آزمایش‌ها ۷۸
۴-۴- نتیجه گیری ۸۹

فصل ۵- پیاده‌سازی الگوریتم پیشنهادی روی FPGA ۹۲
۵-۱- شبکه پیشنهادی با تابع عضویت ساده شده ۹۲
۵-۲- طراحی شبکه پیشنهادی روی FPGA ۹۴
۵-۲-۱- زیرسیستم تابع عضویت ۹۶
۵-۲-۲- زیرسیستم محاسبه خروجی یک طبقه ۹۸
۵-۲-۳- نحوه ارتباط زیرسیستم‌ها ۹۹
۵-۲-۴- زیرسیستم محاسبه خروجی کل ۱۰۱
۵-۳- شبیه‌سازی سخت‌افزار طراحی شده ۱۰۲
۵-۳-۱- انتخاب نوع FPGA ۱۰۲
۵-۳-۲- انتخاب نوع کوانتیزاسیون و تعداد بیت ۱۰۲
۵-۳-۳- نتایج حاصل از شبیه‌سازی‌ها ۱۰۳
۵-۴- نتیجه‌گیری ۱۰۶

فصل ۶- نتیجه‌گیری و پیشنهاد برای کارهای آینده ۱۰۹
۶-۱- خلاصه و جمع‌بندی ۱۰۹
۶-۲- پیشنهاد برای کارهای آینده ۱۱۰
فهرست مراجع ۱۱۳
واژه‌نامه فارسی به انگلیسی ۱۱۵

واژه‌نامه انگلیسی به فارسی ۱۱۸
چکیده انگلیسی ۱۲۱

فهرست شکل ها و نمودارها

عنوان **صفحه**

شکل ۲-۱: شماتیک نرون مصنوعی ۹
شکل ۲-۲: تابع عضویت گاوسی برای اعداد نزدیک صفر ۱۳
شکل ۲-۳: مجموعه‌های فازی"بالا"، "متوسط" و "پایین"، برای متغیر زبانی "سرعت ماشین" ۱۵
شکل ۲-۴: قسمت‌های تشکیل دهنده یک سیستم فازی ۱۷
شکل ۲-۵: سیستم‌های عصبی-فازی مشارکتی ۲۱
شکل ۲-۶: سیستم‌های عصبی-فازی همزمان ۲۲
شکل ۲-۷: نحوه ارتباط مقدّم و تالی قواعد اگر-آنگاه برای انواع مختلف سیستم‌های
استنتاج فازی ۲۳
شکل ۲-۸: ساختار شبکه ANFIS ۲۵
شکل ۲-۹: نحوه افراز فضای ورودی توسط NEFCLASS، دایره‌های توخالی داده‌های
کلاس $C1$ و دایره‌های توپر داده‌های کلاس $C2$ را نمایش می‌دهند. ۲۷
شکل ۲-۱۰: ساختار شبکه NEFCLASS ۲۷
شکل ۲-۱۱: نمایش یک ابرجعبه توسط نقاط مینیمم و ماکزیمم ابرجعبه در فضای سه بعدی ۳۱
شکل ۲-۱۲: تابع عضویت ابرجعبه در شبکه FMNN ۳۲
شکل ۲-۱۳: مرز دو کلاس داده به ازای دو مقدار متفاوت γ. الف) γ کوچک ب) γ بزرگ ۳۳
شکل ۲-۱۴: یک حالت خاص همپوشانی بین ابرجعبه‌های j و k و نقاط ماکزیمم و
مینیمم ابرجعبه حاصل از همپوشانی ۳۵
شکل ۲-۱۵: انواع حالات همپوشانی و نحوه برخورد FMNN با آنها ۳۶
شکل ۲-۱۶: الف) ساختار شبکه FMNN ب) ساختار یک نرون ابرجعبه ۳۷
شکل ۲-۱۷: ساختار FMCN ۴۱
شکل ۲-۱۸: تابع عضویت استفاده شده در شبکه GFMN و FMCN ۴۱
شکل ۲-۱۹: مثالی دو بعدی از نحوه عملکرد نرون‌های CCN و OCN ۴۳
شکل ۲-۲۰: نحوه تغییر مقدار عضویت ابرجعبه در ناحیه همپوشانی توسط نرون OCN ۴۴
شکل ۲-۲۱: الف) نرون OCN در شبکه FMCN، ب) نرون CCN در شبکه FMCN ۴۵
شکل ۲-۲۲: فلوچارت الگوریتم آموزش شبکه FMCN ۴۶

شکل ۳-۱: فرایند پیاده‌سازی روی FPGA ۶۳
شکل ۳-۲: فاز تست شبکه MRC-FMMC ۶۹
شکل ۳-۳: ساختار کلی سخت‌افزار MRC-FMMC ۷۰
شکل ۴-۱: ساختار شبکه پیشنهادی ۷۳
شکل ۴-۲: تابع عضویت ابرجعبه، استفاده شده در شبکه پیشنهادی ۷۴
شکل ۴-۳: نحوه جداسازی دو کلاس داده دو بعدی، توسط شبکه پیشنهادی ۷۷
شکل ۴-۴: فلوچارت الگوریتم آموزش روش پیشنهادی ۷۸
شکل ۴-۵: (الف)داده‌های دو بعدی استفاده شده در آزمایش اول، مرز ایجاد شده بین
دو کلاس، توسط شبکه‌های (ب) FMNN، (ج) FMCN و (د)روش پیشنهادی ۸۲
شکل ۴-۶: نتایج حاصل از آزمایش اول روی داده نیم‌دایره ۸۳
شکل ۴-۷: نتایج حاصل از آزمایش دوم روی داده یونوسفر ۸۴
شکل ۴-۸: نتایج حاصل از آزمایش سوم روی داده شیشه ۸۵
شکل ۴-۹: نتایج حاصل از آزمایش چهارم روی داده گل زنبق ۸۶
شکل ۴-۱۰: نتایج حاصل از آزمایش پنجم روی داده WDBC ۸۷
شکل ۴-۱۱: نتایج حاصل از آزمایش ششم روی داده شراب ۸۸
شکل ۵-۱: الف) تابع f در رابطه (۲-۱۷)، ب) تابع g در رابطه (۵-۲) ۹۳
شکل ۵-۲: بلوک دیاگرام کلی سیستم سخت‌افزاری ۹۵
شکل ۵-۳: زیرسیستم تابع عضویت ۹۷
شکل ۵-۴: ساختار داخلی زیرسیستم‌های mf ۹۸
شکل ۵-۵: زیرسیستم محاسبه خروجی یک طبقه ۹۹
شکل ۵-۶: زیرسیستم کلی محاسبات مربوط به فاز تست شبکه ۱۰۰
شکل ۵-۷: زیرسیستم محاسبه خروجی کل ۱۰۱
شکل ۵-۸: بلوک دیاگرام کلی سیستم طراحی شده ۱۰۶
شکل ۶-۱: توزیع داده‌های تست و خطای تست شبکه پیشنهادی در نواحی چهارگانه برای
داده گل زنبق ۱۱۰
شکل ۶-۲: نحوه تصمیم‌گیری شبکه پیشنهادی (الف) داده‌ها و ابرجعبه‌های دو کلاس
۱ و۲ (ب) مرزهای تصمیم‌گیری شبکه پیشنهادی ۱۱۱

فهرست جدول‌ها

عنوان **صفحه**

جدول ۲-۱: انواع روش‌های ادغام منطق فازی با شبکه عصبی ۱۹
جدول ۲-۲: انواع شبکه‌های ممکن، حاصل از فازی‌سازی مستقیم شبکه عصبی منظم ۲۰
جدول ۲-۳: مقادیر عضویت حاصل از ابرجعبه‌ها برای مثال شکل ۲-۱۹ ..... ۴۴
جدول ۲-۴: مقایسه روش‌های عصبی فازی Min-Max با طبقه‌بندهای مرسوم روی داده گل زنبق... ۴۸
جدول ۳-۱: برخی از پیاده‌سازی‌های سخت‌افزاری ANN ها، گزارش شده در مقالات..... ۵۶
جدول ۳-۲: برخی از کاربردهای HNN و انواع HNN استفاده شده برای آن ۵۷
جدول ۳-۳: نتایج حاصل از طبقه‌بندی توسط سخت‌افزار شبکه MRC-FMMC ۷۰
جدول ۴-۱: جدول درستی برای تصمیم گیری شبکه پیشنهادی، در نرون g ۷۶
جدول ۴-۲: مقایسه نتایج حاصله از روش‌های مختلف عصبی فازی Min-Max ۹۰
جدول ۵-۱: نتایج مربوط به دقت تشخیص و فرکانس کاری برای سخت‌افزار طراحی شده..... ۱۰۴
جدول ۵-۲: تعداد منابع استفاده شده در FPGA برای سخت‌افزار طراحی شده ۱۰۵

فهرست اختصارات

اختصار	اصطلاح
ANN	Artifical Neural Networks
ANFIS	Addaptive Neuro-Fuzzy Inference System
ARC	Adaptive Resolution Classifiers
ART	Adaptive Resonance Theory
ASIC	Application Spesific Integrated Cirquit
BP	Back Propagation
BRAM	Block RAM
C	Classifier
CCD	Charge Coupled Devices
CCN	Containment Compensation Neurons
CLB	Configurable Logic Block
CLN	Classifying Neurons
CM	Classification Model
CP	Classification Problem
CPS	Connections-Per-Second
CUPS	Connection-Updates-Per-Second
ETANN	Electrically Trainable Analog Neural Network
FIS	Fuzzy Inference Systems
FL	Fuzzy Logic
FMNN	Fuzzy Min-Max Neural Network
FNN	Fuzzy Neural Network
FPGA	Field Programable Gate Array
FSM	Finite State Machine
GA	Genetic Algoritms
Hb	Hyperbox
HNN	Hardware Neural Network
LUT	Look-Up Table
NEFCLASS	Neuro-Fuzzy Classification
NFIS	Neuro-Fuzzy Inference Systems
OCN	Overlap Compensation Neurons
OCR	Optical Character Recognition
OV	Overlap
SOM	Self Orgnizing Maps
SRL	Shift Register Look-Up Table
TA	Training Algorithm
TDM	Time-Division Multiplexing
WCPS	Watt Per Connection-Per-Second

فهرست علائم و نشانه‌ها

عنوان	علامت اختصاری
اجتماع	$\vee$
اجتماع فازی	S
استلزام فازی	I
اشتراک	$\wedge$
اشتراک فازی	T
پارامتر فازی کنندگی	γ
تابع عضویت	μ
شماره بعد دارای کمترین همپوشانی	Δ
ضریب انبساط	θ
عکس ضریب انبساط	η
فضای جهانی	U
کارایی طبقه‌بند	π
کمترین همپوشانی بین ابرجعبه‌ها	δ
مجموعه داده آموزش	S_{tr}
مجموعه داده تست	S_{ts}
مکمل	$-$
مکمل فازی	C
نقطه ماکزیمم ابرجعبه	W
نقطه مینیمم ابرجعبه	V

فصل اول

مقدمه

۱-۱- پیشگفتار

در دنیای واقعی مسائلی وجود دارند، که هیچ راه‌حل تحلیلی برای حل آنها موجود نیست، یا مسائلی وجود دارند که از لحاظ تئوری حل شده‌اند، ولی به دلیل زمان و منابع عظیمی که برای محاسبه نیاز دارند، حل آنها عملاً غیرممکن است. برای حل اینگونه مسائل، روش‌هایی وجود دارد که از طبیعت الهام گرفته شده‌اند و اغلب به شکل کارا و مؤثری عمل می‌کنند. نتایج حاصل از این روش‌ها اغلب با نتایج تحلیلی برابر نیستند، ولی معمولاً پاسخی نزدیک، که برای مقاصد عملی کافی باشد، را ایجاد می‌کنند. به اینگونه روش‌های الهام گرفته شده از بیولوژیک، محاسبات نرم[۱] گفته می‌شود. سه روش اصلی در محاسبات نرم عبارتند از: شبکه‌های عصبی مصنوعی(ANN)[۲]، الگوریتم‌های ژنتیک(GA)[۳] ومنطق فازی(FL)[۴].

هر تکنیک هوشمند خواص محاسباتی ویژه‌ای دارد، که آن را برای مسائل خاصی مناسب می‌سازد. برای مثال اگر چه ANN برای تشخیص الگو مناسب است، برای توضیح اینکه چگونه تصمیماتش را می‌گیرد مناسب نیست. در مقابل FL که توانایی رویارویی با اطلاعات نادقیق را دارد، برای توضیح تصمیماتش مناسب است. ولی نمی‌تواند قواعد[۵] لازم برای تصمیم‌گیریش را به طور خودکار کسب کند. این محدودیت‌ها نیروی محرکه اصلی برای ایجاد سیستم‌های خبره ترکیبی[۶]، با ترکیب دو یا چند تکنیک مجزا بوده است. همچنین با توجه به حوزه‌های کاربردی مختلف که تکنیک ویژه‌ای را می‌طلبند، سیستم ترکیبی که خواص دو تکنیک را دارا می‌باشد، می‌تواند موثر واقع شود [۱]. به عنوان مثال، مادامی که FL یک مکانیسم استنتاج[۷] تحت ابهام شناختی ارائه می‌دهد، ANN مزایایی

1 - Soft Computing
2 - Artifical Neural Networks
3 - Genetic Algoritms
4 - Fuzzy Logic
5 - Rules
6 - Hybrid
7 - Inference

چون یادگیری، سازگاری، تحمل در برابر خطا، موازی‌کاری و تعمیم[1] را ارائه می‌دهد. برای قادر ساختن یک سیستم برای برخورد با ابهام شناختی با رفتاری مشابه رفتار انسانی، ترکیب ANN و FL می‌تواند ایده خوبی باشد.

ترکیب تکنولوژی‌ها ممکن است منجر به نتایج نامطلوبی شود. بنابراین، این کار باید با دقت صورت گیرد. سیستم‌های ترکیبی زمانی به کار برده می‌شوند که نتایج بهتری را نسبت به استفاده از یک تکنولوژی بدست دهند. هر کدام از این تکنولوژی‌ها راه‌حل‌های کارایی را برای محدوده وسیعی از مسائل در زمینه‌های گوناگون فراهم کرده‌اند. با این حال هر کدام از اینها مزایا و معایبی دارند. بنابراین ترکیب اینها می‌تواند باعث شود، ضعف یکی با قوت دیگری پوشیده شود. ترکیب ANN با FL، به خاطر مزایای خوبی که دارد، در دهه‌های اخیر مورد توجه محققان زیادی قرار گرفته است و ساختارهای متعددی در این زمینه ارائه شده است.

به طور کلی، ترکیب FL و ANN را می‌توان به دو دسته کلی تقسیم کرد. دسته اول شبکه‌های عصبی هستند، که از FL برای بهبود عملکردشان استفاده می‌کنند. مثلا با قاعده بسط(اصل توسعه زاده)[2]، می‌توان ANN را برای پردازش ورودی‌های فازی توسعه داد یا از اپراتورهای توسعه یافته به جای اپراتورهای معمول، استفاده کرد. این دسته را می‌توان شبکه‌های عصبی فازی(FNN)[3] نامید تا تاکید شود که FL برای ارتقای ANN استفاده شده است [۲]. دسته دوم، سیستم‌های استنتاج فازی(FIS)[4] هستند که از ANN برای تنظیم پارامترهایشان استفاده می‌کنند، که در متون تخصصی، تحت عنوان سیستم‌های استنتاج عصبی-فازی (NFIS)[5] نامیده شده‌اند.

یکی از کارهای صورت گرفته در زمینه ترکیب منطق فازی با شبکه عصبی، ترکیب FL با شبکه ART1[6]، برای خوشه‌یابی[1] است [۳]. در سال ۱۹۹۲، آقای Simpson دیدگاه مشابهی را برای حل

1 - Generalization
2 - Zade's Expansion Principle
3 - Fuzzy Neural Network
4 - Fuzzy Inference Systems
5 - Neuro-Fuzzy Inference Systems
6 - Adaptive Resonance Theory

مسائل طبقه‌بندی[۲] به کار گرفت [۴]. شبکه Simpson برای تعیین یک کلاس مجموعه‌ای از ابرجعبه‌ها[۳] را به کار می‌گرفت. هر ابرجعبه در فضای n–بعدی، با دو بردار (دو نقطه n-بعدی) مینیمم و ماکزیمم تعیین می‌شود. هر ابرجعبه نماینده یک مجموعه فازی[۴] بوده و دارای یک تابع عضویت می‌باشد. شبکه Simpson به دلیل استفاده از ابرجعبه‌های فازی تعیین شونده با نقاط مینیمم و ماکزیمم، شبکه عصبی فازی Min-Max (FMNN)[۵]، نام گرفت. شبکه‌های عصبی فازی Min-Max، یکی از تکنیک‌های نسبتاً جدید و کارآمد در حیطه طبقه‌بندی الگو و خوشه‌یابی می‌باشد. این نوع شبکه‌ها از سرعت آموزش بالایی برخوردار بوده و به دلیل یکبارگذر[۶] بودن، می‌توانند در کاربردهای بلادرنگ[۷] مورد توجه قرار گیرند. در فصل ۲ این شبکه‌ها به تفصیل بررسی شده‌اند.

از طرفی برای مقاصد عملی در دنیای واقعی، در حیطه کاربردهای بلادرنگ، سرعت پردازش بالا امری اجتناب ناپذیر است. آرایه گیت برنامه‌پذیر در محل (FPGA)[۸]، به دلیل قابلیت پردازش موازی، الگوریتم‌هایی که قابلیت محاسبه موازی دارند را، با سرعت چندین برابر بیشتر از سریع‌ترین پردازنده‌ها، می‌توانند اجرا کنند [۵].

FPGA یک مدار مجتمع همه منظوره می‌باشد، که به جای برنامه‌ریزی توسط سازنده، توسط خود طراح برنامه‌ریزی می‌شود. بر خلاف مدارهای مجتمع ASIC[۹]، که برای پیاده‌سازی یک تابع خاص به کار می‌روند، یک FPGA، قابلیت برنامه‌ریزی مجدد، حتی پس از به کار رفتن در یک سیستم را، دارا می‌باشد. FPGA های با قابلیت پیکربندی مجدد، منابع قابل برنامه‌ریزی برای پیاده‌سازی و آزمودن طرح‌های مختلف، در زمانی کوتاه را فراهم می‌کنند. آنها ارزان قیمت و در دسترس هستند، دارای

---

1 - Clustering
2 - Classification
3 - Hyperbox
4 - Fuzzy Set
5 - Fuzzy Min-Max Neural Network
6 - One Pass Through
7 - Real Time
8 - Field Programmable Gate Array
9 - Application Specific Integrated Circuit

قابلیت پیکربندی مجدد[1] می‌باشند و انعطاف‌پذیری مشابه نرم‌افزار ارائه می‌دهند. قابلیت پیکربندی مجدد جزئی[2] و برخط[3] در نسل‌های اخیر FPGA ها، مزایای بیشتری را فراهم کرده‌اند [۶].

ذات موازی ANN و قابلیت‌های محاسباتی FPGA، انتخاب این دو برای کاربردهای بلادرنگ را به ما الهام می‌کند.

۱-۲- نوآوری پایان‌نامه

نوآوری‌های صورت گرفته در این پایان‌نامه، هم در بخش طراحی شبکه و هم در بخش پیاده‌سازی سخت‌افزاری، در زیر بیان شده‌اند.

۱-۲-۱- طراحی شبکه

در این پایان‌نامه، یک روش جدید برای طراحی شبکه‌های عصبی فازی بر پایه ابرجعبه‌های فازی Min-Max ارائه شده است. الگوریتم پیشنهادی از آموزش با نظارت[4] برای طبقه‌بندی داده‌های ورودی استفاده می‌کند. همانند شبکه FMNN، این شبکه از تجمیع[5] ابرجعبه‌های فازی برای تعیین مرزهای کلاس‌ها استفاده می‌کند، با این تفاوت که تست همپوشانی و انقباض ابرجعبه‌ها از مراحل آموزش حذف شده است که منجر به الگوریتم آموزش ساده‌تر و سرعت آموزش بیشتر شده است. شبکه ارائه شده از دو دسته ابرجعبه با ضریب انبساط[6] متفاوت، استفاده می‌کند که باعث ایجاد یک ساختار دو طبقه برای شبکه شده است. همچنین این امر باعث می‌شود در مرز کلاس‌ها از ابرجعبه‌های کوچک‌تر، برای افزایش کارایی[7] شبکه، استفاده شود. شبکه با یک بار ارائه داده‌های آموزش، آموزش می‌بینند. نتایج حاصل از شبیه‌سازی‌های رایانه‌ای، بیانگر کارایی خوب این شبکه نسبت به شبکه‌های مشابه،

1 - Reconfigurable
2 - Partial
3 - Online
4 - Supervised
5 - Aggregation
6 - Expansion Coefficient
7 - Performance

می‌باشد. روش پیشنهادی در اکثر موارد نتایج مشابهی نسبت به بهترین روش‌های قبلی را با تعداد نرون‌های کمتر ایجاد کرده است.

۱-۲-۲- پیاده سازی شبکه روی FPGA

در پایان‌نامه حاضر، یک طرح سخت‌افزاری، برای پیاده‌سازی شبکه معرفی شده، روی FPGA ارائه شده است. در این طرح، تابع عضویت ابرجعبه‌ها به منظور صرفه‌جویی در ناحیه مصرفی FPGA، ساده شده است. به طوری که طرح حاصله از ضرب کننده استفاده نمی‌کند. همچنین طرح سخت‌افزاری ارائه شده، به طور موثری از تکنیک خط‌لوله[1] و موازی‌کاری، برای افزایش فرکانس کاری و سرعت مدار، استفاده می‌کند.

۱-۳- ساختار پایان‌نامه

این پایان‌نامه در ۶ فصل تدوین شده است. پس از مقدمه که در این فصل بیان شد. مطالب بیان شده در فصول بعدی به صورت زیر می‌باشند.

در بخش اول فصل ۲ مروری کلی بر ANN صورت گرفته و مزایا و معایب آنها بررسی شده است. در بخش دوم، مفاهیم پایه FL و سیستم‌های فازی معرفی شده است. مفاهیمی مانند مجموعه‌های فازی، عملیات بر روی مجموعه‌های فازی، انواع استنتاج فازی، قسمت‌های تشکیل دهنده سیستم‌های فازی و ... از مباحث بحث شده در این بخش می‌باشند. در بخش سوم ترکیب ANN با FL و دو نوع کلی این ترکیب‌ها یعنی شبکه‌های عصبی فازی و سیستم‌های استنتاج عصبی-فازی، بررسی شده و انواع مختلف هر کدام از این دو نوع، معرفی شده است. در ادامه چند ساختار معروف عصبی فازی که در طبقه‌بندی الگو کاربرد دارند، مانند ANFIS[2] و NEFCLASS[3] معرفی شده‌اند. بخش چهارم این فصل، به شبکه‌های عصبی فازی Min-Max اختصاص یافته است. این بخش، با مقدمه‌ای در مورد سیستم‌های طبقه‌بند و دسته‌بندی این سیستم‌ها با توجه به نوع الگوریتم آموزش، شروع شده است و

1 - Pipeline
2 - Addaptive Neuro-Fuzzy Inference System
3 - Neuro-Fuzzy Classification

در ادامه به معرفی شبکه‌های عصبی فازی Min-Max و دو نوع یکبارگذر این شبکه‌ها یعنی FMNN و FMCN پرداخته شده است.

در بخش اول فصل ۳، به پیاده‌سازی سخت‌افزاری ANN پرداخته شده است. انواع پیاده‌سازی‌ها، مشکلات پیش رو و نحوه ارزیابی سخت‌افزار، از مباحث مطرح شده در این بخش می‌باشد. بخش دوم این فصل، به معرفی FPGA، تکنولوژی‌ها، روند طراحی، کاربردها، مزایا و معایب و... این ادوات، پرداخته است. بخش سوم به تکنیک‌های بهینه‌سازی طراحی FPGA پرداخته است و در بخش آخر، کارهای صورت گرفته در زمینه پیاده سازی شبکه عصبی فازی Min-Max روی FPGA بررسی شده است.

در فصل ۴، روش پیشنهادی برای طراحی طبقه‌بند بر پایه شبکه‌های عصبی فازی Min-Max، معرفی شده است. در ادامه، نتایج شبیه‌سازی‌های رایانه‌ای روی چندین داده آورده شده و کارایی روش ارائه شده با شبکه‌های قبلی مقایسه شده است.

در فصل ۵ پیاده‌سازی شبکه پیشنهادی بر روی FPGA ارائه شده و نتایج شبیه‌سازی‌ها، برای بررسی عملکرد سخت‌افزار طراحی شده، ارائه شده است.

فصل ۶ به جمع‌بندی مطالب ارائه شده و ارائه پیشنهاداتی برای کارهای آینده پرداخته است.

فصل دوم

مروری بر

شبکه‌های عصبی مصنوعی

و

منطق فازی

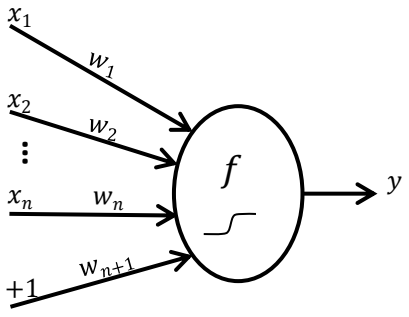
۲-۱- شبکه عصبی مصنوعی

یک شبکه عصبی مصنوعی، گروهی از گره‌های به هم متصل شبیه شبکه نرونی مغز می‌باشد. سیستم پردازش اطلاعات ANN، از سیستم نرون بیولوژیکی الهام گرفته شده است. یک جنبه‌ی کلیدی ANN این است که باید آموزش داده شود. یک ANN مانند سیستم نرونی مغز انسان با مثال‌ها آموزش می‌بیند. در سیستم بیولوژیکی، یادگیری با تنظیم ارتباطات سیناپسی بین نرون‌ها صورت می‌گیرد، این امر برای ANN نیز صادق می‌باشد.

نرون مصنوعی: ANN از تعداد زیادی عناصر محاسباتی به نام نرون تشکیل شده است که با همدیگر برای حل یک مسئله خاص ارتباط برقرار کرده‌اند. یک تک نرون را می‌توان با رابطه (۲-۱) مدل کرد:

$$y = w_{n+1} + \sum_{i=1}^{n} x_i w_i \qquad (۱-۲)$$

در این رابطه، y خروجی نرون، x_i ورودی iام نرون، w_i وزن ضرب شونده در ورودی iام، w_{n+1} مقدار بایاس نرون و n تعداد ورودی‌ها می‌باشد. شماتیک نرون در شکل ۲-۱ نشان داده شده است. در این شکل f، تابع فعال‌سازی یا تابع عبور نرون نامیده می‌شود و عموماً یک تابع سیگموئیدی می‌باشد.



شکل ۲-۱: شماتیک نرون مصنوعی

نرون مصنوعی، چندین ورودی و خروجی دارد. نرون در دو حالت کار می‌کند: حالت استفاده و حالت آموزش. در حالت آموزش نرون یاد می‌گیرد که آیا برای یک ورودی خاص فعال شود یا نه. در حالت

استفاده وقتی الگوی خاصی به ورودی نرون اعمال می‌شود، نرون فعال شده و خروجی متناسب با ورودی را ایجاد می‌کند.

۲-۱-۱- کاربردهای شبکه‌های عصبی

خوشه‌یابی: یافتن تشابهات بین الگوهای ورودی و قراردادن الگوهای مشابه در یک خوشه.

طبقه‌بندی الگو: تخصیص الگوی ورودی به یکی از چندین کلاس خروجی.

تقریب توابع: تخمین تابع ناشناخته‌ی $f(x)$.

پیش‌بینی: پیش‌بینی چندین مقدار آینده‌ی دنباله‌های زمانی.

۲-۱-۲- ساختارهای شبکه‌های عصبی

شبکه‌های پیشخور[1]: شبکه‌هایی که سیگنال‌ها فقط در یک جهت و از ورودی به خروجی جریان می‌یابند. در این شبکه‌ها خروجی یک لایه از خود لایه و لایه‌های بعد، تأثیر نمی‌پذیرد.

شبکه‌های پسخور[2]: یک شبکه پسخور دارای حلقه‌هایی در ساختارش می‌باشد که عبور سیگنال را در دو جهت ممکن می‌سازد. حلقه‌ها شبکه پسخور را بسیار پویا می‌سازند و این شبکه حالت خود را مرتباً تغییر می‌دهد تا به یک حالت پایدار برسد.

۲-۱-۳-روش‌های یادگیری

با نظارت: در این روش، خروجی مطلوب و ورودی، هر دو موجود هستند. خروجی شبکه برای یک ورودی، با خروجی مطلوب مقایسه می‌شود و بر طبق این اختلاف پارامترهای شبکه تنظیم می‌شود.

بدون نظارت[3]: در این روش خروجی‌های مطلوب دردسترس نیست. سیستم باید به خودی خود داده‌های ورودی را سازمان‌دهی کند. به شبکه‌هایی که از این نوع آموزش استفاده می‌کنند، شبکه‌های خودسازمانده گفته می‌شود.

---
1 - Forward
2 - Backward
3 - Unsupervised

۲-۱-۴-مزایا و محدودیت‌های شبکه‌های عصبی

شبکه‌های عصبی مزایایی از قبیل محاسبات موازی، خودسازمانی، تنزل مطبوع[1] (ادامه‌ی عملکرد سیستم درصورت آسیب دیدن قسمتی از آن)، سازگار بودن (با داده‌های جدید سازگار می‌شود و آنها را یاد می‌گیرد)، دارند. در مقابل می‌توان محدودیت‌هایی را برای آنها برشمرد. یک شبکه عصبی صد در صد دقیق نیست و سیستم‌هایی که نیازمند دقت صد در صد هستند، نمی‌توانند از شبکه عصبی استفاده کنند. شبکه‌های عصبی مانند یک جعبه‌سیاه می‌باشند. یعنی امکان کسب اطلاعات از شبکه آموزش دیده وجود ندارد. همچنین برای شبکه‌های بزرگ زمان زیاد آموزش می‌تواند نامطلوب باشد.

۲-۲- منطق فازی

تئوری مجموعه‌های فازی در سال ۱۹۶۵، توسط پرفسور لطفی‌زاده ارائه شد [۷]. منطق فازی بستری فراهم می‌کند تا بتوان با داده‌های نادقیق و تقریبی سروکار داشت. ما در جملات روزمره عباراتی مانند "هوا خوب است" را به کار می‌گیریم، این جمله در واقع نوعی دانش، در مورد وضعیت هوا را بیان می‌کند. ولی هیچ کمیتی برای "خوب" نمی‌توان تعریف کرد و یا آن را اندازه گرفت. ولی مغز انسان این نوع دانش را درک می‌کند و بر اساس آن تصمیم‌گیری می‌کند. منطق فازی می‌تواند دانش بشری را فرموله کند و امکان استفاده از آن را، در سیستم‌ها فراهم آورد. این امر با استفاده از مفهوم مجموعه‌های فازی صورت می‌گیرد. برخلاف مجموعه‌های قطعی[2]، در مجموعه‌های فازی، برای هر عضو مجموعه یک مقدار عضویت بین صفر و یک منسوب می‌شود. در منطق فازی، اگر به مقدار زبانی[3] "خوب" یک مجموعه فازی نسبت دهیم، می‌توانیم متغیر زبانی[4] "هوا" را توسط آن توصیف نماییم.

1 - Graceful Degradation
2 - Crisp
3 - Linguistic Value
4 - Linguistic Variable

۲-۲-۱- تاریخچه

بسیاری از مفاهیم اساسی این تئوری توسط خود لطفی‌زاده تدوین شد. پس از ارائه تئوری فازی، کنترل‌کننده‌های فازی برای کنترل موتور بخار توسط Mamdani و Assilian، در دهه ۱۹۷۰ ساخته شد. در سال ۱۹۸۰، Sugeno دانشمند ژاپنی، کنترل‌کننده‌ای برای کنترل سیستم تصفیه‌ی آب فوجی ارائه داد. دهه‌ی ۱۹۸۰ دهه‌ای بود که مهندسان ژاپنی پیشرفت‌های قابل توجهی در این زمینه ایجاد کردند. همین امر محققان دیگر کشورها را برانگیخت تا به طور جدی به تحقیق در این زمینه بپردازند. پیشرفت‌های چشمگیر این تئوری در اواخر دهه‌ی ۸۰ و اوایل دهه‌ی ۹۰ روی داد [۸].

در تعاریفی که در ادامه می‌آید، U فضای جهانی n-بعدی بوده و $x \in \mathbb{R}^n$ نقطه‌ای در U می‌باشد.

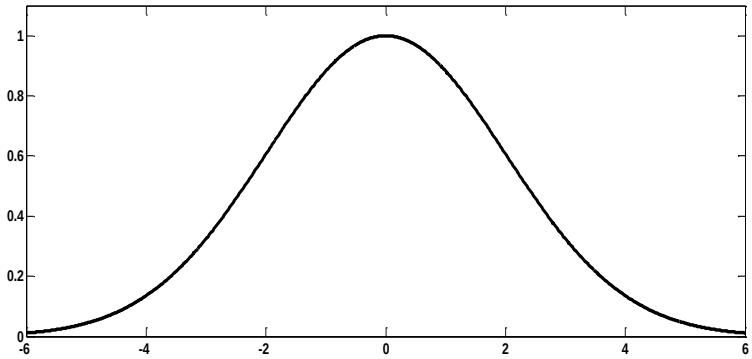
۲-۲-۲-مجموعه فازی

در مجموعه‌های قطعی، یک عنصر یا عضو مجموعه هست و یا عضو آن نیست. ولی در یک مجموعه‌ی فازی، یک عنصر با یک مقدار عضویت در بازه $[0,1]$، عضو مجموعه فازی می‌باشد. مجموعه‌ی فازی A به صورت $A = \{(x, \mu_A(x)) \mid x \epsilon U\}$ تعریف می‌شود. که $\mu_A(x)$ تابع عضویت نامیده می‌شود و برای هر x یک مقدار عضویت نسبت می‌دهد. رابطه (۲-۲) نحوه نمایش مجموعه فازی A را برای فضاهای پیوسته و گسسته نشان می‌دهد.

$$A = \sum_U \frac{\mu_A(x)}{x} \qquad \text{گسسته}$$
$$A = \int_U \mu_A(x)/x \qquad \text{پیوسته} \qquad (۲-۲)$$

در این روابط، $\int$ و $\sum$ نشانه ی انتگرال و جمع نیستند و بیانگر اجتماع x ها، می‌باشند. همچنین /، علامت تقسیم نبوده و بیانگر زوج $(x, \mu_A(x))$ می‌باشد. توابع عضویت متداول به شکل مثلثی یا گاوسی می‌باشند.

برای مثال اگر بخواهیم اعداد نزدیک به صفر را نشان دهیم، می‌توانیم از تابع عضویت گاوسی استفاده کنیم. این تابع عضویت، در شکل ۲-۲ نشان داده شده است.



شکل ۲-۲: تابع عضویت گاوسی برای اعداد نزدیک صفر [۸].

۲-۲-۳- مفاهیم مجموعه‌های فازی

تکیه‌گاه مجموعه فازی A: نقاطی از فضای جهانی U، که مقدار عضویت بزرگتر از صفر دارند.

ارتفاع مجموعه فازی A: بزرگترین مقدار تابع عضویت مجموعه فازی A.

مجموعه فازی طبیعی: مجموعه فازی با ارتفاع یک.

برش α مجموعه فازی A: زیر مجموعه‌ای از U، که مقادیر عضویت عضوهای آن، بزرگتر از α باشد.

مجموعه فازی محدب: مجموعه فازی A محدب است اگر:

$$\forall\, x_1, x_2 \in U\ , \forall\, \lambda \in [0,1]\ \ then\ \ \mu_A[\lambda x_1 + (1-\lambda)x_2] \geq min[\mu_A(x_1), \mu_A(x_2)]$$

۲-۲-۴- عملیات بر روی مجموعه‌های فازی

عملیات اساسی روی مجموعه‌های قطعی را می‌توان به مجموعه‌های فازی اعمال کرد. که در این صورت اجتماع، اشتراک و مکمل اساسی یک مجموعه فازی با تابع عضویت $\mu_A(x)$، مجموعه فازی دیگری را بدست می‌دهد که تابع عضویت آن برای اجتماع، اشتراک و مکمل اساسی با رابطه (۲-۳) بیان می‌شوند.

$$\mu_{A\cup B}(x) = max(\mu_A(x), \mu_B(x))$$
$$\mu_{A\cap B}(x) = min(\mu_A(x), \mu_B(x))$$
$$\mu_{\overline{A}}(x) = 1 - \mu_A(x) \qquad (۲-۳)$$

با تعمیم عملیات اساسی، می‌توان عملیات دیگری روی مجموعه‌های فازی اعمال کرد. این عملیات عبارتند از اجتماع فازی (S-نرم)[1]، اشتراک فازی (T-نرم)[2] و مکمل فازی. رابطه (۲-۴) نحوه تعریف این عملگرها را نشان می‌دهد.

$$\begin{aligned}
&\text{a)} \quad s[\mu_A(x), \mu_B(x)] = \mu_{A\cup B}(x)\ , \\
&\qquad\qquad s(0,a) = s(a,0) = a, s(1,1) = 1\ , \qquad s\colon [0,1]^2 \to [0,1] \\
&\text{b)} \quad t[\mu_A(x), \mu_B(x)] = \mu_{A\cap B}(x)\ , \\
&\qquad\qquad t(1,a) = t(a,1) = a, t(0,0) = 0\ , \qquad t\colon [0,1]^2 \to [0,1] \\
&\text{c)} \quad c[\mu_A(x)] = \mu_{\overline{A}}(x)\ , \\
&\qquad\qquad c(1) = 0\ , c(0) = 1, \qquad c\colon [0,1] \to [0,1]
\end{aligned} \tag{۲-۴}$$

۲-۲-۵- اصل توسعه

اصل توسعه، یک معادله اساسی است که اجازه می‌دهد دامنه یک تابع، از نقاط به مجموعه‌های فازی توسعه پیدا کند [۸]. مثلا اگر $f(x) = x^2$ باشد، و مجموعه فازی "کوچک" به صورت $small = \frac{1}{1} + \frac{1}{2} + \frac{0.8}{3} + \frac{0.6}{4}$ تعریف شده باشد، داریم: $small^2 = \frac{1}{1} + \frac{1}{4} + \frac{0.8}{9} + \frac{0.6}{16}$. حال اگر در این پروسه برای یک عضو دو مقدار عضویت موجود باشد، مقدار بزرگتر را انتخاب می‌کنیم. بنابراین معادله اساسی اصل توسعه به صورت رابطه (۲-۵) تعریف می‌شود که در آن $y = f(x)$.

$$\mu_B(y) = \max_{x\in f^{-1}(y)} \mu_A(x) \tag{۲-۵}$$

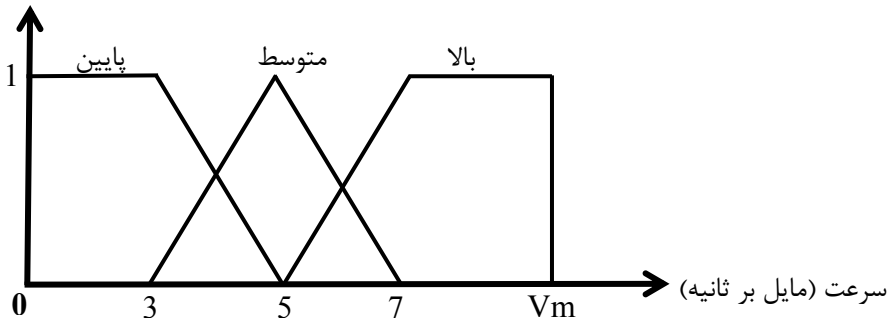
۲-۲-۶- اعداد فازی

عدد فازی، یک مجموعه فازی در R می‌باشد که طبیعی، محدب، دارای تکیه‌گاه کراندار بوده و همه برش‌های α آن در بازه‌های بسته R باشد. اعمالی مانند جمع و ضرب برای اعداد فازی با استفاده از اصل توسعه می‌تواند صورت پذیرد.

1 - S-Norm
2 - T-Norm

۲-۲-۷- متغیر زبانی

اگر یک متغیر بتواند واژه‌هایی از زبان طبیعی را به عنوان مقدار بپذیرد، یک متغیر زبانی نامیده می‌شود. مقادیر زبانی توسط مجموعه‌های فازی در بازه تغییر متغیر، مشخص می‌شود. مثلا اگر بگوییم "سرعت ماشین بالاست"، " سرعت ماشین" متغیر زبانی است که "بالا" را به عنوان مقدار خود پذیرفته است. حال "بالا" را می‌توان توسط یک مجموعه فازی به صورت نشان داده شده در شکل ۲-۳ تعریف کرد.



شکل ۲-۳: مجموعه‌های فازی"بالا"، "متوسط" و "پایین"، برای متغیر زبانی "سرعت ماشین" [۸].

۲-۲-۸- قواعد اگر-آنگاه فازی

در سیستم‌های فازی، دانش بشری به شکل قواعد اگر-آنگاه فازی[1] نشان داده می‌شود. این قواعد به شکل زیر بیان می‌شوند:

اگر <*گزاره‌ی‌فازی*> **آنگاه** <*گزاره‌ی‌فازی*>

گزاره‌های فازی را می‌توان به شکل ساده یا مرکب بیان کرد. مثل:

$X\ is\ S$ گزاره‌های فازی ساده

$(\mathrm{X\ is\ S})\ and\ (X\ is\ not\ F)\ or\ (X\ is\ M)$ گزاره‌های فازی مرکب

که X متغیر زبانی و S, F, M مقادیر زبانی می‌باشند. گزاره‌های مرکب، ترکیب گزاره‌های ساده فازی توسط $not\ ,or\ ,and$ می‌باشند، که این اتصال دهنده‌ها به ترتیب بیانگر اجتماع فازی، اشتراک فازی و مکمل فازی می‌باشند. چون مقادیر زبانی توسط مجموعه‌های فازی مشخص می‌شوند، بنابراین تابع عضویت یک گزاره مرکب را می‌توان با ترکیب توابع عضویت گزاره‌های ساده آن بدست آورد.

1 - Fuzzy if-then Rules

۲-۲-۹-تفسیر قواعد اگر-آنگاه فازی

در منطق کلاسیک گزاره $p \to q$ وقتی درست است که عبارت $(\overline{p} \vee q)$ درست باشد. با تعمیم این روابط به FL، می‌توان قواعد اگر-آنگاه فازی را تفسیر نمود. برای این کار، ما به جای عملگرهای -V,۸, به ترتیب از مکمل فازی، اشتراک فازی و اجتماع فازی استفاده می‌کنیم. در این صورت اگر در رابطه $(\overline{p} \vee q)$ به جای p و q گزاره‌های فازی را جایگزین کنیم، می‌توانیم قواعد اگر-آنگاه فازی را تفسیر کنیم به این نوع تفسیر، استلزام منطقی[۱] می‌گویند. نوع دیگر استلزام، استلزام ممدانی[۲] است که از رابطه $(p \wedge q)$ برای تفسیر قواعد اگر-آنگاه فازی استفاده می‌کند. دو نوع استلزام بیان شده از نوع نگاه افراد به مسائل سرچشمه می‌گیرند. مثلا اگر کسی بگوید: "اگر سرعت بالاست آنگاه مقاومت بالاست" تلویحاً نشان داده‌ایم که "اگر سرعت پایین است آنگاه مقاومت پایین است"، از استلزام منطقی استفاده کرده است. در استلزام ممدانی وقتی می‌گوییم "اگر سرعت بالاست آنگاه مقاومت بالاست"، فقط همین نتیجه قابل بیان است و عبارت دوم را نمی‌توان بیان کرد. نوع استلزام بستگی به نظر افراد خبره دارد. سیستم‌های نوع ممدانی برای تخمین توابع مناسب‌ترند در حالی که سیستم‌های نوع منطقی ممکن است برای مسائل طبقه‌بندی ترجیح داده شوند [۹].

۲-۲-۱۰- استلزام فازی

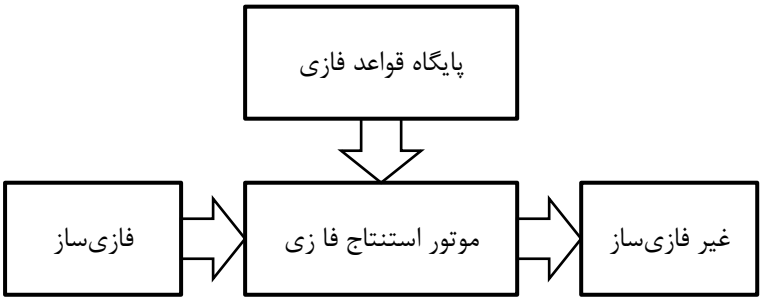
قاعده $Rule: if < P > then < q >$ را در نظر بگیرید. که p با تابع عضویت $\mu_p(x)$ و q، با تابع عضویت $\mu_q(y)$ به ترتیب قسمت مقدّم[۳] و تالی[۴] قاعده فازی می‌باشند. برای بدست آوردن تابع عضویت قانون، از استلزام فازی[۵] استفاده می‌شود که به صورت $\mu_{Rule}(x,y) = I(\mu_p(x), \mu_q(y))$ تعریف می‌شود که در آن $I: [0,1]^2 \to [o,1]$ می‌باشد. استلزام‌های فازی، انواع مختلفی دارند. به

1 - Logical Implication
2 - Mamdani Implication
3 - Antecedent
4 - Consequent
5 - Fuzzy Implication

عنوان مثال: $I(a,b) = S\{1-a,b\} = \max(1-a,b)$، که استلزام کلین-دینس[1] نامیده می‌شود و با جاگذاری max به جای ∨ و مکمل فازی اساسی به جای - در $(\overline{p} \vee q)$ بدست آمده است.

۲-۳- سیستم‌های فازی

سیستم‌های فازی، سیستم‌های مبتنی بر دانش یا قواعد می‌باشد [۸]. این سیستم‌ها از قسمت‌های مختلفی مانند پایگاه قواعد فازی، موتور استنتاج فازی، فازی‌ساز و غیر فازی‌ساز تشکیل شده‌اند. شکل ۲-۴ قسمت‌های مختلف سیستم‌های فازی را نشان می‌دهد.



شکل ۲-۴: قسمت‌های تشکیل دهنده یک سیستم فازی [۸].

۲-۳-۱-پایگاه قواعد فازی

پایگاه قواعد فازی[2]، قلب سیستم‌های فازی را تشکیل می‌دهد. پایگاه قواعد فازی از مجموعه‌ای از قواعد اگر-آنگاه فازی تشکیل می‌شود. این قواعد می‌توانند به طرق مختلفی حاصل شوند. یک روش استفاده از دانش افراد خبره می‌باشد. در این روش، افراد خبره مجموعه‌ای از قواعد اگر-آنگاه را برای عملکرد سیستم وضع می‌کنند. همچنین می‌توان این قواعد را توسط یک شبکه عصبی از روی داده‌های آموزش استخراج کرد. روش دیگر تعیین قواعد از روی داده‌های آموزش، این‌بار با آموزش خود سیستم فازی توسط الگوریتم‌های آموزش شبکه‌های عصبی می‌باشد.

1 - Kleene-Dienes
2 - Fuzzy Rule Base

۲-۳-۲- موتور استنتاج فازی

به زبان ساده می‌توان گفت که موتور استنتاج فازی ورودی سیستم، که در فازی‌ساز[1] فازی شده است، را گرفته و با توجه به قواعد موجود در پایگاه قواعد فازی، خروجی را تولید می‌کند.

۲-۳-۳- فازی‌ساز ها

فازی‌سازها یک نقطه‌ی ورودی را گرفته و آن را به یک مجموعه‌ی فازی نگاشت می‌دهند. مجموعه‌ی فازی حاصل باید در نقطه‌ی ورودی بیشترین مقدار را داشته باشد، بتواند نویز ورودی را کاهش داده و محاسبات موتور استنتاج را ساده‌تر کند. سه فازی‌ساز متداول عبارتند از: فازی‌ساز منفرد، فازی‌ساز گاوسی و فازی‌ساز مثلثی.

۲-۳-۴- غیرفازی‌سازها

غیرفازی‌ساز[2] خروجی فازی موتور استنتاج را، به یک نقطه قطعی نگاشت می‌دهد. غیرفازی‌ساز باید بهترین نقطه‌ای که نماینده‌ی مجموعه‌ی فازی خروجی است را به دست دهد. خروجی غیرفازی‌ساز باید از نظر شهودی نشان دهنده‌ی مجموعه‌ی فازی وارد شده به آن باشد، دارای محاسبات ساده‌ای بوده و پیوستگی داشته باشد. یعنی یک تغییر کوچک در مجموعه‌ی فازی نباید منجر به تغییر بزرگی در نقطه‌ی خروجی شود. غیرفازی‌سازهای متداول عبارتند از: غیرفازی‌ساز مرکز ثقل، غیرفازی‌ساز میانگین مراکز و غیرفازی‌ساز ماکزیمم.

۲-۴- ترکیب شبکه‌عصبی با منطق فازی

از نظر تئوری، شبکه‌های عصبی مصنوعی و سیستم‌های فازی معادل و قابل تبدیل به هم هستند [۸]. با این حال در عمل هر کدام از آنها مزایا و معایب خاص خود را دارند. پروسه آموزش شبکه عصبی نسبتاً کند است و آنالیز شبکه آموزش دیده، سخت است (جعبه سیاه). همانطور که امکان استخراج

1 - Fuzzifier
2 - Defuzzifier

اطلاعات از شبکه عصبی آموزش دیده شده وجود ندارد، امکان دادن اطلاعات ویژه به شبکه، برای تسریع آموزش نیز وجود ندارد. سیستم‌های فازی از این نظر که رفتارشان با قواعد فازی قابل بیان است مطلوبند. اما عموماً بدست آوردن قواعد برای آنها مشکل می‌باشد. ترکیب ANN با FL می‌تواند ضعف‌های یکی را با قوت دیگری پوشش دهد [۱].

۲-۴-۱-شبکه‌های عصبی فازی

شبکه عصبی ساده شکل ۲-۱، با ورودی‌ها، وزن‌ها و خروجی قطعی و تابع فعال‌سازی سیگموئید، شبکه عصبی منظم[1] نامیده می‌شود. رابطه (۲-۱) از ترکیب چند جزء، یعنی سیگنال‌ها (ورودی، خروجی)، وزن‌ها، اپراتورها (جمع، ضرب) و تابع فعال‌سازی تشکیل شده است. شبکه‌های عصبی فازی با تعمیم این اجزا به اعداد، اپراتورها و توابع فازی ایجاد می‌شوند و بسته به اینکه کدام قسمت شبکه شکل ۲-۱ فازی شود، شبکه‌های عصبی فازی مختلفی حاصل می‌شود. جدول ۲-۱ حالت‌های مختلف این نوع شبکه‌ها را نشان می‌دهد.

جدول ۲-۱: انواع روش‌های ادغام منطق فازی با شبکه عصبی [۱].

نام کلی شبکه	سیگنال‌ها	وزن‌ها	تابع فعال‌سازی	اپراتورها
شبکه عصبی منظم	قطعی	قطعی	سیگموئید	اپراتورهای جبری
شبکه عصبی ترکیبی[2]	قطعی	قطعی	هرتابع پیوسته	T-نرم، S-نرم و...
شبکه عصبی فازی منظم[3]	قطعی یا فازی	قطعی یا فازی	سیگموئید	اصل توسعه زاده
شبکه عصبی فازی ترکیبی[4]	قطعی یا فازی	قطعی یا فازی	هرتابع پیوسته	T-نرم، S-نرم و...

۲-۴-۱-۱- شبکه عصبی ترکیبی

برای تحقق این شبکه‌ها نرون‌های فازی[5] توسعه یافتند. از جمله این نرون‌ها می‌توان به نرون فازی AND، نرون فازی OR و نرون فازی $Implication - OR$ اشاره کرد. رابطه (۲-۶) خروجی این نرون‌ها را بدست می‌دهد.

1 - Regular Neural Network
2 - Hybrid Neural Network
3 - Regular Fuzzy Neural Network
4 - Hybrid Fuzzy Neural Network
5 - Fuzzy Neuron

$$AND\ fuzzy\ neuron: y = T(S(x_1, w_1), S(x_2, w_2))$$
$$OR\ fuzzy\ neuron: y = S\big(T(x_1, w_1), T(x_2, w_2)\big)$$
$$Implication - OR\ fuzzy\ neuron: y = S(I(x_1, w_1), I(x_2, w_2)) \quad (۶-۲)$$

۲-۴-۱-۲- شبکه عصبی فازی منظم

با توجه به اینکه در این نوع شبکه، ورودی‌ها و/یا خروجی‌ها و/یا وزن‌ها فازی می‌شوند، چند حالت مختلف پیش می‌آید. انواع حالات ممکن در جدول ۲-۲ آمده است.

جدول ۲-۲: انواع شبکه‌های ممکن، حاصل از فازی‌سازی مستقیم شبکه عصبی منظم [۱].

شبکه عصبی فازی	وزن‌ها	ورودی‌ها	خروجی‌ها
نوع۱	قطعی	فازی	قطعی
نوع۲	قطعی	فازی	فازی
نوع۳	فازی	فازی	فازی
نوع۴	فازی	قطعی	فازی
نوع۵	قطعی	قطعی	فازی
نوع۶	فازی	قطعی	قطعی
نوع۷	فازی	فازی	قطعی

در این جدول، شبکه‌های عصبی فازی از نوع ۱، برای مسائل طبقه‌بندی استفاده می‌شوند. شبکه‌های نوع ۲ و ۳ و ۴ برای پیاده‌سازی قواعد اگر-آنگاه فازی استفاده می‌شوند. شبکه نوع ۵ صحیح نمی‌باشد، چون وزن‌ها و ورودی‌ها قطعی هستند و خروجی باید قطعی باشد. در شبکه‌های نوع ۵ و ۶ نیز فازی‌سازی وزن‌ها لازم نیست چون خروجی مطلوب، قطعی می‌باشد.

۲-۴-۲- سیستم‌های استنتاج عصبی-فازی

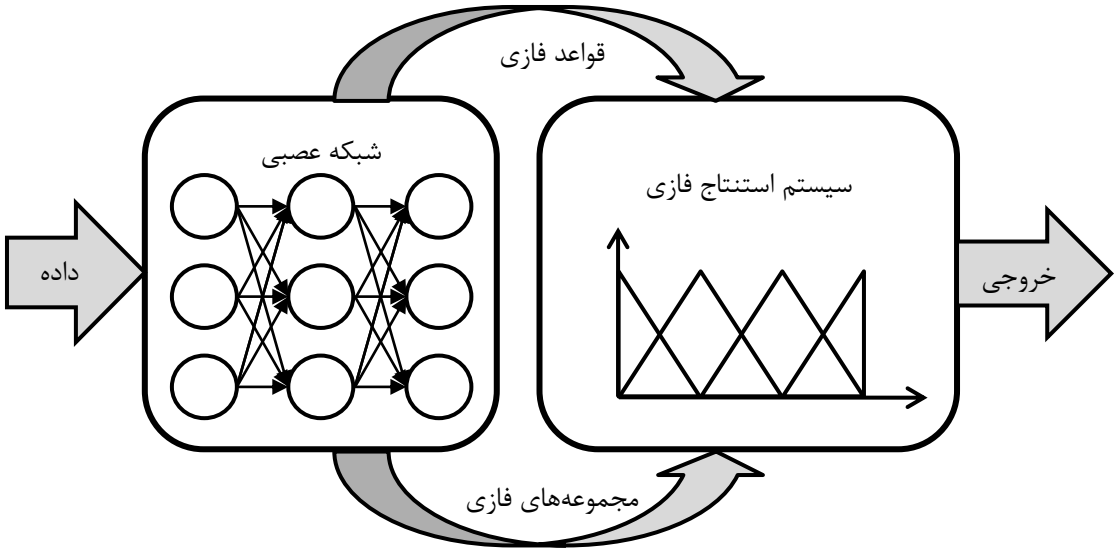
تجمیع شبکه‌های عصبی با سیستم‌های استنتاج فازی به سه دسته تقسیم می‌شود:

۲-۴-۲-۱- سیستم‌های عصبی-فازی مشارکتی

در سیستم‌های عصبی-فازی مشارکتی[1]، خروجی یک تکنیک به عنوان ورودی تکنیک دیگر به کار می‌رود. در واقع تکنیک اول به عنوان پیش‌پردازنده به کار می‌رود. شکل ۲-۵ این نوع سیستم را نشان

1 - Cooperative

می‌دهد. این حالت ساده‌ترین نوع ترکیب می‌باشد. به عنوان مثال مکانیسم آموزش شبکه عصبی، قواعد فازی یا توابع عضویت را از داده‌های آموزش استخراج کرده و به سیستم استنتاج فازی می‌فرستد. حافظه‌های انجمنی فازی[1] و استخراج قواعد فازی توسط SOM[2] مثال‌های خوبی از این دسته می‌باشند [۱۰].

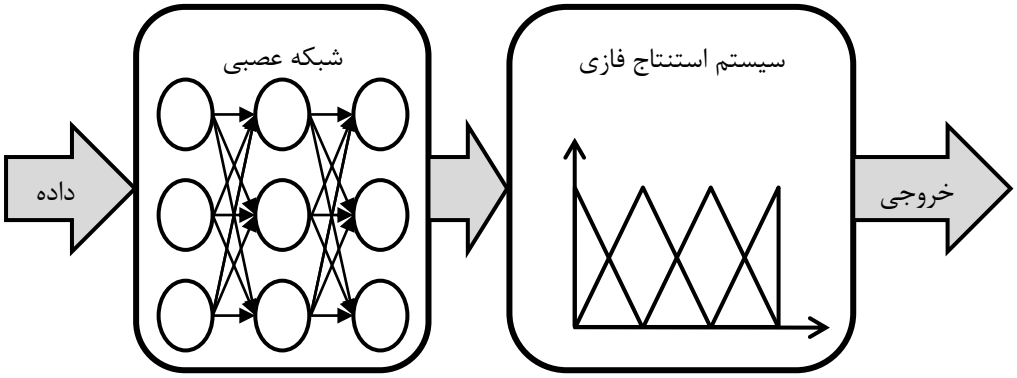


شکل ۲-۵: سیستم‌های عصبی-فازی مشارکتی [۱۰].

۲-۴-۲-۲- سیستم‌های عصبی-فازی همزمان

در سیستم‌های عصبی-فازی همزمان[3]، سیستم فازی به طور مداوم شبکه عصبی را برای تعیین پارامترهای لازم فراخوانی می‌کند (یا بالعکس). این نوع ترکیب سیستم فازی را بهینه نمی‌کند، ولی کارایی کلی سیستم را افزایش می‌دهد. آموزش در شبکه عصبی صورت می‌گیرد و سیستم استنتاج فازی در این مدت بدون تغییر می‌ماند. در برخی موارد نیز ممکن است خروجی سیستم فازی، مستقیماً قابل استفاده نباشد، در این حالت شبکه عصبی می‌تواند به عنوان یک پس پردازشگر استفاده شود. شکل ۲-۶ این نوع سیستم را نشان می‌دهد.

1 - Fuzzy Associative Memories
2 - Self Orgnizing Maps
3 - Concurrent



شکل ۲-۶: سیستم‌های عصبی-فازی همزمان [۱۰].

۲-۴-۲-۳- سیستم‌های عصبی-فازی یکپارچه

در سیستم‌های عصبی-فازی یکپارچه[1]، الگوریتم آموزش شبکه عصبی برای تعیین پارامترهای سیستم استنتاج فازی به کار می‌رود. این سیستم‌ها از قابلیت یادگیری شبکه‌های عصبی و قابلیت توصیف زبان طبیعی سیستم‌های فازی، بهره می‌برند. به خاطر ویژگی‌های خوب این سیستم‌ها، محققان زیادی در این زمینه وارد شده‌اند و ساختارهای متنوعی، با کاربردهایی در کنترل، طبقه‌بندی و... ارائه داده‌اند.

روش معمول برای اعمال الگوریتم آموزش شبکه عصبی به سیستم فازی، این است که سیستم فازی را به شکل یک شبکه عصبی خاص نمایش دهیم. مشکلی که مطرح می‌شود این است که الگوریتم‌های آموزش معمول، مثل گرادیان نزولی، نمی‌توانند مستقیما به چنین سیستمی اعمال شوند. چون توابع استفاده شده در بخش استنتاج، عمدتاً مشتق ناپذیر هستند. این مشکل را می‌توان با استفاد از توابع مشتق‌پذیر در بخش استنتاج و یا عدم استفاده از الگوریتم آموزش استاندارد مرتفع نمود [۱۰].

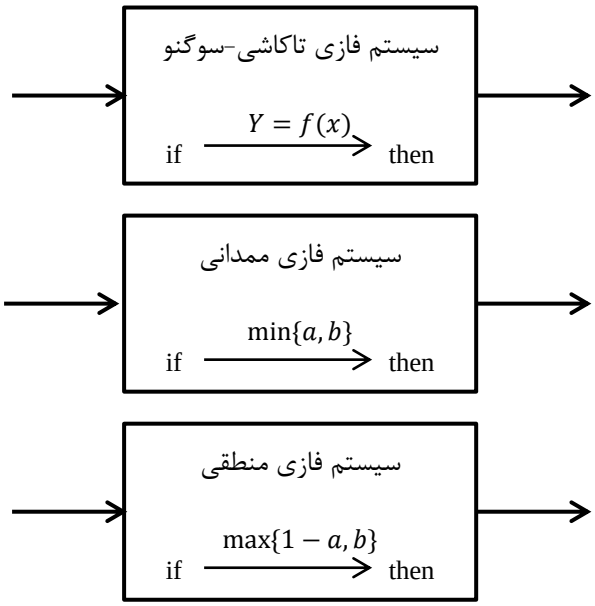
سیستم‌های فازی معمول بر حسب اتصال قسمت مقدّم و تالی قوانین فازی، به سه دسته تقسیم می‌شوند. شکل ۲-۷ این سیستم‌ها و نحوه ارتباط مقدّم و تالی آنها را نشان می‌دهد. در این شکل $b = \mu_q, a = \mu_p$ بیانگر توابع عضویت مقدّم و تالی قانون $if < p > then < q >$ می‌باشند.

سیستم فازی تاکاشی-سوگنو[2]: بخش تالی تابعی از ورودی‌ها می‌باشد.

1 - Integrated
2 - Takagi-Sugeno

سیستم فازی نوع ممدانی[1]: مقدّم ها و تالی‌ها توسط اپراتور min، یا به طور عمومی‌تر T-نرم به هم مرتبط می‌شوند.

سیستم فازی نوع منطقی[2]: مقدّم‌ها و تالی‌ها توسط اپراتور max، یا به طور عمومی‌تر S-نرم به هم مرتبط می‌شوند.



شکل ۲-۷: نحوه ارتباط مقدّم و تالی قواعد اگر-آنگاه برای انواع مختلف سیستم‌های استنتاج فازی

در بخش‌های بعدی، برخی از ساختارهای عصبی فازی استفاده شده در مسائل طبقه‌بندی، آورده شده است.

۲-۴-۳- ANFIS

سیستم استنتاج عصبی-فازی تطبیقی (ANFIS)، جزء اولین سیستم‌های عصبی-فازی، با کاربردهای فراوان در زمینه‌های گوناگون، از جمله طبقه‌بندی الگو می‌باشد [۱۱]. ANFIS سیستم استنتاج تاکاشی-سوگنو را مدل می‌کند. شبکه دارای یک ساختار پنج لایه، به فرم نشان داده شده در شکل ۲-۸ می‌باشد. در این شکل، گره‌های دایروی ثابت می‌باشند و گره‌های مربعی گره‌های تطبیقی شبکه

1 - Mamdani
2 - Logical

را تشکیل می‌دهند. پارامترهای گره‌های تطبیقی با آموزش تنظیم می‌شوند. ساختار لایه‌های مختلف شبکه به قرار زیر است:

لایه اول: این لایه عمل فازی کننده را انجام می‌دهد. در این لایه، فضای هر ورودی به چند بازه تقسیم می‌شود. برای مثال اگر x_2, x_1 به ترتیب بیانگر سرعت و شتاب یک اتومبیل باشند، می‌توان هر کدام از آنها را توسط سه متغیر زبانی "پایین"، "متوسط" و "بالا" توصیف کرد. بنابراین فضای هر ورودی به سه بازه تقسیم می‌شود و هر بازه توسط یک تابع عضویت نمایندگی می‌شود. خروجی این لایه، مقادیر عضویت ورودی‌ها به این توابع عضویت می‌باشند.

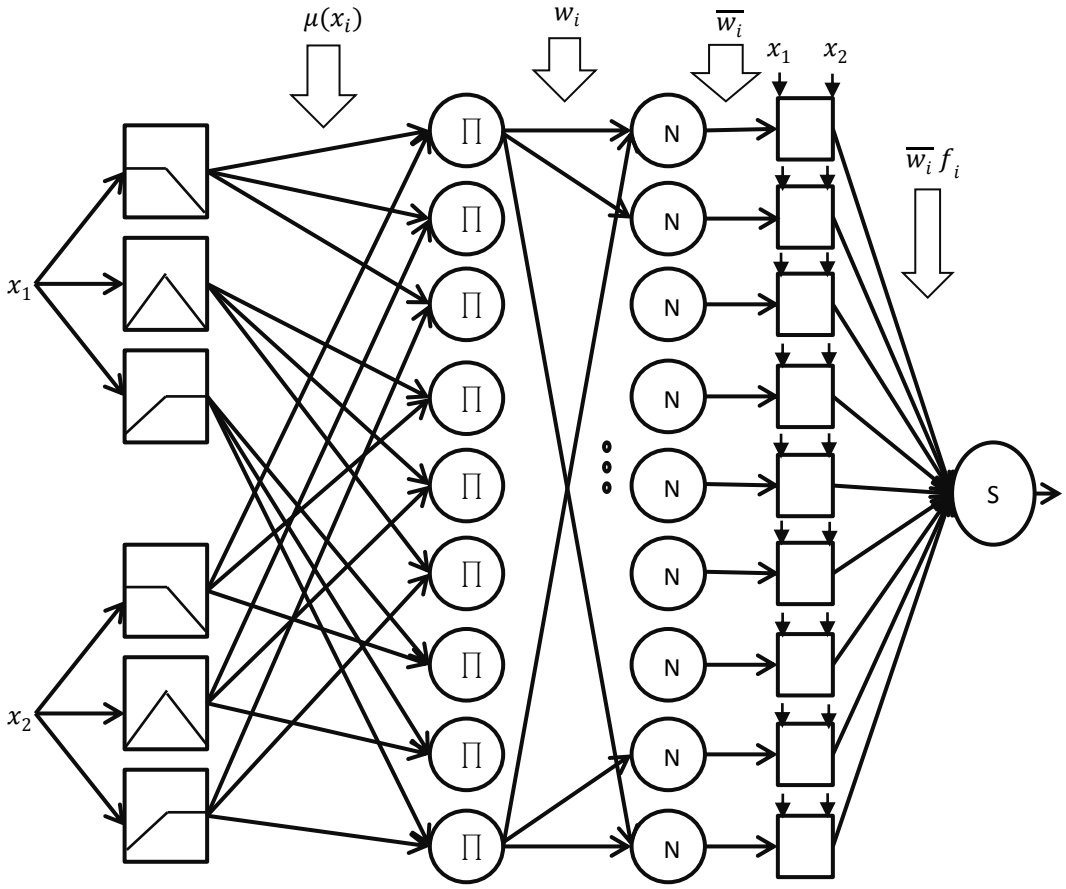
لایه دوم: این لایه قسمت مقدّم قانون فازی را تشکیل می‌دهد. در این لایه، حالت‌های مختلفی که متغیرهای ورودی اختیار می‌کنند، در نظر گرفته می‌شود. برای مثال بالا، ۹ حالت مختلف می‌تواند رخ دهد (سرعت پایین و شتاب پایین، سرعت پایین و شتاب متوسط، ...)، که ۹ قانون فازی را تشکیل می‌دهد. تجمیع مقادیر عضویت ورودی‌ها در این لایه، توسط اشتراک فازی (مثل ضرب) صورت می‌گیرد.

لایه سوم: این لایه، لایه نرمالیزه کننده است. خروجی‌های لایه دوم قدرت آتش[1] قوانین مختلف را بیان می‌کنند. در لایه سوم، نسبت قدرت آتش هر قانون به قدرت آتش کل قوانین سنجیده می‌شود.

لایه چهارم: در این لایه خروجی لایه قبل در تابعی از ورودی‌های شبکه ضرب می‌شوند. برای ANFIS مرتبه صفر این تابع تنها یک عدد a_0 و برای ANFIS مرتبه ۱ به صورت $a_0 + a_1x_1 + \cdots + a_nx_n$ می‌باشد. این لایه قسمت تالی قواعد فازی را تشکیل می‌دهد.

لایه پنجم: این لایه شامل یک نرون است که خروجی کلی سیستم را تعیین می‌کند. کلیه سیگنال‌های خروجی لایه چهارم در این لایه با هم جمع می‌شوند. خروجی نرون این لایه یک عدد قطعی است که حاصل غیرفازی سازی توسط غیرفازی ساز میانگین مراکز می‌باشد.

1 - Firing Strong



شکل ۲-۸: ساختار شبکه ANFIS

۲-۴-۳-۱- آموزش ANFIS

ANFIS برای تنظیم پارامترهایش، از یک الگوریتم ترکیبی استفاده می‌کند. به این صورت که برای تنظیم پارامترهای لایه چهارم، پارامترهای لایه اول ثابت نگه داشته می‌شوند و از الگوریتم کمترین مربعات[1] استفاده می‌شود. در این مرحله شبکه در حالت مستقیم کار می‌کند و سیگنال‌های شبکه خروجی گره‌ها می‌باشند. برای تنظیم پارامترهای لایه اول، پارامترهای لایه چهارم ثابت نگه داشته می‌شوند و از الگوریتم گرادیان نزولی[2] استفاده می‌شود. در این مرحله، شبکه در حالت معکوس کار می‌کند و سیگنال‌های شبکه، سیگنال‌های خطا می‌باشند.

1 - Least Square Method
2 - Gradient Descent

NEFCLASS –۴–۴–۲

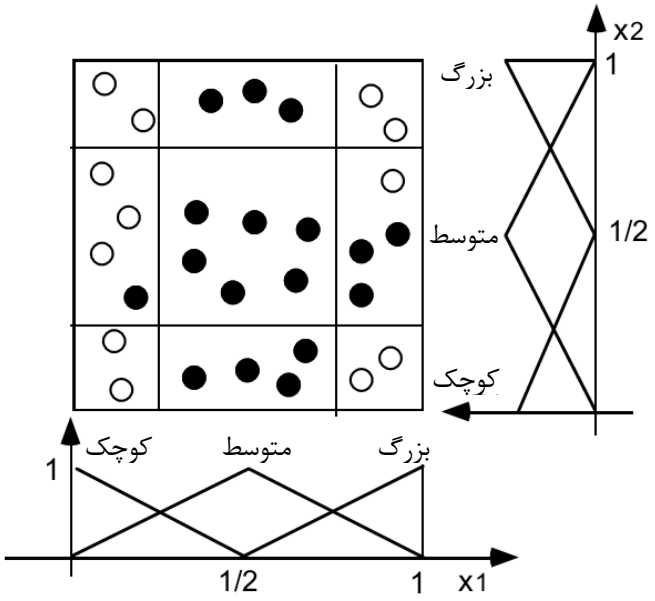
NEFCLASS همانطور که از نامش پیداست، یک طبقه‌بند عصبی-فازی می‌باشد [۱۲]. بدین منظور NEFCLASS از قواعد اگر-آنگاه فازی استفاده می‌کند. این شبکه فضای هر ورودی را به چندین زیر بازه تقسیم کرده و برای هر زیر بازه یک تابع عضویت مثلثی نسبت می‌دهد. شکل ۲-۹ یک مسأله طبقه‌بندی دو بعدی را نشان می‌دهد. در این شکل، هر ورودی با سه مقدار زبانی {کوچک، متوسط و بزرگ} بیان شده است و هر کدام از این مقادیر زبانی با یک تابع عضویت مثلثی بیان شده‌اند. برای طبقه بندی دو کلاس داده، قواعد اگر-آنگاه فازی حاصل از شکل به صورت زیر می‌باشد:

قانون ۱: اگر $x1$ کوچک باشد و $x2$ بزرگ باشد، آنگاه x متعلق به کلاس $C1$ است.
قانون ۲: اگر $x1$ کوچک باشد و $x2$ متوسط باشد، آنگاه x متعلق به کلاس $C1$ است.
قانون ۳: اگر $x1$ کوچک باشد و $x2$ کوچک باشد، آنگاه x متعلق به کلاس $C1$ است.
قانون ۴: اگر $x1$ بزرگ باشد و $x2$ کوچک باشد، آنگاه x متعلق به کلاس $C1$ است.
قانون ۵: اگر $x1$ بزرگ باشد و $x2$ بزرگ باشد، آنگاه x متعلق به کلاس $C1$ است.
قانون ۶: اگر $x1$ متوسط باشد و $x2$ کوچک باشد، آنگاه x متعلق به کلاس $C2$ است.
قانون ۷: اگر $x1$ متوسط باشد و $x2$ متوسط باشد، آنگاه x متعلق به کلاس $C2$ است.
قانون ۸: اگر $x1$ متوسط باشد و $x2$ بزرگ باشد، آنگاه x متعلق به کلاس $C2$ است.
قانون ۹: اگر $x1$ بزرگ باشد و $x2$ متوسط باشد، آنگاه x متعلق به کلاس $C2$ است.

قوانین ۶ و ۷ و ۸ را می‌توان ترکیب کرد و با یک قانون بیان کرد:

قانون: اگر $x1$ متوسط باشد، آنگاه x متعلق به کلاس $C2$ است.

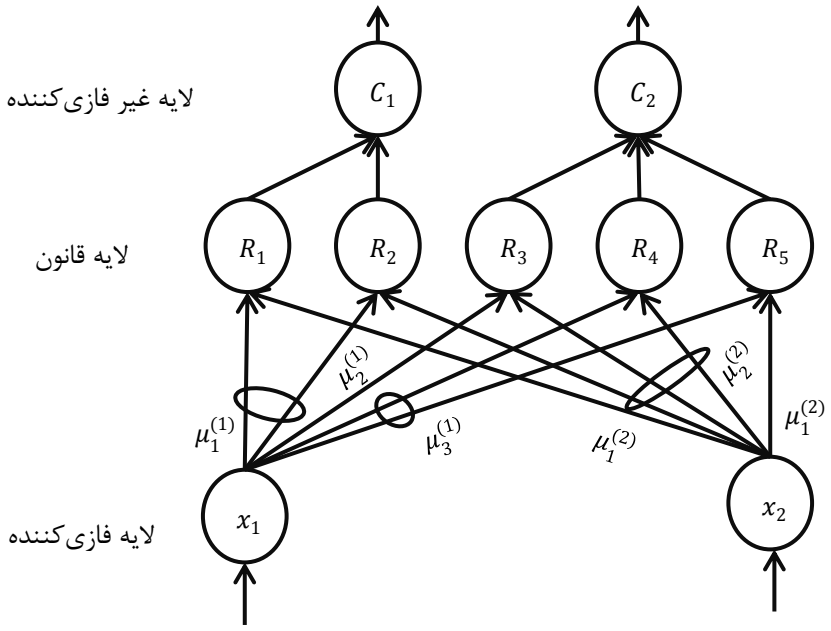
ساختار NEFCLASS یک ساختار سه لایه است که لایه اول لایه ورودی است. لایه دوم لایه قوانین فازی می‌باشد و تعداد نرون‌های این لایه به تعداد قوانین می‌باشد. لایه سوم نیز لایه کلاس‌های خروجی می‌باشد. وزن‌های بین لایه اول و دوم مجموعه‌های فازی می‌باشند. تعداد این مجموعه‌ها به نحوه افراز فضای ورودی بستگی دارد. وزن‌های اتصالات بین لایه دوم و لایه سوم برابر یک می‌باشد. هر قانون فازی در لایه دوم به کلاس مربوط به خود در لایه خروجی وصل می‌شود. NEFCLASS از نرون خروجی برنده برای تعیین کلاس داده ورودی استفاده می‌کند.



شکل ۲-۹: نحوه افراز فضای ورودی توسط NEFCLASS، دایره‌های توخالی داده‌های کلاس $\boldsymbol{C_1}$ و دایره‌های توپر داده‌های کلاس $\boldsymbol{C_2}$ را نمایش می‌دهند. [۱۰].

شکل ۲-۱۰ ساختار NEFCLASS را نشان می‌دهد. به عنوان یک مثال، قانون R1 در این ساختار به صورت زیر توصیف می‌شود:

$$if\ x_1\ is\ \mu_1^{(1)}\ and\ x_2\ is\ \mu_1^{(2)}\ then\ x = (x_1, x_2)\ belongs\ to\ class\ C_1$$



شکل ۲-۱۰: ساختار شبکه NEFCLASS [۱۲].

NEFCLASS می‌تواند با داشتن اطلاعات جزئی در مورد الگوها ایجاد شده و با آموزش، تنظیم شود. یا از همان ابتدا توسط الگوریتم آموزش ایجاد شود. کاربر باید تعداد مجموعه‌های فازی توصیف کننده هر ورودی و همچنین حداکثر تعداد قوانین را تعیین کند. در پروسه آموزش توابع عضویت تنظیم می‌شود. NEFCLASS را می‌توان برای استخراج قواعد از داده‌های عددی به کار برد [۱۰]. از مزایای NEFCLASS آموزش ساده، سادگی، قابلیت تفسیر و قابلیت دادن اطلاعات مفید به شبکه می‌باشد.

۲-۵- شبکه‌های عصبی فازی Min-Max

عموماً وقتی یک شئ را به یک کلاس منسوب می‌کنیم، یک عمل طبقه‌بندی انجام داده‌ایم. در حوزه سیستم‌های خبره، مسائل طبقه‌بندی از اهمیت ویژه‌ای برخوردارند و طراحی طبقه‌بندهای کارآمد همواره از چالش‌های مهم بوده است. یک مساله طبقه‌بندی (CP)[۱]، شامل ایجاد یک سیستم محاسباتی (طبقه‌بند) است که ویژگی‌های خاصی از شئ، که در یک بردار ویژگی قرار دارند را به عنوان ورودی دریافت کرده و آن شئ را به یک کلاس منسوب می‌کند. یک طبقه‌بند توسط زوجهای ورودی-خروجی مجموعه‌ی آموزش (S_{tr}) تعیین می‌شود. به طوری که باید خطای مشخصی روی مجموعه‌ی تست (S_{ts})، کمینه شود. این خطا عموماً توسط درصد انتساب‌های غلط[۲] سنجیده می‌شود. مجموعه‌ی آموزش و تست از زوجهای ورودی-خروجی انتخاب می‌شوند به طوری‌که: $S_{tr} \cap \mathrm{S_{ts}} = \emptyset$. یک سیستم طبقه‌بند متشکل از زوج $< \mathrm{CM} , \mathrm{TA} >$ می‌باشد. TA الگوریتم آموزشی[۳] است که توسط آن، یک طبقه‌بند (C)[۴] از مدل طبقه‌بندی (CM)[۵]، ایجاد می‌شود. معمولاً رفتار TA به مقدار پارامترهای آموزش $\theta = \{\theta_1, \theta_2, \dots, \theta_p\}$ وابسته است، به طوری که با انتخاب یک θ، یک طبقه‌بند خاص با کارایی خاصی روی S_{ts} بدست می‌آید. اگر کارایی طبقه‌بند را با π نشان دهیم، خواهیم داشت: $\pi = \pi(C, S_{ts})$ که در آن $C = C(S_{tr}, TA(\theta))$.

1 - Classification Problem
2 - Misclassification
3 - Training Algorithm
4 - Classifier
5 - Classification Model

با در نظر گرفتن مسأله‌ی نمونه $\overline{\mathrm{CP}}$ از CP (یعنی یک زوج $< S_{tr}, S_{ts} >$)، کارایی تابعی از θ می‌شود: $\pi = \pi(\theta)$. که θ پارامتر تعیین شده توسط کاربر می‌باشد. اگر TA در مقابل تغییرات θ مقاوم نباشد، مشکل جدی در کاربرد عملی سیستم طبقه‌بند ایجاد می‌شود. از طرفی در بعضی از سیستم‌های طبقه‌بند، کاربر علاوه بر تعیین پارامترهای الگوریتم آموزش (θ)، باید پارامترهای مدل را نیز تعیین کند. اگر پارامترهای مدل را با Ω نشان دهیم، کاربر باید $\overline{\Omega}$ بخصوصی را تعیین کند. بنابراین طبقه‌بند تابع $\overline{\Omega}$ می‌باشد. یعنی: $C = CM(\overline{\Omega})$. که در آن $\overline{\Omega} = \Omega(S_{tr}, TA(\theta))$ می‌باشد.

برای مثال درMLP، Ω وزنهای ارتباطی بین نرون‌ها می‌باشد. البته در اینجا فرض شده که پارامترهای ساختاری MLP مانند تعداد لایه‌ها و تعداد نرون‌های هر لایه در ابتدا توسط کاربر تعیین می‌شود. در غیر اینصورت طبقه‌بند به صورت $C = CM(\overline{\Omega}, \overline{\Omega}_s)$ درمی‌آید که $\overline{\Omega}_s$ پارامترهای انتخاب شده از مجموعه‌ی پارامترهای ساختاری Ω_s می‌باشد. انتخاب صحیح $\overline{\Omega}_s$ در کارایی طبقه‌بند بسیار مهم است. مثلا در مورد MLP انتخاب نادرست ممکن است منجر به بیش‌یادگیری[1] یا عدم همگرایی شود.

بنابراین می‌توان گفت که کارایی طبقه‌بند، وابسته به مقادیر پارامترهای آموزش و پارامترهای ساختاری می‌باشد. یعنی: $\pi = \pi(\theta, \Omega_s)$ Ω_s را توسط برخی از الگوریتم‌های بهینه‌سازی می‌توان تعیین کرد. همچنین ایجاد الگوریتم آموزشی که به طور خودکار ساختار طبقه‌بند را تعیین کند ممکن می‌باشد. به چنین الگوریتم‌هایی سازنده[2] می‌گویند. این الگوریتم‌ها قادرند هر دو $\overline{\Omega}, \overline{\Omega}_s$ را تعیین کنند به طوریکه: $C = CM(\overline{\Omega}, \overline{\Omega}_s)$. که در آن $\overline{\Omega} = \Omega(S_{tr}, TA(\theta))$ و $\overline{\Omega}_s = \Omega_s(S_{tr}, TA(\theta))$

از بحث‌های بالا مشخص می‌شود که دو تا از مهم‌ترین ویژگی‌های سیستم‌های طبقه‌بند، قابلیت تعمیم و درجه خودکار بودن[3] آن می‌باشد. یک سیستم طبقه‌بند ایده‌آل باید شرایط زیر را داشته باشد [۱۳]:

۱- الگوریتم آموزش آن سازنده باشد.
۲- تعداد پارامترهای آموزش که در ابتدا توسط کاربر تعیین می‌شود کم باشد.
۳- نسبت به پارامترهای آموزش مقاوم باشد.

---
1 - Over Fitting
2 - Constructive
3 - Automation Degree

البته ارضای شرایط بالا دلیل بر خودکار بودن سیستم طبقه‌بند نیست و اگر چندین سیستم طبقه‌بند دو شرط اول را ارضا کنند، می‌توان درجه‌ی خودکار بودنشان را با در نظر داشتن معیار سوم با هم مقایسه کرد.

به عنوان مثال MLP با الگوریتم آموزش پس‌انتشارخطا (BP)[1]، سازنده نیست چون تعداد نرون‌ها و اتصالات شبکه در ابتدا تعیین می‌شوند. شبکه عصبی-فازی Min-Max دارای الگوریتم سازنده بوده و پارامترهای (تعیین شونده توسط) کاربر اندکی دارد. همین امر شبکه را برای کاربردهای برخط مناسب می‌سازد و دلیل خوبی برای بررسی و توسعه این شبکه می‌باشد.

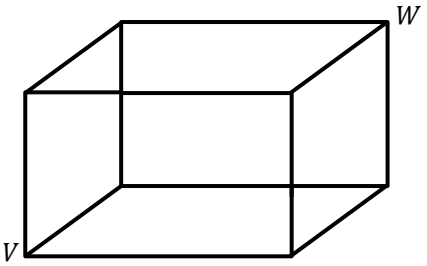
۲-۵-۱- شبکه FMNN

FMNN در سال ۱۹۹۲، توسط آقای Simpson ارائه شد. FMNN برای تعیین یک کلاس مجموعه‌ای از ابرجعبه‌ها را به کار می‌گیرد. هر ابرجعبه در فضای n –بعدی با دو بردار (دو نقطه n -بعدی) مینیمم و ماکزیمم تعیین می‌شود. آموزش شبکه به صورت با نظارت است. Simpson همین دیدگاه را در خوشه‌بندی داده‌ها با آموزش بدون نظارت نیز به کار گرفت [۱۴]. در این نوشته شبکه‌های طبقه‌بند مد نظر می‌باشد. شکل ۲-۱۱، یک ابرجعبه و نحوه تعیین آن با دو بردار V و W (به ترتیب نقاط مینیمم وماکزیمم) را نشان می‌دهد. در فضای سه بعدی، ابرجعبه به شکل مکعب و برای فضای دو بعدی به شکل مستطیل می‌باشد. هر ابرجعبه یک تابع عضویت نیز دارد تا آن را تبدیل به یک مجموعه فازی بکند. نحوه تعریف یک ابرجعبه در FMNN به صورت رابطه (۲-۷) می‌باشد.

$$B_j = \{X_h, V_j, W_j, f(X_h, V_j, W_j)\} \ \forall X_h \in I^n \qquad (۲-۷)$$

در رابطه بالا، X_h نمونه ورودی و f تابع عضویت ابرجعبه می‌باشد.

1 - Back Propagation



شکل ۲-۱۱: نمایش یک ابرجعبه توسط نقاط مینیمم $\boldsymbol{V}$ و ماکزیمم $\boldsymbol{W}$ ابرجعبه در فضای سه بعدی

رابطه (۲-۸)، تابع عضویت استفاده شده برای FMNN طبقه‌بند را نشان می‌دهند. در رابطه مذکور، γ پارامتر فازی‌کنندگی[1] نامیده می‌شود و بیانگر شیب تابع عضویت با دور شدن از ابرجعبه می‌باشد. γ بزرگ بیانگر شیب زیاد (و برعکس) می‌باشد.

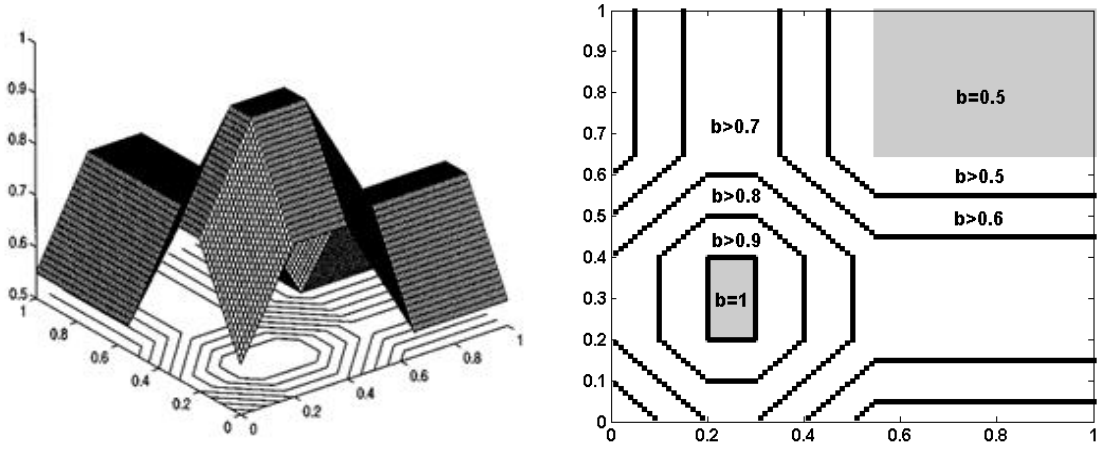
$$b_j(X_h, V_j, W_j) = \frac{1}{2n}\sum_{i=1}^{n}[\max(0,1-\max(0,\gamma\min(1,x_{hi}-w_{ji}))) + \max(0,1-\max(0,\gamma\min(1,v_{ji}-x_{hi})))] \quad (۲-۸)$$

در این رابطه، b_j تابع عضویت ابرجعبه jام، X_h ورودی hام، V_j, W_j، نقاط ماکزیمم و مینیمم ابرجعبه jام، i شماره بعد فضای ورودی و n تعداد ابعاد فضای ورودی می‌باشد.

شکل ۲-۱۲، نحوه تغییر تابع عضویت رابطه (۲-۸)، با دور شدن از ابرجعبه را نشان می‌دهد. همانطورکه در این شکل دیده می‌شود، نقاط داخل ابرجعبه مقدار عضویت یک دارند و با دور شدن از ابرجعبه، این مقدار عضویت کاهش می‌یابد تا به کمترین مقدار عضویت حاصل از رابطه (۲-۸)، یعنی مقدار ۰٫۵ برسد.

هر ابرجعبه دارای یک برچسب[2] می‌باشد که نشان دهنده کلاس داده‌های داخل آن است. FMNN با تجمیع ابرجعبه‌های با برچسب یکسان، مرزهای یک کلاس را تعیین می‌کند. شکل ۲-۱۳ تأثیر اندازه γ در مرز ایجاد شده بین دو کلاس داده را نشان می‌دهد. با کاهش γ، ابرجعبه نقاط دورتری را تحت پوشش قرار می‌دهد.

1 - Fuzziness Control Parameter
2 - Label

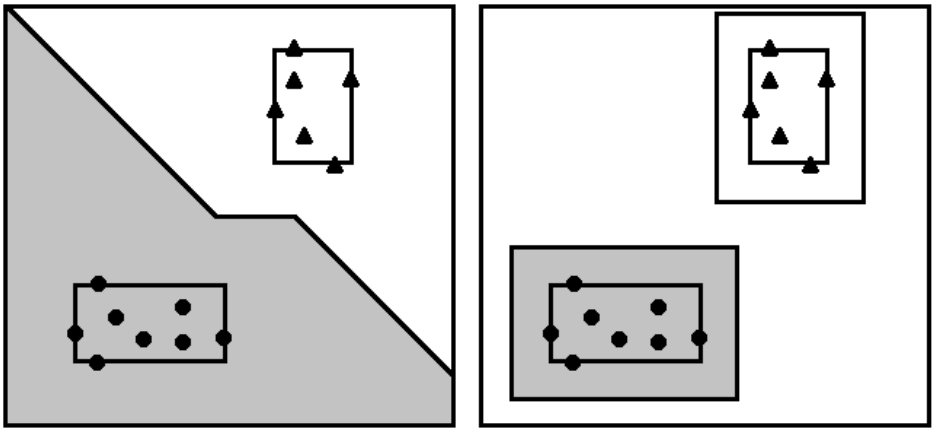


(الف) (ب)

شکل ۲-۱۲: تابع عضویت ابرجعبه در شبکه FMNN به ازای $V = (0.2,0.2), W = (0.3,0.4), \gamma = 4$، الف) نمودار سه‌بعدی تابع عضویت و ب) منحنی‌های حاصل از برش مقطعی (الف) [۱۵].

آموزش طبقه‌بند FMNN به این صورت است که با ارائه هر داده آموزش، الگوریتم دنبال ابرجعبه‌ای می‌شود که بتواند با انبساط آن، داده را در خود جای دهد. اگر چنین ابرجعبه‌ای پیدا شود، انبساط یافته و داده آموزش را در بر می‌گیرد. پس از انبساط، بررسی می‌شود که ابرجعبه منبسط شده با ابرجعبه‌های سایر کلاس‌ها همپوشانی نداشته باشد. اگر همپوشانی ایجاد شود دو ابرجعبه همپوشان منقبض می‌شوند تا از هم مجزا گردند. اگر هیچ ابرجعبه‌ای برای جای دادن داده جدید یافت نشود، ابرجعبه جدیدی ایجاد می‌شود.

پروسه انبساط با یک پارامتر تعیین شده توسط کاربر به نام ضریب انبساط θ کنترل می‌شود. این ضریب بیانگر بیشینه اندازه مجاز ابرجعبه می‌باشد. بنابراین الگوریتم آموزش FMNN، سه مرحله انبساط، تست همپوشانی و انقباض ابرجعبه‌ها را شامل می‌شود. الگوریتم آموزش FMNN یک الگوریتم سازنده است و شبکه در حین آموزش ساخته می‌شود. بنابراین کارایی شبکه وابسته به ترتیب ارائه داده‌های آموزش می‌باشد، که این مساله از عیوب این نوع شبکه‌ها می‌باشد. مزیت این شبکه یکبارگذر بودن آن است، یعنی داده‌های آموزش فقط یکبار به شبکه ارائه می‌شوند. همین امر باعث می‌شود تا سرعت آموزش بسیار زیاد شود و شبکه را برای کاربردهای برخط مناسب سازد.



(ب) (الف)

شکل ۲-۱۳: مرز دو کلاس داده به ازای دو مقدار متفاوت $\boldsymbol{\gamma}$. الف) $\boldsymbol{\gamma}$ کوچک ب) $\boldsymbol{\gamma}$ بزرگ [۱۷].

۲-۵-۲- آموزش FMNN

مراحل الگوریتم آموزش شبکه FMNN به صورت زیر می‌باشد:

با ارائه اولین داده آموزشی، یک ابرجعبه ایجاد می‌شود و نقاط ماکزیمم و مینیمم ابرجعبه، طبق رابطه (۲-۹) برابر داده آموزشی قرار داده می‌شود.

$$v_{1i} = w_{1i} = x_{hi} \quad \forall\, i = 1,2,\ldots,n \tag{۹-۲}$$

با ارائه داده‌های آموزشی بعدی، الگوریتم مراحل انبساط، تست همپوشانی و انقباض را به صورت گفته شده در زیر دنبال می‌کند:

۲-۵-۲-۱- انبساط

با ارائه داده آموزش جدید، الگوریتم دنبال ابرجعبه‌ای می‌گردد که بتواند با انبساط یافتن، داده جدید را در داخل خود جای دهد. ابرجعبه انتخاب شده باید دو شرط زیر را دارا باشد:

۱- دارای بیشترین درجه عضویت باشد.

۲- در صورت انبساط، اندازه آن از اندازه مجاز بیشینه θ تجاوز نکند (طبق رابطه (۲-۱۰)).

$$n\theta \geq \sum_{i=1}^{n} \left(\max\left(w_{ji}, x_{hi}\right) - \min\left(v_{ji}, x_{hi}\right)\right) \tag{۱۰-۲}$$

اگر ابرجعبه‌ای دو شرط بالا را ارضا کرد، نقاط مینیمم و ماکزیمم جدید ابرجعبه، طبق رابطه (۲-۱۱) محاسبه می‌شوند.

$$v_{ji}^{new} = \min\left(v_{ji}^{old}, x_{hi}\right) \qquad \forall\, i = 1,2,\dots,n$$
$$w_{ji}^{new} = \max\left(w_{ji}^{old}, x_{hi}\right) \qquad \forall\, i = 1,2,\dots,n \qquad (۱۱-۲)$$

۲-۵-۲-۲- تست همپوشانی

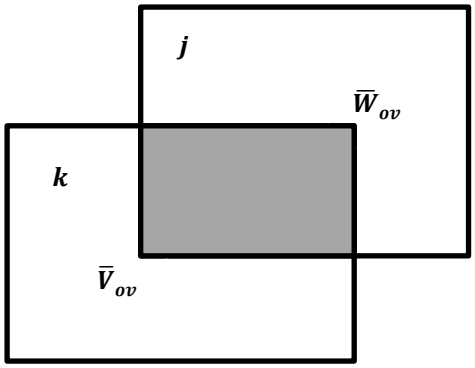
فرض کنید ابرجعبه ***j***ام مربوط به کلاس ۱، منبسط شده است و می‌خواهیم بررسی کنیم که آیا با ابرجعبه ***k***ام که مربوط به کلاس ۲ است، همپوشانی دارد یا نه. FMNN برای این کار چهار تست مختلف روی هر بعد انجام می‌دهد. این تست‌ها در رابطه (۲-۱۲) آورده شده‌اند. این چهار تست حالت‌های مختلف همپوشانی را بررسی می‌کنند. برای هر بعد یکی از این حالات رخ می‌دهد. اگر بعدی وجود داشته باشد که برای آن، هیچ یک از چهار حالت رخ ندهد، دو ابرجعبه مجزا هستند و تست همپوشانی، با بررسی ابرجعبه بعدی ادامه می‌یابد. در حین تست همپوشانی، شماره بعدی که کمترین همپوشانی را دارد و همچنین نوع همپوشانی (یکی از موارد ۱ تا ۴ در رابطه (۲-۱۲))، ذخیره می‌شوند تا در مرحله انقباض مورد استفاده قرار گیرند. برای این کار از دو متغیرΔ و $\boldsymbol{\delta}$ استفاده می‌شود. که Δ شماره بعد دارای کمترین همپوشانی را ذخیره می‌کند و $\boldsymbol{\delta}$ کمترین مقدار همپوشانی را نگه می‌دارد. مقدار اولیه Δ برابر ۱- و مقدار اولیه $\boldsymbol{\delta}$ برابر ۱ می‌باشد.

$$\boldsymbol{case1}: v_{ji} < v_{ki} < w_{ji} < w_{ki}$$
$$\delta^{new} = \min(w_{ji} - v_{ki}, \delta^{old})$$
$$\boldsymbol{case2}: v_{ki} < v_{ji} < w_{ki} < w_{ji}$$
$$\delta^{new} = \min(w_{ki} - v_{ji}, \delta^{old})$$

$$\boldsymbol{case3}: v_{ji} < v_{ki} < w_{ki} < w_{ji}$$
$$\delta^{new} = \min(\min(w_{ji} - v_{ki}, w_{ki} - v_{ji}), \delta^{old})$$
$$\boldsymbol{case4}: v_{ki} < v_{ji} < w_{ji} < w_{ki}$$
$$\delta^{new} = \min(\min(w_{ji} - v_{ki}, w_{ki} - v_{ji}), \delta^{old}) \qquad (۱۲-۲)$$

شکل ۲-۱۴ یک حالت خاص همپوشانی را نشان می‌دهد. در این شکل نقاط مربوط به ماکزیمم ومینیمم ناحیه همپوشان نشان داده شده است. در حالت برداری می‌توان با بدست آوردن این نقاط (رابطه (۲-۱۳)-a) و بررسی این که آیا این دو نقطه تشکیل ابرجعبه می‌دهند یا نه (رابطه (۲-۱۳)-b)، به وجود همپوشانی پی برد.

$$a)\ \bar{V}_{ov} = \max(\bar{V}_j\ , \bar{V}_k)$$
$$\bar{W}_{ov} = \min(\bar{W}_j\ , \bar{W}_k)$$
$$b)\ if\ \bar{V}_{ov} \le \bar{W}_{ov}\ then\ overlap\ occured \qquad (۲-۱۳)$$



شکل ۲-۱۴: یک حالت خاص همپوشانی بین ابرجعبه‌های $\boldsymbol{j}$ و $\boldsymbol{k}$ و نقاط ماکزیمم و مینیمم ابرجعبه حاصل از همپوشانی

۲-۵-۲-۳- انقباض

در مرحله انقباض، ابرجعبه‌ها در جهتی که کمترین کاهش اندازه را داشته باشند منقبض می‌شوند و بر حسب اینکه چه نوع همپوشانی (طبق رابطه (۲-۱۳)) بین ابرجعبه‌های jام و kام رخ داده باشد، مرزهای ابرجعبه‌های همپوشان جابجا می‌شوند. این کار توسط رابطه (۲-۱۴) صورت می‌گیرد.

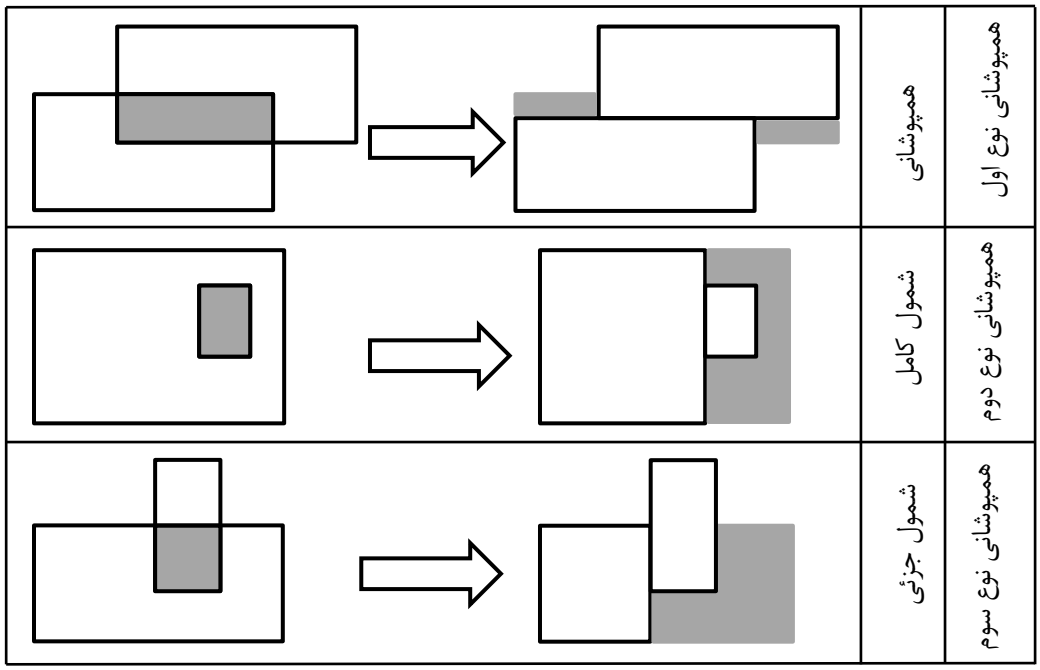
$$\boldsymbol{case1}:\ v_{k\Delta}^{new} = w_{j\Delta}^{new} = (v_{k\Delta}^{old} + w_{j\Delta}^{old})/2$$

$$\boldsymbol{case2}:\ v_{j\Delta}^{new} = w_{k\Delta}^{new} = (v_{j\Delta}^{old} + w_{k\Delta}^{old})/2$$

$$\boldsymbol{case3}:\ if\ (w_{k\Delta} - v_{j\Delta} < w_{j\Delta} - v_{k\Delta})$$
$$then\ (v_{j\Delta}^{new} = w_{k\Delta}^{old})\ else\ (w_{j\Delta}^{new} = v_{k\Delta}^{old})$$

$$\boldsymbol{case4}:\ if\ (w_{k\Delta} - v_{j\Delta} < w_{j\Delta} - v_{k\Delta})$$
$$then\ (w_{k\Delta}^{new} = v_{j\Delta}^{old})\ else\ (v_{k\Delta}^{new} = w_{j\Delta}^{old}) \qquad (۲-۱۴)$$

شکل ۲-۱۵، حالت‌های مختلف همپوشانی و روش برخورد FMNN با آنها را نشان می‌دهد. همانطور که در این شکل دیده می‌شود، در مرحله انقباض اطلاعات مربوط به نواحی خاکستری از دست می‌رود و این امر، باعث ایجاد خطای طبقه‌بندی می‌شود.



شکل ۲-۱۵: انواع حالات همپوشانی و نحوه برخورد FMNN با آنها [۱۶]

۲-۵-۳- ساختار شبکه FMNN

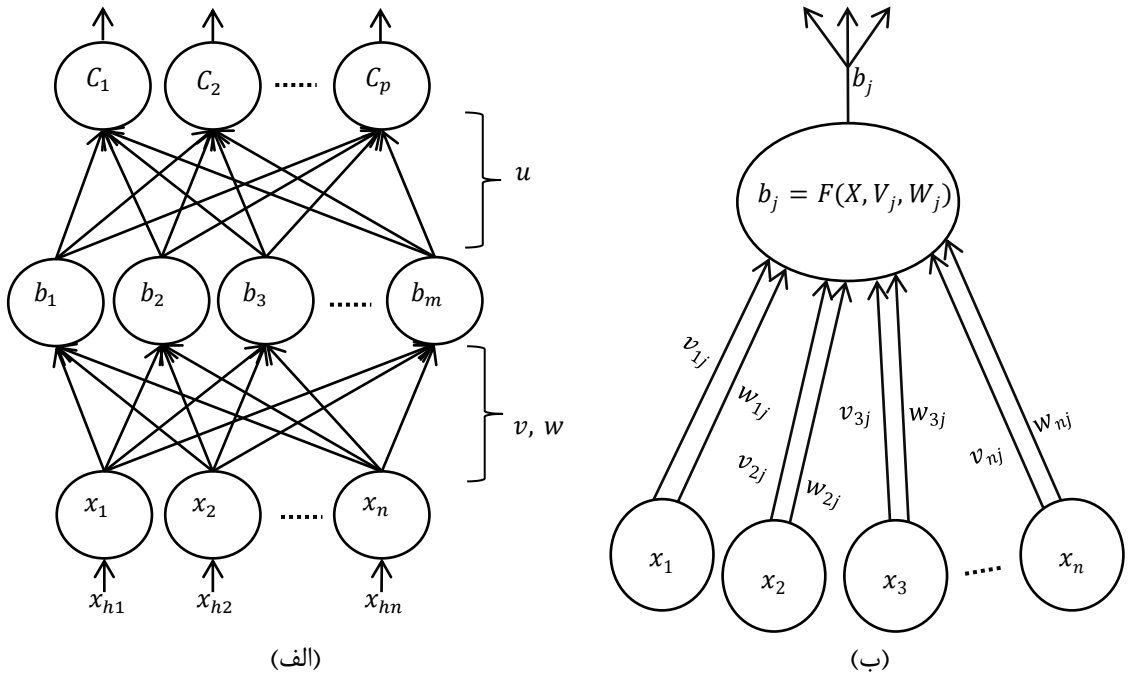
پس از آموزش، ساختار شبکه، یعنی تعداد نرون‌ها (ابرجعبه‌ها) و نحوه اتصالات آنها مشخص می‌شود. FMNN دارای ساختار سه لایه به صورت نشان داده شده در شکل ۲-۱۶-الف می‌باشد. در این شکل، لایه اول لایه ورودی بوده و بردار ورودی را روی لایه دوم پخش می‌کند. لایه دوم لایه ابرجعبه می‌باشد. خروجی هر نرون این لایه، مقدار عضویت ورودی نرون به آن ابرجعبه می‌باشد. لایه سوم در بر دارنده گره‌های کلاس می‌باشد. تعداد نرون‌های این لایه به تعداد کلاس‌ها می‌باشد. FMNN در این لایه از استراتژی نرون برنده استفاده می‌کند. یعنی خروجی لایه ۲ را گرفته و ماکزیمم آن را به خروجی نرون می‌دهد. رابطه خروجی حاصل از نرون kام لایه آخر را نشان می‌دهد.

$$C_k = \max_{j=1,\dots,m} (b_j u_{kj}) \qquad k = 1, \dots, p\,, \quad j = 1,2, \dots ., m \tag{۱۵-۲}$$

در رابطه بالا، u_{kj} وزن ارتباطی بین ابرجعبه jام و نرون کلاس kام، b_j ابرجعبه jام، C_k نرون کلاس kام، p تعداد کلاس‌ها و m تعداد ابرجعبه‌ها می‌باشد. u_{kj} نیز به صورت رابطه بیان می‌شود.

$$u_{kj} = \begin{cases} 1, & if\ \{b_j \in C_k\} \\ o, & if\ \{b_j \notin C_k\} \end{cases} \qquad (۱۶\text{-}۲)$$

وزن‌های ارتباطی بین لایه دو و لایه سه، صفر یا یک است. به طوری که ابرجعبه‌های مربوط به یک کلاس، به نرون آن کلاس در خروجی وصل باشند (رابطه (۲-۱۶)). هر نرون لایه اول با دو اتصال وزن‌دار، به هر نرون لایه دوم متصل است. این اتصالات، در بر دارنده نقاط ماکزیمم و مینیمم ابرجعبه می‌باشند. شکل ۲-۱۶-ب یک نرون ابرجعبه و وزن‌های ورودی آن را نشان می‌دهد.



شکل ۲-۱۶: الف) ساختار شبکه FMNN ب) ساختار یک نرون ابرجعبه [۴].

علاوه بر وابستگی ساختار شبکه به ترتیب ارائه ورودی‌ها، یک عیب عمده FMNN این است که کارایی آن وابسته به مقدار θ می‌باشد. θ بزرگ منجر به شبکه ساده‌تر با کارایی کم و θ کوچک منجر به تعداد ابرجعبه‌های بیشتر و شبکه پیچیده‌تر می‌شود، ولی در عوض کارایی شبکه افزایش می‌یابد [۱۶].

۲-۵-۴- بهبود FMNN

الگوریتم‌های دیگری پس از FMNN برای رفع اشکالات آن ارائه شدند. برخی از این الگوریتم‌ها، با انتخاب اتوماتیک θ سعی در بهبود رفتار FMNN داشتند [۱۶]. در [۱۵] یک الگوریتم بازگشتی[1]، به نام GFMN ارائه شده است که با یک مقدار بزرگ θ شروع کرده و رفته رفته آن را کاهش می‌دهد تا به کارایی بهینه برسد. یک حد پایین θ برای جلوگیری از بیش‌یادگیری در نظر گرفته می‌شود. FMNN برای طبقه‌بندی و خوشه‌یابی دو روش مجزا ارائه می‌دهد. ایده به کار رفته در GFMN شبکه عمومی‌تری را ارائه می‌دهد که قادر به آموزش بانظارت، بدون نظارت و یا ترکیبی از این دو (برای داده‌هایی که برخی نمونه‌ها برچسب ندارند) می‌باشد. همچنین بر خلاف FMNN، که نقاط ورودی نقاط قطعی هستند، نقاط ورودی در روش GFMN[2]، می‌توانند به صورت ابرجعبه‌های فازی نیز باشند. در این روش شرط مربوط به بیشینه اندازه ابرجعبه، به تک تک ابعاد اعمال می‌شود و همچنین، یک تابع عضویت جدید برای ابرجعبه‌ها معرفی شده است. که در ادامه، این تابع عضویت معرفی شده است. در [۱۳] یک الگوریتم سازنده برای FMNN معرفی شده است که یک تابع هدف را برای کارایی بهینه شبکه، در یک پروسه بازگشتی کمینه می‌کند، تا به تعادل مناسبی بین تعداد ابرجعبه‌ها و دقت طبقه‌بندی دست یابد.

برخی دیگر از روش‌ها برای رهایی از مقداردهی اولیه θ و وابستگی به ترتیب ارائه ورودی‌ها از آموزش برون‌خط[3]، با دریافت کل داده‌های آموزش در ابتدا، استفاده می‌کنند [۱۶]. در [۱۷]، Abe و Lan یک الگوریتم طبقه‌بندی بازگشتی با مفهوم ابرجعبه‌های فعال‌ساز[4] و بازدارنده[5] ارائه کرده‌اند. ابرجعبه فعال‌ساز شامل داده‌هایی از یک کلاس و ابرجعبه بازدارنده شامل داده‌هایی از چندین کلاس می‌باشند. ترکیبی از این ابرجعبه‌ها قواعد فازی را برای طبقه‌بندی تعیین می‌کنند. Meneganti و همکارانش در

1 - Recursive
2 - General Fuzzy Min-Max Neural Network
3 - Offline
4 - Activation
5 - Inhibition

[۱۸] الگوریتمی ارائه دادند که با ابرجعبه‌هایی به تعداد کلاس‌های خروجی شروع می‌کند که هر ابرجعبه شامل کل داده‌های آن کلاس می‌باشد. و برحسب همپوشانی ابرجعبه‌ها، تجزیه، ترکیب، حذف و انبساط ابرجعبه‌ها شکل می‌گیرد. در [۱۹]، Rizzi و همکارانش یک الگوریتم آموزش بازگشتی متفاوت به نام طبقه‌بند بادقت تطبیقی ARC[1] ارائه داده‌اند. این الگوریتم که از "برش ابرجعبه" استفاده می‌کند با یک ابرجعبه ترکیبی (شامل چند کلاس)، که شامل کل داده‌های آموزش است شروع کرده و به صورت بازگشتی، ابرجعبه‌های ترکیبی را برش می‌دهد تا به ابرجعبه‌های خالص (شامل یک کلاس) برسد.

در روش‌های بررسی شده، بهینه‌سازی از طریق تکرار حاصل شده است. بنابراین این الگوریتم‌ها مزیت یکبارگذر بودن و برخط بودن را از دست می‌دهند. Nandedkar و Biswas در [۱۶]، الگوریتم FMCN[2] را ارائه کرده‌اند که مزیت برخط بودن و یکبارگذر بودن را حفظ کرده است و از کارایی خوبی نسبت به الگوریتم‌های مشابه برخوردار می‌باشد.

۲-۵-۵- شبکه FMCN

FMCN در سال ۲۰۰۷ توسط Nandedkar و Biswas ارائه شد. این روش نسبت به روش‌های قبلی کارایی خیلی بهتری دارد و قابلیت تعمیم بالایی بدست می‌دهد. ورودی‌ها از نوع قطعی می‌باشند (البته در سال ۲۰۰۹ همین مؤلفین ساختار عمومی این روش را نیز ارائه کردند که از ابرجعبه‌های قطعی به عنوان ورودی استفاده می‌کند و قادر به طبقه‌بندی، خوشه‌یابی و یا ترکیبی از این دو می‌باشد [۲۰].) و ساختار شبکه با یکبار ارائه داده‌های آموزش ایجاد می‌شود. این الگوریتم مرحله انقباض ابرجعبه‌ها را حذف کرده است. در عوض نواحی همپوشانی ابرجعبه‌های دو کلاس مختلف، توسط نرون‌های جبران‌ساز مدیریت می‌شوند. نرون‌های جبران‌ساز، با توجه به نوع همپوشانی ایجاد

1 - Adaptive Resolution Classifiers
2 - Fuzzy Min-Max Neural Network with Compensatory Neurons

شده بین ابرجعبه‌ها، به دو دسته OCN[1] و CCN[2] تقسیم می‌شود. با توجه به شکل ۲-۱۵، که سه حالت مختلف همپوشانی را نشان می‌دهد، عملکرد نرون‌های جبران‌ساز به صورت زیر می‌باشد:

۱- نرون‌های بخش OCN، همپوشانی نوع اول را مدیریت می‌کنند (overlap در شکل ۲-۱۵).

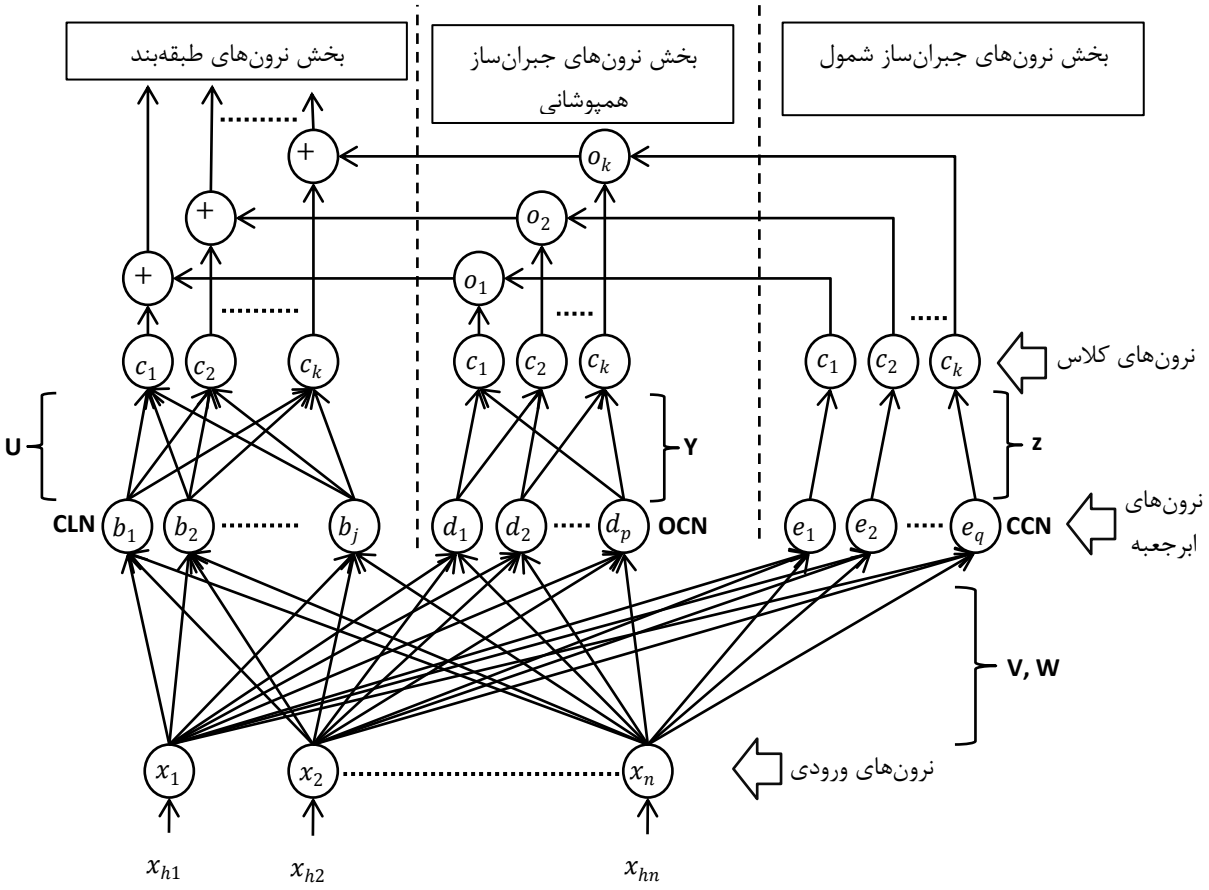
۲- نرون‌های بخش CCN، همپوشانی نوع دوم و سوم را مدیریت می‌کنند (Full Containment و Partial Containment در شکل ۲-۱۵).

شکل ۲-۱۷ ساختار شبکه FMCN و قسمت‌های مختلف آن را نشان می‌دهد. همانطور که از شکل پیداست، FMCN از سه بخش CLN[3]، OCN و CCN تشکیل شده است. بخش CLN مشابه FMNN مربوط به ابرجعبه‌ها می‌باشد. ورودی‌ها توسط لایه اول روی هر سه بخش پخش می‌شود. برای نرون‌های هر بخش از تابع عضویت متفاوتی استفاده می‌شود. خروجی‌های این سه بخش با هم ترکیب می‌شوند تا خروجی نهایی شبکه را مشخص کنند.

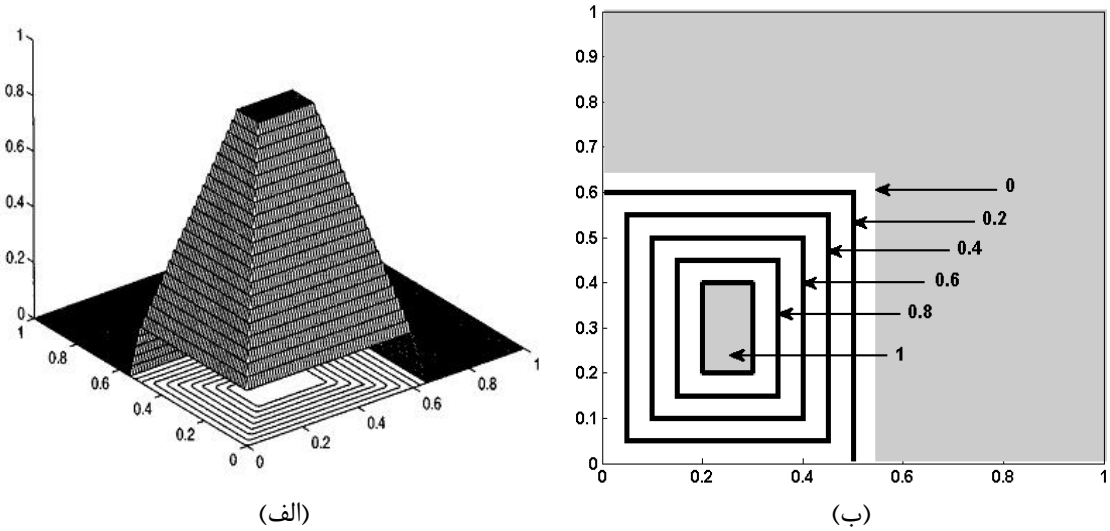
۲-۵-۵-۱- بخش CLN

این بخش مربوط به نرون‌های طبقه‌بند می‌باشد. ساختار گره‌ها و وزن‌های این قسمت مشابه FMNN می‌باشد. یک ابرجعبه در این بخش با داده ورودی، ماکزیمم، مینیمم و تابع عضویتش مشخص می‌شود (مشابه رابطه (۲-۷)). شبکه برای نرون‌های این بخش از تابع عضویت معرفی شده در شبکه GFMN استفاده می‌کند. رابطه (۲-۱۷) این تابع عضویت را توصیف می‌کند. تابع عضویت رابطه (۲-۸) با دور شدن از ابرجعبه به صورت یکنواخت کاهش نمی‌یابد و برای نقاط دور نیز مقدار عضویت بالایی بدست می‌دهد. اما تابع عضویت رابطه (۲-۱۷) این مشکل را حل کرده است. شکل ۲-۱۸ نحوه تغییر یکنواخت این تابع را با دور شدن از ابرجعبه نشان می‌دهد.

1 - Overlap Compensation Neurons
2 - Containment Compensation Neurons
3 - Classifying Neurons



شکل ۲-۱۷: ساختار FMCN [۱۶].



(الف) (ب)

شکل ۲-۱۸: تابع عضویت استفاده شده در شبکه GFMN و FMCN به ازای $\boldsymbol{V}=(\boldsymbol{0.2},\boldsymbol{0.2}),\boldsymbol{W}=(\boldsymbol{0.3},\boldsymbol{0.4}),\boldsymbol{\gamma}=\boldsymbol{4}$ الف) نمودار سه‌بعدی تابع عضویت و ب) منحنی‌های حاصل از برش مقطعی (الف) [۱۵].

$$b_j(X_h, V_j, W_j) = \min_{i=1\dots n}\left(\min(1 - f(x_{hi} - w_{ji}, \gamma), 1 - f(v_{ji} - x_{hi}, \gamma))\right)$$

$$, \qquad f(x,\gamma) = \begin{cases} 1, & \gamma x > 1 \\ \gamma x, & 1 \geq \gamma x \geq 0 \\ 0, & \gamma x < 0 \end{cases} \qquad (۱۷-۲)$$

۲-۵-۵-۲- بخش OCN

نرون‌های این بخش، مسئول تصمیم‌گیری در ناحیه همپوشانی نوع اول (شکل ۲-۱۵) دو ابرجعبه غیر هم‌کلاس می‌باشند. با ایجاد همپوشانی نوع اول، یک نرون OCN تولید می‌شود. از آنجا که برای یک داده واقع شده در این ناحیه، هر دو ابرجعبه مقدار عضویت ۱ تولید می‌کنند، نرون جبران‌ساز ایجاد شده دو خروجی تولید می‌کند که هر کدام از این دو خروجی، نهایتاً با مقدار عضویت یکی از ابرجعبه‌ها جمع می‌شود تا مقدار عضویت جدید ابرجعبه حاصل شود. بنابراین، با این تکنیک دیگر نیازی به منقبض کردن ابرجعبه‌ها برای حل این مشکل وجود ندارد. برای نرون‌های این بخش از تابع عضویت رابطه (۲-۱۸) استفاده شده است.

$$d_{jp} = U(b_j(X_h, V_j, W_j) - 1) \times (-1 + \frac{1}{n}\sum_{i=1}^{n} \max(\frac{x_{hi}}{w_{pi}}, \frac{v_{pi}}{x_{hi}})) \qquad (۱۸-۲)$$

j: شماره نرون OCN

p: شماره خروجی نرون که برابر ۱ یا ۲ می‌باشد.

$U(x)$: تابع پله واحد

b_j: مشابه رابطه (۲-۱۷)

W_j , V_j: نقاط مینیمم و ماکزیمم OCN

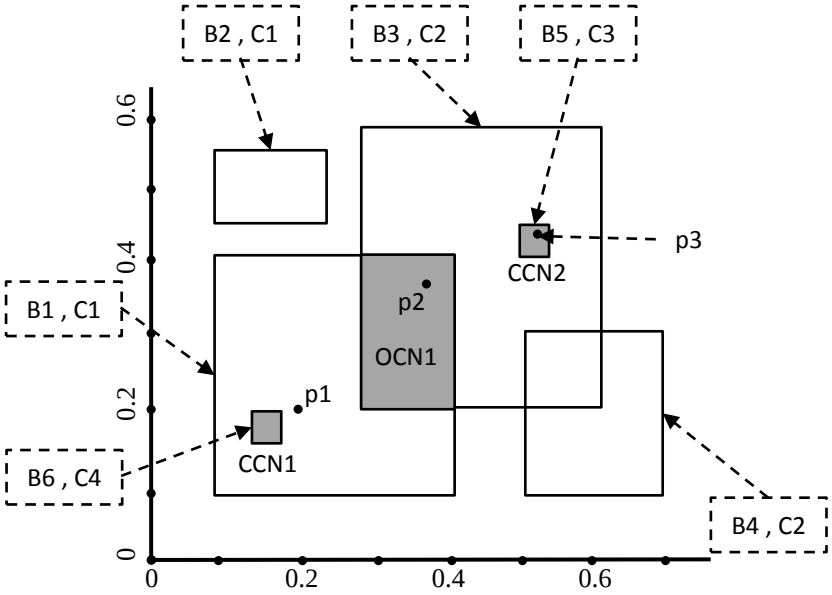
W_2, V_2, W_1, V_1: نقاط مینیمم و ماکزیمم ابرجعبه‌های همپوشان

X_h: نمونه ورودی

ترم اول رابطه (۲-۱۸) (یعنی: $U(b_j(X_h, V_j, W_j) - 1)$)، ناحیه تصمیم‌گیری نرون OCN را به داخل ناحیه همپوشانی محدود می‌کند. ترم دوم این رابطه، برحسب موقعیت داده تست در ناحیه همپوشان، دو خروجی منفی ایجاد می‌کند. این امر با استفاده از فاصله داده مورد نظر از نقاط مینیمم و ماکزیمم ابرجعبه‌ها صورت می‌گیرد. به طوری که با نزدیک‌تر شدن داده به مرزهای ابرجعبه، این

مقدار بزرگ‌تر (نزدیک‌تر به صفر) و با دور شدن از مرزهای ابرجعبه این مقدار منفی‌تر می‌شود. با این سازوکار، ناحیه همپوشانی به دو قسمت تقسیم می‌شود.

شکل ۲-۱۹ مثالی را نشان می‌دهد که در آن چند داده نمونه، در نواحی مختلف ایجاد شده توسط ابرجعبه‌ها واقع شده‌اند. جدول ۲-۳ مقادیر عضویت ایجاد شده برای هر داده، توسط این ابرجعبه‌ها را نشان می‌دهد. همانطور که در شکل ۲-۱۹ دیده می‌شود، نقطه p2 در ناحیه همپوشانی ابرجعبه‌های B1 و B3، که به ترتیب متعلق به کلاس C1 و C2 می‌باشند، واقع شده است. بنابراین این دو ابرجعبه مقدار عضویت ۱ برای داده مورد نظر بدست می‌دهند. این امر در جدول ۲-۳ دیده می‌شود. در این جدول مقادیر تولید شده توسط نرون OCN1 نیز آورده شده است. مشاهده می‌شود که خروجی مربوط به C2 با مقدار ۰,۲۸- ، منفی‌تر از خروجی مربوط به C1 با مقدار ۰,۰۴- می‌باشد.



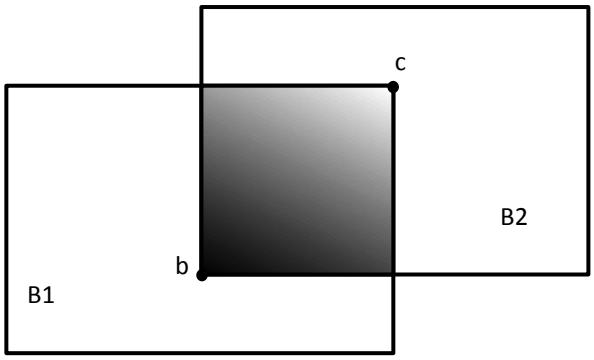
شکل ۲-۱۹: مثالی دو بعدی از نحوه عملکرد نرون‌های CCN و OCN [۱۶].

این مقادیر با مقدار عضویت ابرجعبه‌ها یعنی ۱ جمع شده و مقدار عضویت نهایی برای C1 و C2 به ترتیب برابر ۰,۹۶ و ۰,۷۲ بدست آمده است. بنابراین داده p2 به کلاس C1 منسوب شده است. (در واقع این داده باید به کلاس C2 منسوب شود، می‌توان با عوض کردن ترتیب خروجی‌های OCN1 این مشکل را حل کرد).

جدول ۲-۳: مقادیر عضویت حاصل از ابرجعبه‌ها برای مثال شکل ۲-۱۹ [۱۶].

نقطه تست	خروجی بخش طبقه‌بند						خروجی بخش جبران‌ساز				مقدار عضویت نهایی پس از افزودن جبران‌سازی			
	کلاس						کلاس							
	C1		C2		C3	C4	C1		C2		کلاس			
	B1	B2	B3	B4	B5	B6	OCN1	CCN1	OCN1	CCN2	C1	C2	C3	C4
P1(0.22,0.22)	1	0.08	0.68	0	0	0.92	0	0	0	0	1	0.68	0	0.92
P2(0.38,0.39)	1	0.48	1	0.52	0.52	0.24	-0.04	0	-0.28	0	0.96	0.72	0.52	0.24
P3(0.55,0.45)	0.4	0	1	0.4	1	0	0	0	0	-1	0.4	0	1	0

شکل ۲-۲۰ نحوه تغییر مقدار عضویت ابرجعبه‌ها توسط رابطه (۲-۲۰)، در ناحیه همپوشان را نشان می‌دهد. در این شکل عضویت مربوط به ابرجعبه B1 در ناحیه همپوشان نشان داده شده است، این مقدار از نقطه b تا c کاهش می‌یابد. برای ابرجعبه B2 این مقدار از c تا b کاهش می‌یابد.



شکل ۲-۲۰: نحوه تغییر مقدار عضویت مربوط به ابرجعبه B1 در ناحیه همپوشانی توسط نرون OCN، مقدار عضویت نقطه c به B1 برابر ۱ است و با تیره‌تر شدن شکل مقدار عضویت کاهش می‌یابد.

در شکل ۲-۱۷ نرون‌های بخش OCN با d نشان داده شده‌اند. شکل ۲-۲۱-الف نیز شماتیک یک نرون OCN را نشان می‌دهد.

۲-۵-۵-۳- بخش CCN

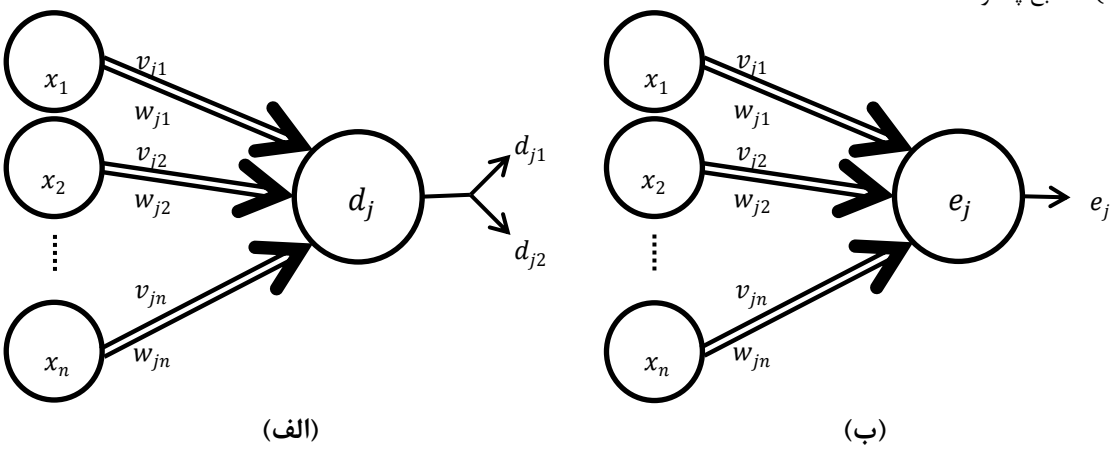
این بخش برای تصمیم گیری در نواحی همپوشان از نوع شمول[1] (شامل شمول کامل[2] و شمول جزئی[3] یعنی همپوشانی نوع دوم و سوم در شکل ۲-۱۵) تعبیه شده است. نرون‌های این قسمت یک

1 - Containment
2 - Full Containment
3 - Partial Containment

خروجی تولید می‌کنند که نهایتاً با خروجی CLN حاصل از ابرجعبه شامل شونده، تجمیع می‌شود. مثلا اگر ابرجعبه j داخل ابرجعبه k باشد و داده تست داخل ابرجعبه j بیافتد، این دو ابرجعبه هردو فعال می‌شوند بنابراین نرون CCN ایجاد شده خروجی ابرجعبه ***k***ام را خنثی می‌کند تا ابرجعبه ***j***ام برنده شود. تابع عضویت استفاده شده برای نرون‌های این بخش به صورت رابطه (۲-۱۹) می‌باشد. در شکل ۲-۱۷، نرون‌های این بخش با ***e*** نشان داده شده‌اند. شماتیک یک CCN در شکل ۲-۲۱-ب نشان داده شده است.

$$e_j = -1 \times U(b_j(X_h, V_j, W_j) - 1) \qquad (۱۹-۲)$$

e_j: خروجی نرون CCN
W_j , V_j: نقاط مینیمم و ماکزیمم CCN
b_j: مشابه رابطه (۲-۱۷)
X_h: نمونه ورودی
$U(x)$: تابع پله واحد



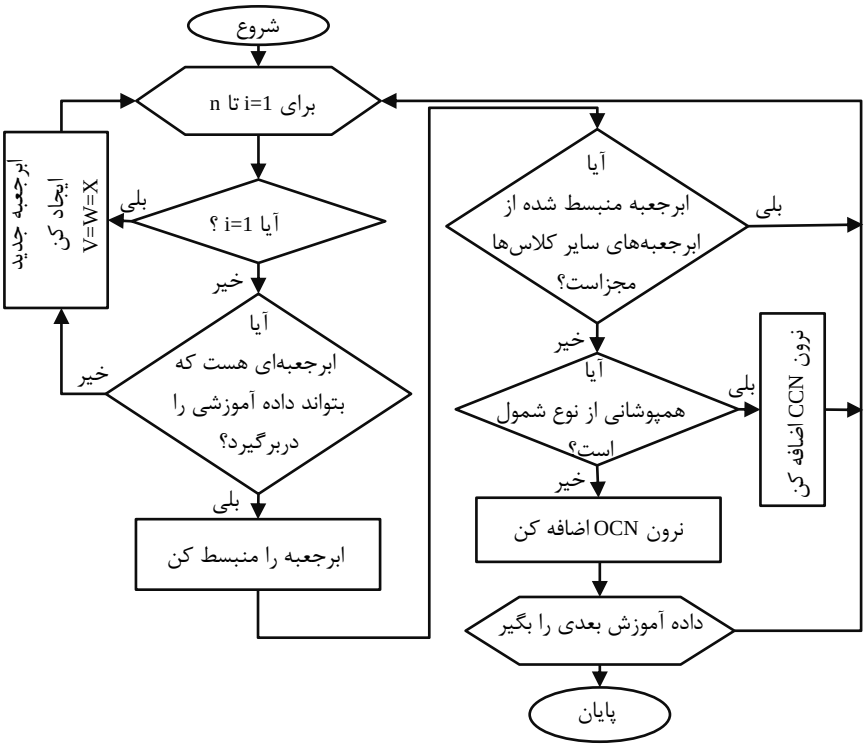
شکل ۲-۲۱: الف) نرون OCN در شبکه FMCN، ب) نرون CCN در شبکه FMCN [۱۶].

دوباره به مثال شکل ۲-۱۹ و جدول ۲-۳ برمی‌گردیم. داده p3 در داخل ابرجعبه B5 متعلق به کلاس C3 و ابرجعبه B3 متعلق به کلاس C2 واقع شده است و بنابراین همانطورکه در جدول ۲-۳ دیده می‌شود، هر دو این ابرجعبه‌ها مقدار عضویت ۱ بدست داده‌اند. این در حالی است که ابرجعبه B5 در داخل ابرجعبه B3 افتاده است. بنابراین نرون CCN2 برای مدیریت این ناحیه ایجاد شده است و برای

کلاس ابرجعبه B3 یعنی C2، مقدار عضویت ۱- تولید کرده است که این امر باعث برنده شدن کلاس ابرجعبه B5 یعنی C3 شده است.

۲-۵-۵-۴- آموزش FMCN

فلوچارت الگوریتم آموزش شبکه FMCN، در شکل ۲-۲۲ نشان داده شده است. این فلوچارت روش آموزش شبکه را به صورت مرحله به مرحله بیان کرده است. مشابه شبکه FMNN، با ارائه اولین داده آموزشی، یک ابرجعبه ایجاد می‌شود. با ارائه داده‌های آموزشی بعدی، الگوریتم آموزش دنبال ابرجعبه‌ای که بتواند داده جدید را در خود جای دهد، می‌گردد و اگر چنین ابرجعبه‌ای پیدا نشود، یک ابرجعبه جدید ایجاد می‌شود.



شکل ۲-۲۲: فلوچارت الگوریتم آموزش شبکه FMCN [۱۶].

ابرجعبه مورد نظر باید سه شرط زیر را برآورده سازد تا بتواند انتخاب شود:

۱) دارای بیشترین درجه عضویت باشد.
۲) اندازه آن از حداکثر مجاز تجاوز نکند.
۳) مربوط به هیچ نرون جبران‌سازی نباشد.

ابرجعبه یافت شده ممکن است برای جای دادن داده جدید، منبسط شود. در اینصورت بررسی می‌شود که آیا ابرجعبه منبسط شده با ابرجعبه‌های سایر کلاس‌ها همپوشانی دارد یا نه. در صورت وجود همپوشانی، نوع همپوشانی مشخص می‌شود و نرون جبران‌ساز مربوطه ایجاد می‌شود.

۲-۵-۵-۵- فاز تست

در این فاز با ارائه هر داده ورودی به FMCN، مقدار عضویت برای این ورودی در کلیه نرون‌های هر سه بخش شبکه محاسبه می‌شود. سپس این مقادیر طبق رابطه (۲-۲۰) با هم تجمیع می‌شوند تا خروجی نهایی شبکه را تولید کنند. FMCN نیز مانند FMNN، برای تعیین کلاس خروجی از استراتژی نرون برنده استفاده می‌کند.

$$\mu_i = \max_{j=1\dots k}(b_i u_{ji}) + \min(\min_{j=1\dots k}(d_j y_{ji}) + \min_{j=1\dots k}(e_j z_{ji})) \quad ,$$

$$u_{ij} = \begin{cases} 1, if\ \{b_j \in c_i\} \\ o, if\ \{b_j \notin c_i\} \end{cases} \quad ,$$

$$y_{ip}\ and\ y_{jp} = \begin{cases} 1, & if\ \{d_p \in c_i \cap c_j\ , i \neq j\} \\ o, & otherwise \end{cases} \quad ,$$

$$z_{iq} = \begin{cases} 1, & if\ c_j\ is\ contained\ fully\ or\ partially\ by\ c_i\ , i \neq j \\ o, & otherwise \end{cases} \quad (۲-۲۰)$$

در این رابطه، μ_i خروجی iام شبکه FMCN، u_{ij} وزن ارتباطی بین نرون ابرجعبه jام و نرون کلاس iام بخش CLN، y_{ip} وزن ارتباطی بین نرون ابرجعبه pام و نرون کلاس iام بخش OCN و z_{iq} وزن ارتباطی بین نرون ابرجعبه qام و نرون کلاس iام بخش CCN می‌باشد.

جدول ۲-۴ شبکه‌های Min-Max را با سایر طبقه‌بندهای مرسوم مقایسه کرده است. این جدول حاکی از کارایی خوب این نوع طبقه‌بندها می‌باشد.

جدول ۲-۴: مقایسه روش‌های عصبی فازی Min-Max با طبقه‌بندهای مرسوم روی داده گل زنبق [۱۶].

نوع طبقه‌بند	تعداد انتساب‌های غلط
Bayes classifier[۱]	۲
K nearest neighborhood[۱]	۴
Fisher ratios[۱]	۳
Ho-Kashyap[۱]	۲
Perceptron[۲]	۳
Fuzzy Perceptron[۲]	۲
FMNN[۱]	۲
GFMN[۱]	۰/۱
GFMN[۲]	۰
FMCN[۱]	۰
FMCN[۲]	۰

۱- ۷۵ داده به عنوان مجموعه آموزش (۲۵ داده از هر کلاس) و بقیه داده‌ها به عنوان مجموعه آزمایش

۲- داده آموزش و آزمایش یکسان

فصل سوم

پیاده‌سازی سخت‌افزاری

در بخش اول این فصل، به پیاده‌سازی سخت‌افزاری ANN پرداخته شده است. انواع پیاده‌سازی‌ها، مشکلات پیش رو و نحوه ارزیابی سخت‌افزار، از مباحث مطرح شده در این بخش می‌باشد. بخش دوم این فصل، به معرفی FPGA، تکنولوژی‌ها، روند طراحی، کاربردها، مزایا و معایب و... این ادوات، می‌پردازد. بخش سوم به تکنیک‌های بهینه‌سازی طراحی FPGA پرداخته است و در بخش آخر، کارهای صورت گرفته در زمینه پیاده سازی شبکه عصبی فازی Min-Max روی FPGA بررسی شده است.

۳-۱- پیاده‌سازی سخت‌افزاری شبکه‌های عصبی

به تحقق سخت‌افزاری یک ANN، شبکه عصبی سخت‌افزاری (HNN)[1] گفته می‌شود [۵]. ادوات سخت‌افزاری برای تحقق ANN و الگوریتم‌های آموزش مربوطه، مخصوصاً برای حصول مزایای ساختار موازی ذاتی این شبکه‌ها، طراحی شده‌اند. اگرچه اکثر کاربردهای تجاری ANN به شکل نرم‌افزار توسعه یافته‌اند، کاربردهای ویژه مانند فشرده‌سازی ویدئو، که نیازمند پردازش‌های حجم بالای بلادرنگ و یادگیری از پایگاه داده‌های[2] بزرگ در زمانی معقول می‌باشند، استفاده ازHNN با ساختار موازی مناسب و توان مصرفی پایین را توجیه می‌کند. سخت‌افزارهای ANN ویژه دارای مزایای محسوسی هستند که در زیر بیان شده است.

سرعت: سخت‌افزارهای خاص منظوره با قیمت محدود، قدرت محاسباتی بالایی را فراهم می‌کنند. مخصوصا در حوزه‌ی ANN با قابلیت محاسبات موازی، می‌توان به چندین برابر سرعت رسید.

هزینه: پیاده‌سازی سخت‌افزاری با کاهش مصرف توان و تعداد قطعات کلی، می‌تواند منجر به کاهش قیمت سیستم شود. این امر برای کاربردهای خاصی مانند ادوات پردازش تصویر می‌تواند مهم باشد.

1 - Hardware Neural Network
2 - Data Base

تنزل مطبوع: سیستم‌های پردازش معمولاً در برابر خطاها آسیب‌پذیرند و خطاها منجر به قطع عملیات سیستم می‌شوند. مطالعات اخیر نشان می‌دهد، حتی با معرفی پردازشگرهای چندهسته‌ای، بازهم نیاز به یک مکانیزم کارآمد در برابر خطاها وجود دارد. ولی معماری‌های موازی و توزیع‌شده در صورت بروز خطا در برخی قسمت‌هایشان، با کاهش اندک کارایی، همچنان به کار خود ادامه می‌دهند (تنزل مطبوع). و این می تواند برای کاربردهای خاصی مانند کاربردهای امنیتی مهم باشد.

۳-۱-۱- مشکلات پیش رو در طراحی HNN

نگاشتن ساختارهای پیچیده در صفحات دوبعدی منظم، که مستلزم محاسبات پیچیده و ارتباطات توزیع‌شده می‌باشد، کشمکش بزرگی برای طراحی VLSI شبکه عصبی سخت‌افزاری می‌باشد. همچنین قیود سخت‌افزاری (مخصوصاً در قطعات آنالوگ)، ممکن است خطاهای محاسباتی ایجاد کند، که منجر به تنزل یادگیری وکاهش دقت می‌شود. این خطاها با انحراف مسیر آموزش، زمان لازم برای رسیدن به همگرایی را افزایش می‌دهند. کشمکش دیگر، توابع فعال‌سازی غیرخطی می‌باشد. همچنین یک سیستم کاربردی علاوه برANN نیازمند قطعات دیگری مثل سنسورها، A/D، D/A و ... می‌باشد که به ساختار HNN اضافه می‌شوند. همچنین در یک سیستم کاربردی، سخت‌افزارهای پیش‌پردازش و پس پردازش نیز باید به ساختار مزبور اضافه شوند. برای غلبه بر این مشکلات، تکنولوژی‌های فراوانی مانند دیجیتال، آنالوگ، ترکیبی، مبتنی برFPGA و پیاده‌سازی‌های نوری ارائه شده‌اند.

۳-۱-۲-پارامترهای ارزیابی HNN

یک ANN عموماً با توپولوژی شبکه، تابع فعال‌سازی، الگوریتم آموزش، تعداد و نوع ورودی-خروجی، تعداد نرون‌ها، لایه‌ها، وزن‌های ارتباطی و ... معین می‌شود. برای سخت‌افزار هم مشخصاتی مثل تکنولوژی مورد استفاده (دیجیتال، آنالوگ، ترکیبی یا FPGA)، نوع نمایش داده (ممیزثابت[1] یا

1 - Fixed Point

ممیزشناور[1])، حافظه مصرفی، تعداد بیت‌های دقت، اتصالات قابل پیکربندی یا اتصالات ثابت[2]، آموزش روی- چیپ[3] یا چیپ در حلقه[4]، تابع‌انتقال روی- چیپ یا خارج- چیپ[5] (به عبارتی جدول جستجو[6]) و میزان قابلیت کاسکد شدن، تعیین کننده می‌باشد.

بر پایه این پارامترها، اشکال مختلفی از شایستگی برای بررسی کارایی سخت‌افزار حاصله، به کار رفته است که معمول‌ترین آن‌ها شامل پارامترهای ارزیابی زیر می‌باشند:

اتصالات بر ثانیه (CPS)[7] برای سرعت پردازش: نسبت عملیات جمع/ضرب و محاسبه‌ی تابع انتقال در فاز تست. این پارامتر تعیین می‌کند که الگوریتم ارائه شده چقدر مناسب معماری می‌باشد.

اتصال به‌روز شده بر ثانیه (CUPS)[8] برای سرعت آموزش: نسبت تغییر وزن‌ها درطول آموزش، شامل محاسبه و به‌روز کردن وزن‌ها. این پارامتر سرعت سیستم برای نگاشت‌های ورودی- خروجی را می‌سنجد.

انرژی سیناپسی[9]: انرژی متوسط مورد نیاز برای محاسبه و به‌روز کردن هر سیناپس، که با WCPS (وات بر ثانیه برای هر اتصال)[10] بیان می‌شود.

در پیاده‌سازی دیجیتال ANN، توابع فعال‌سازی غیرخطی چالش بزرگی به‌حساب می‌آیند. این توابع منجر به افزایش زمان و ناحیه‌ی چیپ می‌شوند. راه‌حل‌های ممکن استفاده از جدول جستجو و تقریب با توابع تکه‌ای-خطی می‌باشد. البته جداول جستجوی بزرگ، تعداد بیت‌های دقت را محدود می‌کند (برای کاهش حافظه مصرفی). همچنین از جملات درجه‌ی ۱و۲ بسط تیلور نیز می‌توان استفاده کرد

1 - Floating Point
2 - Hardwired
3 - On-Chip
4 - Chip-In-The-Loop
5 - Off-Chip
6 - Look-Up Table
7 - Connections-Per-Second
8 - Connection-Updates-Per-Second
9 - Synaptic Energy
10 - Watt Per Connection-Per-Second

که با تعداد ۱۶ تا ۲۰ بیت دقت خوبی بدست می‌دهد. یک مطالعه‌ی آماری نشان داده است که درMLP دولایه با آموزش پس انتشار خطا، تعداد ۱۶ بیت کافی بوده و حداقل ۱۲ بیت لازم است. همچنین در بعضی مقالات حدود بالا و پایین برای تعداد بیت‌ها در کاربردهای مختلف تعیین شده است. استفاده از این اطلاعات می‌تواند در انتخاب معماری مناسب برای طراحی کارا، مفید باشد.

در شبکه‌های بزرگ بهینه‌سازی حافظه نیز مهم می‌باشد. مصالحه‌ای بین مصرف توان و چگالی حافظه وجود دارد. مثلا ۶ ترانزیستورSRAM توان کمتری از یک ترانزیستورDRAM مصرف می‌کنند اما یک ترانزیستورDRAM فضای تراشه‌ی کمتری نسبت به ۶ ترانزیستورSRAM اشغال می‌کند.

۳-۱-۳- طبقه‌بندی انواع HNN

طبقه‌بندی HNN که برای اهداف کاربردی مفید باشد، بسیار پیچیده است. اولین منبع پیچیدگی مربوط به مشخصه‌های فراوان مرتبط با معماری سخت‌افزار و مدل ANN می‌باشد. در طبقه‌بندی HNN از این مشخصه‌ها استفاده می‌شود، بنابراین در این طبقه‌بندی علاوه‌بر معماری، ساختار شبکه نیز مطرح می‌باشد.

۳-۱-۴- رویکردهای سخت‌افزاری طراحی نرون

پیاده‌سازی آنالوگ، معمولا کارایی بهتری را در سرعت پردازش و ناحیه‌ی اشغالی چیپ بدست می‌دهد، ولی از دقت کمتری برخوردار است. در عوض در پیاده‌سازی‌های دیجیتال دقت به قیمت کاهش کارایی حاصل می‌شود (ناحیه‌ی بزرگتر اشغال‌شده‌ی چیپ، قیمت بالا، مصرف توان بیشتر). بنابراین مصالحه‌ای بین دقت و پیاده‌سازی کارا وجود دارد.

۳-۱-۴-۱- نرون دیجیتال

در یک نرون دیجیتال، وزن‌های سیناپسی در شیفت رجیسترها، لچ‌ها یا حافظه‌ها ذخیره می‌شود. جمع‌کننده‌ها، تفریق‌کننده‌ها و ضرب‌کننده‌ها به‌عنوان مدارهای استاندارد لازمند. توابع غیرخطی می‌توانند توسط جداول جستجو یا جمع‌کننده‌ها و ضرب‌کننده‌ها و ... ساخته شوند. پیاده‌سازی

دیجیتال شامل مزایایی از قبیل سادگی، سیگنال به نویز بالا، انعطاف‌پذیری، قابلیت سری‌سازی ساده و ساخت ارزان در مقابل معایبی چون عملکرد کند (بویژه در محاسبه‌ی وزن × ورودی) می‌باشند.

۳-۱-۴-۲- نرون آنالوگ

در نرون آنالوگ وزن‌ها عموما توسط مقاومت، CCD[1]، خازن‌ها یا EEPROM گیت شناور ذخیره می‌شوند. در حوزه‌ی آنالوگ توابع غیرخطی غالبا مستقیما حاصل می‌شود (مثل مشخصه‌های جریان و ولتاژ اشباع ترانزیستور). توابعی که در پیاده‌سازی نرم‌افزاری به‌کار می‌روند، به‌سادگی درVLSI قابل پیاده‌سازی نیستند. بنابراین برخی تقریب‌ها روی این توابع صورت می‌گیرد. همچنین مزیت پیاده‌سازی آنالوگ نرون، بهره گرفتن از اثرات فیزیکی ساده برای پیاده کردن برخی از توابع شبکه می‌باشد. برای مثال عمل جمع توسط قانون جریان کیرشهف قابل پیاده‌سازی است.

عناصر آنالوگ غالباً کوچک‌تر و ساده‌تر از عناصر دیجیتالی متناظر می‌باشند، از طرفی بدست آوردن یک مدار آنالوگ دقیق، نیازمند طراحی و ساخت پیچیده می‌باشد.

۳-۱-۵- تراشه‌های HNN

در این بخش مروری بر HNN هایی که به صورت تراشه پیاده‌سازی شده‌اند و تحت عنوان تراشه‌عصبی[2] شناخته می‌شوند، صورت گرفته است. این تراشه‌ها شامل تراشه‌های دیجیتال، آنالوگ، ترکیبی، تراشه‌های عصبی بر پایه FPGA، تراشه‌های عصبی بر پایه RAM و تراشه‌های عصبی برای حافظه‌های انجمنی می‌باشند. در ادامه، برخی از این تراشه‌ها به طور خلاصه معرفی شده‌اند. تراشه عصبی می‌تواند همه منظوره[3] یا خاص منظوره[4] باشد. تراشه عصبی همه منظوره بیش از یک الگوریتم عصبی را برای کاربرد خاص پیاده‌سازی می‌کند. ولی تراشه عصبی خاص منظوره، یک الگوریتم عصبی خاص را برای چندین کاربرد پیاده‌سازی می‌کند [۵]. در این تراشه‌ها، عموماً یک بلوک فعال‌سازی،

1 - Charge Coupled Devices
2 - Neurochip
3 - General-Purpose Neurochip
4 - Special-Purpose Neurochip

عمل ضرب ورودی در وزن‌ها و جمع آن‌ها را انجام می‌دهد. بقیه بلوک‌ها شامل حالت نرون، وزن‌ها و توابع فعال‌سازی می‌باشد. که می‌توانند روی تراشه یا خارج آن واقع شوند، یا برخی از این توابع ممکن است توسط یک کامپیوتر میزبان[1] ایجاد شوند. حالت نرون و وزن‌ها می‌توانند به شکل دیجیتال یا آنالوگ ذخیره شوند و وزن‌ها می‌توانند به صورت استاتیک بارگذاری شوند یا به صورت پویا به روز شوند.

۳-۱-۵-۱- تراشه عصبی دیجیتال

این تراشه‌ها عموماً با تکنولوژی CMOS ساخته می‌شوند. مزایای این نوع تراشه‌ها ذخیره سازی در RAM، طراحی انعطاف پذیر و درک کامل فرایند ساخت آنها می‌باشد. کشمکش اصلی برای طراحان در این نوع تراشه‌ها، ضرب کننده وزن‌ها می‌باشد که کندترین المان در ساختار سیستم می‌باشد.

۳-۱-۵-۲- تراشه عصبی آنالوگ

یک مثال خوب از این نوع تراشه‌ها، تراشه ETANN[2] شرکت Intel می‌باشد. ETANN 80170NX یک تراشه حرفه‌ای با ۶۴ نرون با اتصال کامل[3] است. این تراشه یک تراشه عصبی همه‌منظوره می‌باشد. وزن‌های آنالوگ به صورت بار الکتریکی در گیت‌های شناور ذخیره می‌شوند. ETANN آموزش روی تراشه را پشتیبانی نمی‌کند و فقط حالت تراشه-در-حلقه با یک کامپیوتر میزبان، پشتیبانی می‌شود و در پایان آموزش، وزن‌ها روی تراشه دانلود می‌شوند. تراشه دارای سرعت 2 GCPS[4]، دقت ۴ بیت، باس ۶۴ بیت و ۱۰۲۴۰ سیناپس قابل برنامه‌ریزی می‌باشد. این تراشه می‌تواند در ساختار کاسکد[5]، شبکه‌ای با حداکثر ۱۰۲۴ نرون و حداکثر ۸۱۹۲۰ وزن را ایجاد کند.

---
1 - Host Computer
2 - Electrically Trainable Analog Neural Network
3 - Fully Connected
4 - Giga CPS
5 - Cascade

۳-۱-۵-۳- تراشه عصبی ترکیبی

تراشه‌های ترکیبی دو تکنیک آنالوگ و دیجیتال را، برای استفاده از مزایای آنها، ترکیب می‌کنند. برای مثال می‌توان برای افزایش سرعت، محاسبات را به صورت آنالوگ انجام داد و در عین حال، وزن‌ها را به صورت دیجیتال ذخیره کرد.

۳-۱-۵-۴- پیاده‌سازی‌ها بر پایه FPGA

FPGA های با قابلیت پیکربندی مجدد، منابع قابل برنامه‌ریزی برای پیاده‌سازی و آزمودن طرح‌های مختلف، در زمانی کوتاه را فراهم می‌کنند. آنها ارزان قیمت و در دسترس هستند، دارای قابلیت پیکربندی مجدد می‌باشند و انعطاف پذیری مشابه نرم‌افزار فراهم می‌کنند. قابلیت پیکربندی مجدد جزئی و برخط در نسل‌های اخیر FPGA ها، مزایای بیشتری را فراهم کرده است. عیب عمده این ادوات، چگالی پایین مدارهای پیاده شده توسط آنها می‌باشد که عامل محدود کننده‌ای برای پیاده‌سازی مدارهایی با هزاران نرون می‌باشد.

در جدول ۳-۱ برخی از کارهای انجام شده در زمینه پیاده‌سازی سخت‌افزاری شبکه‌های عصبی به همراه تکنولوژی مورد استفاده، آورده شده است.

جدول ۳-۱: برخی از پیاده‌سازی‌های سخت‌افزاری ANNها، گزارش شده در مقالات [5].

ANN	**HNN**			
	Digital	Analog	Hybrid	FPGA
MLP	■			■
RBF	■	■		■
SOFM	■	■		
Feed-forward Network	■			■
Associative Memory		■	■	■
Recurrent NN		■		
Stochastic NN				■

۳-۱-۶- کاربردها

کاربردهای فراوانی برای HNN، ولی نه به وسعت نرم‌افزار، وجود دارد. کاربردهایی مانند OCR[1]، تشخیص صدا، مانیتورینگ ترافیک، کنترل تطبیقی و رباتیک. کاربردهای دیگری نیز وجود دارند که برخی از آن‌ها در جدول ۳-۲ آورده شده است.

جدول ۳-۲: برخی از کاربردهای HNN و انواع HNN استفاده شده برای آن [۵].

کاربردها	مثال‌ها (نوع HNN)
فیزیک انرژی بالا	دیجیتال
شناسایی الگو	دیجیتال، FPGA
شناسایی عکس/شئ	نوری، بر پایه RAM
قطعه‌بندی تصویر[2]	دیجیتال، FPGA
پردازش تصویر/ویدئو عمومی[3]	آنالوگ، نوری، بر پایه RAM، FPGA
تحلیل هوشمند ویدئو	ترکیبی، FPGA
استخراج ویژگی اثر انگشت	آنالوگ
رباتیک خودمختار[4]	دیجیتال، ترکیبی، DSP، FPGA
OCR	دیجیتال

۳-۲- مقدمه کوتاهی در مورد FPGA

آرایه گیت برنامه‌پذیر در محل (FPGA)، یک مدار مجتمع همه منظوره می‌باشد، که به جای برنامه‌ریزی توسط سازنده، توسط خود طراح برنامه‌ریزی می‌شود. بر خلاف مدارهای مجتمع ASIC، که برای پیاده‌سازی یک تابع خاص به کار می‌روند، یک FPGA، دارای قابلیت برنامه‌ریزی مجدد حتی پس از به کار رفتن در یک سیستم می‌باشد. یک FPGA با دانلود یک برنامه پیکربندی به نام رشته‌بیت[5] به SRAM موجود در روی قطعه برنامه‌ریزی می‌شود. این رشته بیت خروجی ابزارهای کامپایل می‌باشد. این ابزارها سطوح تجرید بالا، که توسط طراح ایجاد می‌شود، را به یک چیز معادل

1 - Optical Character Recognition
2 - Image Segmentation
3 - Generic
4 - Autonomous Robotics
5 -Bit Stream

که قابل اجرا و در سطح تجرید پایین‌تری می‌باشد، تبدیل می‌کنند. نرم‌افزار Xilinx System Generator، پیشگام ایده کامپایل کردن برنامه FPGA، از یک مدل Simulink سطح بالا می‌باشد.

System Generator، ابزار طراحی DSP از شرکت Xilinx می‌باشد که استفاده از محیط طراحی Simulink شرکت MathWorks را، برای طراحی FPGA ممکن می‌سازد. تمام مراحل پیاده سازی مانند سنتز، جانمایی و مسیر یابی و تولید رشته بیت، در این نرم‌افزار قابل اجرا می‌باشد. بالغ بر ۹۰ بلوک DSP، اعم از بلوک‌های عمومی مانند جمع کننده‌ها، ضرب کننده، شیفت رجیستر و بلوک‌های پیچیده‌تر مانند FFT، فیلترها و حافظه‌ها، در مجموعه Xilinx DSP blockset فراهم شده است که به Simulink، اضافه می‌شود.

یک FPGA، آرایه‌های دو بعدی قابل پیکربندی فراهم می‌کند که می‌توانند گستره وسیعی از توابع منطقی و محاسباتی را پیاده‌سازی کنند. این منابع شامل بلوک‌های DSP تعبیه شده، ضرب کننده‌ها، حافظه‌های دو درگاهه، جداول جستجو، رجیسترها، بافرهای سه حالته، مولتی پلکسرها و مدیرهای کلاک دیجیتال می‌باشند. همچنین FPGA های Xilinx، شامل مکانیزم‌های I/O پیچیده می‌باشند که می‌توانند گستره وسیعی از پهنای باند و سطوح ولتاژ را پشتیبانی کنند.

FPGA های جدید شامل میکروکنترلرهای تعبیه شده و فرستنده/گیرنده‌های سریال چند گیگا بیتی نیز می‌باشند. مثلاً Virtex®4 شرکت Xilinx، شامل میکروکنترلرهای IBM PowerPc®405 می‌باشند. منابع I/O و محاسباتی توسط شبکه‌ای از اتصالات قابل برنامه‌ریزی پوشیده شده‌اند و رشته بیت در واقع با فعال کردن این اتصالات باعث وصل شدن این منابع به همدیگر می‌شود.

FPGA ها، ادوات پردازش اطلاعات با کارایی بالا هستند. کارایی این ادوات در کاربردهای DSP، از توانایی آنها برای ایجاد ساختارهای محاسباتی با موازی کاری بالا حاصل می‌شود. در مقابل، کارایی میکروپروسسورها یا پردازنده‌های سیگنال دیجیتال (DSP)، به خاطر فرکانس بالای کلاک می‌باشد. در FPGA، ترکیبی از سرعت کلاک روزافزون (هم اکنون کلاک 100-200Mhz متداول است) و حافظه‌های توزیع شده، به طراحان اجازه می‌دهد تا به نحو مؤثری از موازی کاری در طرح‌های خود

استفاده کنند. برای مثال، پهنای باند یک FPGA بزرگ که با کلاک 150Mhz کار می‌کند، می‌تواند تا چند صد ترابایت بر ثانیه برسد. کاربردهای فراوانی وجود دارند که می‌توانند تنها با استفاده از IC های خاص یا FPGA پیاده‌سازی شوند و پروسسور Von-Newman فاقد قابلیت محاسباتی و حافظه با پهنای باند بالا، برای انجام آن می‌باشد [۶].

در موقع کار کردن با System Generator باید مد نظر داشت که در یک FPGA درجات آزادی زیادی برای پیاده‌سازی توابع پردازش سیگنال وجود دارد. برای مثال امکان انتخاب تعداد بیت‌های داده در System Generator وجود دارد یا می‌توان از پردازنده‌های داده مجزا مانند موتورهای ضرب-انباره[1] استفاده کرد. System Generator سطح تجرید بالایی را فراهم می‌کند که به طراح اجازه می‌دهد فقط با فکر کردن به الگوریتم، آن را پیاده‌سازی کند. البته هرچقدر در مورد FPGA اطلاعات بیشتری بدانیم، بیشتر می‌توانیم از قابلیت‌های ویژه‌ای که یک FPGA خاص فراهم می‌کند برای طراحی کارآمدتر استفا ده کنیم.

۳-۲-۱-کاربرد FPGA

در زیر برخی از زمینه‌های مرتبط با کاربرد FPGA بیان شده است:

مدارهای ASIC: یک FPGA وسیله پیاده‌سازی مدارهای منطقی دیجیتال است. این تراشه‌ها برای پیاده‌سازی مدارهای ASIC مناسبند. امروزه مدارهای آنالوگ FPGA هم با روند رو به رشدی در بازار قابل دسترسی هستند. FPGA ها برای تهیه نسخه اولیه یک مدار منطقی، ایده‌آل هستند. هزینه‌ی پایین پیاده‌سازی و زمان کوتاه تحقق فیزیکی یک طرح مفروض، فواید عمده‌ای را نسبت به روش‌های متداول ساخت نسخه‌ی اولیه فراهم نموده است. علاوه براین تغییرات بعدی بر روی طرح اولیه بسیار کم هزینه است.

1 - Multiply-Accumulate Engine

ماشین‌های محاسباتی مبتنی بر FPGA: امکان برنامه‌ریزی مجدد FPGA بصورت درون‌سیستمی رده جدیدی از کامپیوترها را معرفی نموده است. این ماشین‌ها دربردارنده‌ی بردی هستند که حاوی چنین FPGA هایی است. ایده‌ی نهفته در این طرح آن است که یک برنامه نرم‌افزاری به مجموعه‌ای از قطعات سخت‌افزاری کامپایل شود. این سخت‌افزار با برنامه‌ریزی مناسب FPGA تحقق می‌یابد. یا بعبارت دیگر هر الگوریتم نرم‌افزاری قابلیت پیاده‌سازی مستقیم توسط یک سخت‌افزار اختصاصی را دارد که این سخت‌افزار می‌تواند توسط یک FPGA تحقق یابد.

سخت افزارهای با قابلیت تغییر وظیفه در محل کار: چنانچه لازم باشد آرایش عملیاتی یا عملکرد لحظه‌ای ماشینی که در حال انجام وظیفه است تغییر یابد، جذابیت بیشتر FPGAها نمایان می‌شود. یک مثال از چنین ماشینی، تجهیزات کامپیوتری است که در محل دوری قرار گرفته است و لازم است ترکیب آن در محل عملیاتی خود بمنظور تصحیح خطای عملکرد و یا خطای طراحی تغییر یابد. در چنین حالتی، بدون جداسازی برد اصلی این ماشین جهت اعمال تغییرات، می توان این عملیات را در همان مکان تعبیه‌ی ماشین انجام داد.

۳-۲-۲- مزایا و معایب FPGA

در زیر چند چند مورد از مزایا و معایب FPGA، بیان شده است.

۳-۲-۲-۱- مزایای FPGA

هزینه‌های پایین ابزار مورد استفاده: در فرایند ساخت ASIC از ماسک‌هایی استفاده می‌شود که هر کدام حدود چند هزار دلار قیمت دارد و هزینه‌ی ماسک طبیعتاً روی تعداد کل محصولات تولید شده سرشکن می‌شود. از آنجاییکه موارد محدودی از پروژه‌ها نیاز به بیش از ده‌هزار واحد از محصول تولید شده دارند، لذا هزینه‌ی تمام شده‌ی ماسک به ازای هر قطعه نسبتاً بالاست. چنین مرحله‌ای برای FPGA وجود ندارد و از این بابت در تولیدات با تعداد کم، FPGA مقرون به صرفه است.

عرضه سریع محصول: برنامه‌ریزی FPGA در حدود چند دقیقه طول می‌کشد. در حالی که فرایند ساخت ASIC به دلیل عملیاتی مانند ساخت ماسک ممکن است چند هفته طول بکشد.

ریسک کم: در مورد FPGA، تکرار طراحی ناشی از وجود خطا در یکی از مراحل، هزینه‌ی زیادی نداشته و مدت زمان زیادی طول نمی‌کشد.

بازبینی مؤثر طراحی: به جای استفاده از شبیه‌ساز، برای بازبینی عملکرد طرح، کاربر قادر است FPGA را روی برد واقعی و در شرایط واقعی قرار داده و آنرا از نظر عملیاتی بازبینی کند.

۳-۲-۲-۲- معایب FPGA

FPGA ها دارای هزینه‌ی سربار ناشی از وجود عناصر برنامه‌ریزی هستند. وجود این اجزاء علاوه بر تحمیل تأخیر انتشار سیگنال، از فضایی که به بخش فعال و عملیاتی اختصاص می‌یابد، می‌کاهد. یک مدار پیاده شده روی FPGA دارای سرعت کمتر و چگالی پایین‌تر نسبت به مدار ASIC با همان تکنولوژی می‌باشد.

۳-۲-۳- فرآیند پیاده‌سازی روی FPGA

شکل ۳-۱ مراحل مختلف پیاده‌سازی یک طرح، روی FPGA را نشان می‌دهد. اولین مرحله طراحی مدار روی FPGA، توصیف الگوریتم توسط زبان توصیف سخت‌افزار (مانند VHDL یا Verilog) یا توصیف شماتیکی الگوریتم می‌باشد. در مرحله بعد، کد HDL یا طرح شماتیکی سنتز منطقی[1] می‌شود. در این مرحله طرح وارد شده، به یک نتلیست[2] از گیت‌ها و بلوک‌های واقعی موجود در FPGA تبدیل می‌شود. پس از این، طراح می‌تواند شبیه سازی رفتاری[3] مدار طراحی شده را انجام دهد تا مطمئن شود که مدار طراحی شده، معادل با الگوریتم می‌باشد. در مرحله بعد، نگاشت تکنولوژی[4] صورت می‌گیرد. در این مرحله، نتلیست تولید شده در مرحله قبل خوانده شده و به جای

1 - Logical Synthesis
2 - Netlist
3 - Behavioral Simulation
4 - Technology Mapping

گیت‌ها و بلوک‌های موجود در آن، گیت‌ها و بلوک‌های ورودی/خروجی که در روی FPGA هدف قرار دارد، جایگزین می‌شوند.

عملیات جانمایی[1] که مرحله بعد از نگاشت تکنولوژی می‌باشد، موقعیت بهینه بلوک‌های مدار را در داخل FPGA تعیین می‌کند. هدف اصلی این مرحله آن است که بلوک‌ها را به گونه‌ای در داخل FPGA جانمایی کند که اتصالات لازم، برای ارتباط این بلوک‌ها با همدیگر، کمینه شود. یک جانمایی خوب برای طرح‌های FPGA بسیار مهم می‌باشد چون کارایی و قابلیت مسیریابی[2] را مستقیماً تحت تأثیر قرار می‌دهد. به طوری که یک جانمایی نامناسب باعث کاهش سرعت و افزایش توان مصرفی می‌شود. الگوریتم‌های جانمایی به دو دسته تقسیم می‌شوند. هدف الگوریتم‌های دسته اول کمینه کردن اتصالات لازم می‌باشد. به منظور بهینه‌سازی قابلیت مسیریابی، الگوریتم‌های دسته دوم با آنالیز زمانی مسیرها و اتصالات بحرانی را برای کمینه سازی تأخیر تعیین می‌کنند.

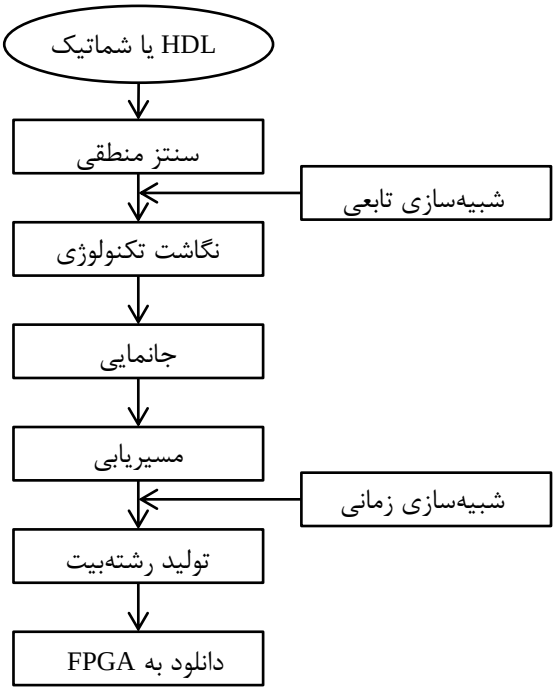
مرحله بعد از جانمایی در فرایند طراحی FPGA، مسیریابی[3] می‌باشد. عملیات مسیریابی FPGA کار مشقت باری است. دلیل آن این است که باید تنها از منابع مسیریابی از پیش ساخته شده مانند تکه‌های سیم، سوئیچ‌های قابل برنامه‌ریزی و مولتی‌پلکسرها استفاده شود.

پس از جانمایی و مسیریابی، شبیه‌سازی زمانی به منظور بررسی کارکرد مدار در حضور تأخیرهای FPGA صورت می‌گیرد. همچنین برای تعیین توان مصرفی، می‌توان آنالیز توان را، با ابزارهای تخمین توان، انجام داد.

مرحله نهایی در طراحی FPGA، تولید رشته‌بیت می‌باشد. در این مرحله رشته بیت لازم برای برنامه‌ریزی FPGA تولید می‌شود. این رشته بیت اتصالات لازم برای ایجاد طرح مورد نظر روی FPGA هدف را ایجاد می‌کند.

---

1 - Placement
2 - Routability
3 - Routing



شکل ۳-۱: فرایند پیاده‌سازی روی FPGA [۲۱]

۳-۳- تکنیک‌های بهینه‌سازی طراحی FPGA

برای یک کاربرد خاص، ممکن است نیاز باشد که بهینه‌سازی طرح برای افزایش کارایی صورت گیرد. برای مثال در کاربردهایی که نیاز به پردازش سریع دارند، کارایی سرعتی بسیار مهم‌تر از ناحیه و توان مصرفی می‌باشد. یا برای کاربردهای سیار که با باتری تغذیه می‌شوند، مصرف توان کم از اهمیت زیادی نسبت به سایر مشخصه‌ها برخوردار است. در زیر تکنیک‌های مهم بهینه‌سازی برای بهینه کردن سرعت، ناحیه و یا توان مصرفی آورده شده است.

۳-۳-۱- تکنیک‌های بهینه‌سازی سرعت

سرعت یک طرح FPGA به فاکتورهایی مثل حداکثر فرکانس کلاک، بازده[1] و تأخیر[2] وابسته می‌باشد [۲۱]. در یک طرح FPGA، حداکثر فرکانس کلاک با بیشترین تأخیر بین دو المان ترتیبی تعیین می‌شود. بازده مبین تعداد عملیات در هر سیکل کلاک می‌باشد و بر حسب بیت بر ثانیه سنجیده

1 - Throughput
2 - Latency

می‌شود. تأخیر نیز توسط فاصله زمانی داده ورودی و داده خروجی پردازش شده تعیین می‌شود و بر حسب تعداد کلاک سنجیده می‌شود. تکنیک‌های اصلی بهینه‌سازی سرعت در زیر معرفی شده است.

۱)تکنیک خط‌لوله: خط‌لوله یک تکنیک مهم برای افزایش حداکثر فرکانس کاری و بازده می‌باشد. این کار با وارد کردن رجیسترهایی در قسمت‌های بحرانی مدار حاصل می‌شود. اگر بخواهیم طراحی کم تأخیر داشته باشیم، این رجیسترها باید حذف شوند که منجر به کاهش حداکثر فرکانس کاری و بازده می‌شود.

۲)تکنیک پردازش موازی: در این روش، خروجی‌های مختلف به صورت موازی در یک کلاک محاسبه می‌شوند. خط‌لوله و پردازش موازی دوگان همدیگرند. به عبارتی اگر طرحی بتواند از خط‌لوله استفاده کند، می‌تواند به صورت موازی نیز پردازش شود.

۳)تکنیک توازن رجیستر: این تکنیک با توزیع مجدد بلوک‌های ترکیبی بین رجیسترها، نیازهای زمانی طرح را فراهم می‌کند. این کار با انتقال بلوک‌های ترکیبی از قسمت‌های بحرانی به ناحیه همسایه صورت می‌گیرد. این کار باعث می‌شود تأخیر میان دو رجیستر کمینه شده و حداکثر فرکانس کلاک افزایش یابد [۲۱].

۳-۳-۲- تکنیک‌های بهینه‌سازی ناحیه

مقایسه چند طرح روی یک FPGA خاص، با مقایسه تعداد منابع سخت‌افزاری به کار رفته می‌تواند صورت گیرد. منابع سخت‌افزاری روی FPGA، عموماً شامل موارد زیر می‌باشد:

- تعداد جداول جستجوی ۴ ورودی (LUT)
- تعداد اسلایس‌ها[1]
- تعداد فلیپ فلاپ‌ها
- تعداد بلوک‌های ورودی/خروجی (IOB)

یک طرح که از منابع تعبیه شده روی یک FPGA مدرن (مانند ضرب کننده یا بلوک‌های DSP) استفاده می‌کند، نسبت به طرحی که از این منابع بهره‌ای نمی‌برد، منابع منطقی کمتری مصرف

1 - Slice

می‌کند، منابع منطقی عمدتاً شامل LUT ها و بلوک‌های منطقی برنامه‌پذیر (CLB)[1] می‌باشند. تکنیک‌های مختلفی برای بهینه‌سازی ناحیه وجود دارد. برخی از مهم‌ترین تکنیک‌ها در زیر آورده شده است.

۱)اشتراک منابع: در این تکنیک از یک بلوک، برای پیاده‌سازی چندین عمل استفاده می‌شود. این روش همچنین با نام TDM[2] شناخته می‌شود. زمانی که منابع یک FPGA محدود است، طراح باید توجه بیشتری به این تکنیک داشته باشد. این تکنیک سطوح منطقی دیگری، برای مولتی پلکس کردن ورودی برای پیاده‌سازی چند تابع، به طرح اضافه می‌کند.

۲)استراتژی ریست مناسب: استفاده از ریست[3] و نوع ریست استفاده شده، تأثیر بسزایی روی کارایی مدار دارد. یک استراتژی ریست نامناسب، می‌تواند منجر به یک طرح بزرگ غیر ضروری شود و طرح را نیازمند بهینه‌سازی‌های ناحیه‌ای ویژه‌ای بکند. استراتژی‌های ریست غیر بهینه می‌تواند [۲۱]:

- مانع استفاده از المان‌های کتابخانه‌ای قطعه، مانند SRL[4] شود.
- مانع استفاده از المان‌های سنکرون بلوک‌های اختصاص داده شده سخت‌افزار شود.
- مانع بهینه‌سازی منطقی داخل ساختار شود.
- جانمایی و مسیریابی را محدود کند. چون سیگنال‌های ریست اغلب برون‌دهی[5] بالایی دارند.

برای FPGA های Xilinx، از ریست کردن شیفت رجیسترها پرهیز کنید. چون مانع استفاده از سلول‌های کتابخانه‌ای بهینه شده‌ی SRL می‌شود و تابع توسط منابع منطقی عمومی پیاده می‌شود، که منجر به افزایش ناحیه می‌شود. همچنین از ریست آسنکرون پرهیز کنید. چون مانع پک شدن رجیسترهای اضافه شده در داخل منابع تخصیص یافته می‌شود.

در FPGA های Xilinx، فقط المان‌های BRAM[6] ریست سنکرون دارند. بنابراین اگر از ریست سنکرون استفاده شود، ابزار سنتز قادر خواهد بود که کد را با استفاده از BRAM پیاده کند. اگر

1 - Configurable Logic Block
2 - Time-Division Multiplexing
3 - Reset
4 - Shift Register Look-Up Table
5 - Fan-Out
6 - Block RAM

RAM یکسانی توسط ریست آسنکرون پیاده کنیم، ابزار سنتز مجبور خواهد شد از بلوک‌های RAM توزیع شده کوچکتر استفاده کند و به منطق اضافه‌ای برای رمزگشایی[1] و پیاده‌سازی ریست آسنکرون نیاز داشته باشد. برای طراحی با ناحیه بهینه، توصیه می‌شود که از set و reset پرهیز شود.

۳-۳-۳- تکنیک‌های بهینه‌سازی توان

با رشد سریع کاربرد ادوات سیار، که از باتری تغذیه می‌کنند، توان مصرفی به فاکتور مهمی در طراحی تبدیل شده است. در مقایسه با مدارهای ASIC، FPGA ها به علت استفاده از ترانزیستورهای اضافی جهت تأمین قابلیت برنامه‌ریزی، توان بیشتری مصرف می‌کنند. حتی با ظهور FPGA هایی با میلیون‌ها گیت، هنوز هم استفاده از آن‌ها در مصارف توان پایین بسیار محدود می‌باشد. بنابراین باید با تکنیک‌های طراحی تا حدی بر این مشکل فائق آمد. بخش اعظم مصرف توان در FPGA های امروزی مربوط به تلفات توان پویا در اثر شارژ و دشارژ خازن‌های پارازیتی می‌باشد. تلفات توان پویا توسط میزان کلیدزنی[2] و بار خازنی خروجی LUT ها تعیین می‌شود. و توسط رابطه (۳-۱) بیان می‌شود [۲۱].

$$P_{avg} = \frac{1}{2}\sum_{i} C_i f_i V^2 \qquad (۳-۱)$$

که P_{avg} بیانگر تلفات توان متوسط، C_i ظرفیت شبکه i، V ولتاژ تغذیه و f_i فرکانس کلیدزنی متوسط شبکه i می‌باشد.

۱)قطع کلاک[3]: قطع کلاک، یکی از موفق‌ترین و پرکاربردترین تکنیک‌ها برای کاهش مصرف توان پویا می‌باشد. هدف این تکنیک، قطع کلاک قسمت‌هایی از طرح است، که محاسبات مفیدی انجام نمی‌دهند. این تکنیک، مصرف توان پویا را با کاهش کلید زنی در فلیپ فلاپ‌ها، گیت‌هایی که در خروجی فلیپ‌فلاپ‌ها واقعند و درخت‌های کلاک، کاهش می‌دهد.

1 - Decode
2 - Switching
3 - Clock Gating

۲)طراحی آسنکرون[1]: تکنیک دیگری که برای کاهش توان مصرفی به کار می‌رود، استفاده از طراحی آسنکرون می‌باشد. که در این تکنیک، زمان‌بندی به صورت محلی مدیریت می‌شود. از آنجا که این عمل ممکن است مشکلات زمان‌بندی در FPGA ایجاد کند، توصیه می‌شود که تا زمانی که ضروری نیست، از این تکنیک استفاده نشود.

۳)کاهش سرعت کلاک: در FPGA، کلاک به خاطر مدارهای منطقی مربوط به انتخاب کلاک هر فلیپ فلاپ، درصد زیادی از توان را مصرف می‌کند. در این روش بهینه‌سازی، کاهش مصرف توان با کاهش کلیدزنی در ورودی فلیپ‌فلاپ‌ها صورت می‌گیرد. با طراحی موازی و کار در فرکانس کلاک پایین، به جای طراحی سریال و کار در فرکانس کلاک بالا، می‌توان به این هدف نائل شد.

۴)رمزگذاری ماشین حالت محدود[2]: رمزگذاری ماشین حالت محدود، تکنیک دیگری برای رسیدن به طراحی FPGA کم مصرف می‌باشد. ایده کلیدی، کاهش فعالیت متوسط توسط کمینه سازی تغییر بیت‌ها، در موقع انتقال حالت‌ها می‌باشد. این کار توسط رمزگذاری باینری مانند کد گری[3]، که در آن حالت همسایه یک تغییر بیت دارد، صورت می‌گیرد.

۵)ارزیابی محافظت شده[4]: در روش ارزیابی محافظت شده، از انجام محاسبات در مدول‌هایی که نتایج آنها مورد استفاده قرار نمی‌گیرد، جلوگیری می‌شود. برای این منظور، لچ‌های شفاف[5] با ورودی فعال‌ساز در ورودی بخش‌هایی از مدار، که باید به صورت انتخابی خاموش شوند، قرار می‌گیرد. اگر مدولی قرار است در یک سیکل کلاک فعال باشد، سیگنال فعال‌ساز لچ را شفاف کرده و کارکرد عادی مدار را ممکن می‌سازد.

1 - Asynchronous Design
2 - Finite State Machine Encoding
3 - Gray Coding
4 - Guarded Evaluation
5 - Transparent Latches

۶)برش درخت کلاک[1]: در FPGA، شبکه کلاک همانند یک درخت می‌باشد که به طور تخمینی، ۱۲ تا ۷۹ درصد توان کل را مصرف می‌کند [۲۱]. برای کاهش این مصرف، برش درخت کلاک صورت می‌گیرد. هدف اصلی در این روش، کمینه کردن اندازه شبکه کلاک به کار رفته می‌باشد. این روش نتایج خوبی در کاهش مصرف توان ایجاد می‌کند.

۷)کمینه‌سازی ناحیه: کمینه‌سازی ناحیه، یکی دیگر از روش‌های کاهش مصرف توان می‌باشد. در این روش خطوط سیگنال با ظرفیت خازنی و کلید زنی بالا، برای بهینه‌سازی مصرف توان در اتصالات طویل، با هم گروه می‌شوند.

۳-۴- مروری بر کارهای صورت گرفته

علاوه‌بر پیاده‌سازی ساختارهای عصبی، FPGA ها به طور گسترده‌ای برای تحقق سیستم‌های استنتاج فازی استفاده شده‌اند. بخش اعظم این کاربردها در حوزه کنترلی می‌باشد. این ساختارها متشکل از مدارهایی هستند که قسمت‌های تشکیل دهند FIS، مانند فازی‌ساز، پایگاه قواعد، استنتاج و غیرفازی‌ساز را پیاده‌سازی می‌کنند.

در حیطه پیاده‌سازی سخت‌افزاری شبکه‌های عصبی فازی Min-Max، کار زیادی صورت نگرفته است. در [۲۲]، یک طرح سخت‌افزاری برای پیاده‌سازی شبکه عصبی فازی MRC-FMMC[2] بر روی FPGA و همچنین پیاده‌سازی این شبکه توسط ASIC، ارائه شده است.

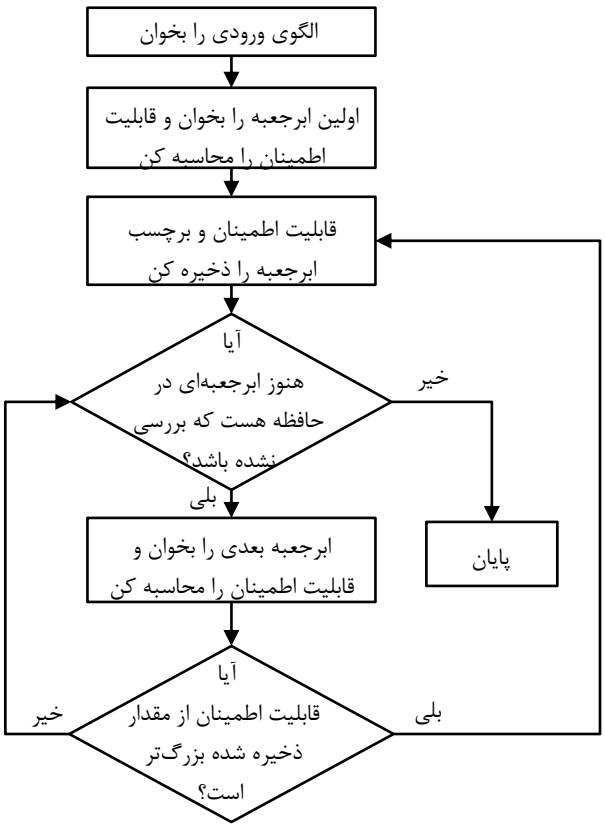
MRC-FMMC یک شبکه عصبی فازی Min-Max می‌باشد که در یک پروسه تکراری آموزش می‌بینند. برای غلبه بر مشکل وابستگی کارایی شبکه به ترتیب ارائه داده‌ها، هر ابرجعبه در MRC-FMMC دارای یک پارامتر دیگر به نام قابلیت اطمینان[3] می‌باشد که در رابطه (۳-۲) با P_j نشان داده

1 - Clock Tree Paring
2 - Multi-resolution Combined Fuzzy Min-Max Classifier
3 - Reliability

شده است و برابر با نسبت الگوهای درست طبقه‌بندی شده به کل الگوها در B_j می‌باشد. این رابطه نحوه تعریف یک ابرجعبه برای این شبکه را بیان می‌کند.

$$B_j = \{X_h, V_j, W_j, P_j\} \quad (۲-۳)$$

فلوچارت شکل ۳-۲ نحوه تعیین کلاس داده ورودی توسط سخت‌افزار طراحی شده را نشان می‌دهد. به طور خلاصه می‌توان عملکرد این فلوچارت را اینگونه توصیف کرد که با دریافت داده تست، الگوریتم تک تک ابرجعبه‌ها را از حافظه می‌خواند و مقدار قابلیت اطمینان را برای آن محاسبه می‌کند و بیشترین قابلیت اطمینان و برچسب ابرجعبه مربوطه را ذخیره می‌کند. ابرجعبه برنده، ابرجعبه‌ای است که بیشترین قابلیت اطمینان را بدست دهد.



شکل ۳-۲: فاز تست شبکه MRC-FMMC [۲۲].

رابطه (۳-۳) نحوه محاسبه قابلیت اطمینان را بیان می‌کند. در این رابطه P_j قابلیت اطمینان ابرجعبه jام، N_b تعداد الگوها قبل از انبساط ابرجعبه، V_1 و V_2 حجم ابرجعبه به ترتیب قبل و بعد از انبساط و p چگالی ابرجعبه قبل از انبساط (برابر با تعداد الگوها n_p به حجم ابرجعبه v_p) می‌باشد.

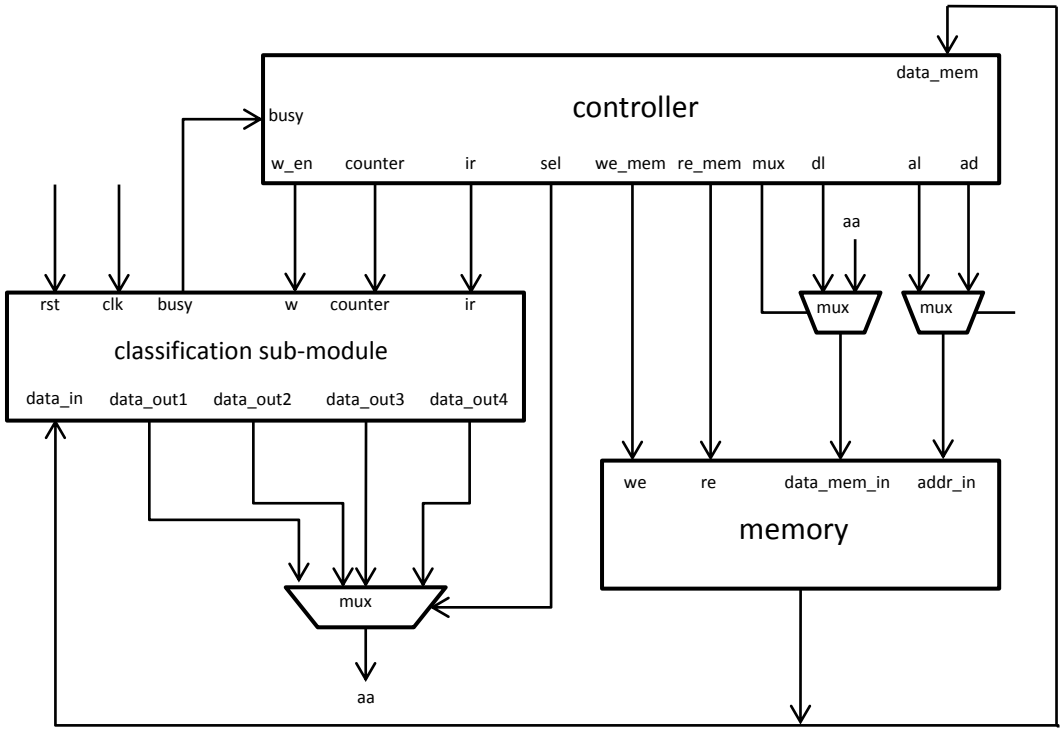
$$P_j = [N_b + 1]/\,[N_b + 1 + (V_2 - V_1) \times p \;\;,\; p = n_p/\,v_p \qquad (۳-۳)$$

جدول ۳-۳ نتایج حاصل از طبقه‌بندی سیستم سخت‌افزاری، روی داده IRIS و BUPA را نشان می‌دهد. فرکانس 50MHz روی FPGA و 100MHz روی ASIC برای طرح مورد نظر حاصل شده است.

جدول ۳-۳: نتایج حاصل از طبقه‌بندی توسط سخت‌افزار شبکه MRC-FMMC [۲۲].

مجموعه داده	تعداد تشخیص‌های صحیح داده‌های آموزش	تعداد تشخیص‌های صحیح داده‌های آزمایش
IRIS	۷۵/۷۵	۷۳/۷۵
BUPA	۲۵۴/۲۵۹	۵۶/۸۶

شکل ۳-۳ شمای کلی این سخت‌افزار را نشان می‌دهد. بلوک کنترل‌کننده مسئولیت هماهنگی بین سایر بخش‌های مدار را دارد. در حافظه، پارامترهای ابرجعبه‌ها و ورودی‌ها قرار دارند. بلوک طبقه‌بندی نیز دارای یک ساختار موازی برای محاسبه همه ابعاد در یک زمان می‌باشد.



شکل ۳-۳: ساختار کلی سخت‌افزار MRC-FMMC [۲۲].

فصل چهارم

روش جدید طبقه بندی بر پایه

ابر جعبه های فازی

MIN-Max

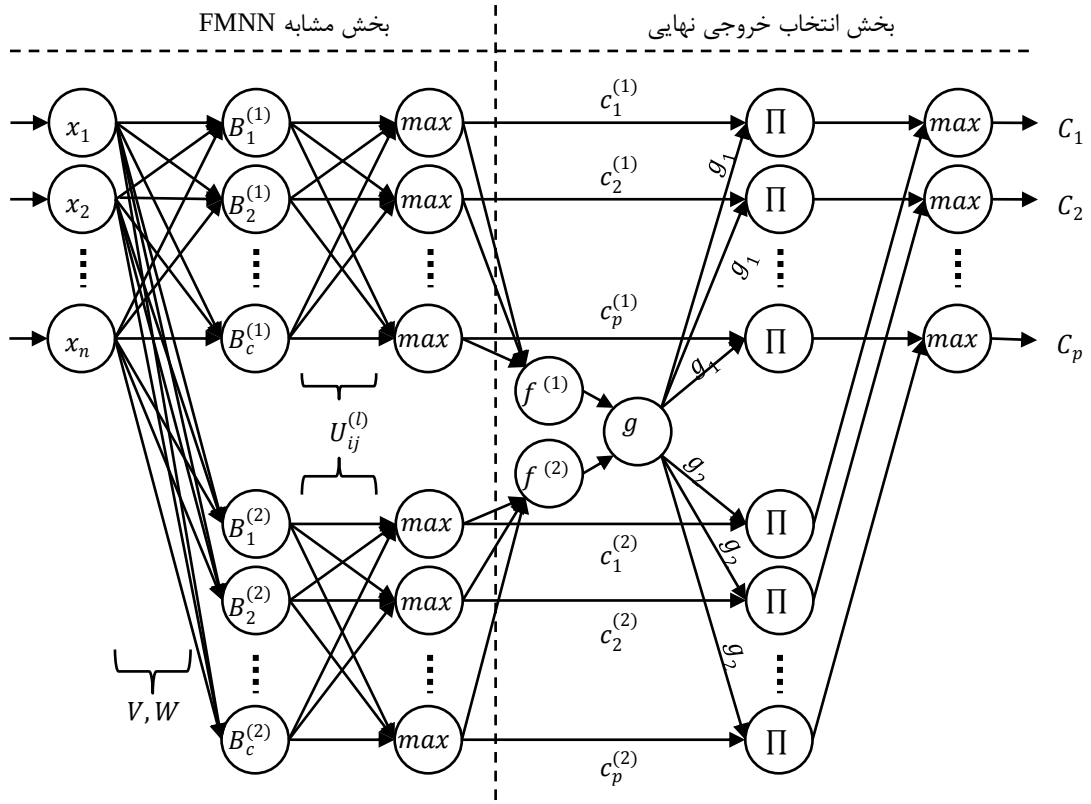
در این فصل یک دیدگاه جدید برای طراحی شبکه‌های عصبی فازی، بر پایه ابرجعبه‌های فازی Min-Max ارائه شده است. الگوریتم پیشنهادی، از آموزش با نظارت برای طبقه‌بندی داده‌های ورودی استفاده می‌کند. همانند شبکه FMNN، این شبکه از تجمیع ابرجعبه‌های فازی، برای تعیین مرز کلاس‌ها استفاده می‌کند. با این تفاوت که تست همپوشانی و انقباض ابرجعبه‌ها، از مراحل آموزش حذف شده است، که منجر به الگوریتم آموزش ساده‌تر و سرعت آموزش بیشتر شده است. شبکه ارائه شده از دو دسته ابرجعبه با ضریب انبساط (θ) متفاوت، استفاده می‌کند که باعث ایجاد یک ساختار دو طبقه برای شبکه شده است. همچنین این امر باعث شده است که شبکه، در مرز کلاس‌ها از ابرجعبه‌های کوچک‌تر، برای افزایش کارایی، استفاده کند. شبکه با یک بار ارائه داده‌های آموزش، آموزش می‌بیند. بنابراین، الگوریتم پیشنهادی، مزیت برخط بودن را حفظ کرده است. نتایج حاصل از شبیه‌سازی‌های رایانه‌ای، بیانگر کارایی خوب این شبکه نسبت به شبکه‌های مشابه، می‌باشد. روش پیشنهادی در اکثر موارد نتایج مشابهی را نسبت به بهترین روش‌های قبلی با تعداد نرون‌های کمتر بدست داده است.

روش ارائه شده یک شبکه دو طبقه می‌باشد، ولی در حالت کلی تعداد طبقات می‌تواند افزایش یابد. هر طبقه از یک شبکه ساده با مجموعه‌ای از ابرجعبه‌ها، تشکیل شده است. کلیه ابرجعبه‌های شبکه مجازند با هم همپوشانی داشته باشند. اما حداکثر اندازه مجاز ابرجعبه‌ها برای طبقات مختلف، متفاوت است و با افزایش شماره طبقه L، $\theta^{(L)}$ کاهش می‌یابد. این امر به الگوریتم اجازه می‌دهد تا در نقاط مرزی از ابرجعبه‌های کوچک‌تری استفاده کند. ضریب انبساط هر طبقه طبق رابطه (۴-۱) محاسبه می‌شود.

$$\theta^{(L+1)} = \mu^{(L)} \times \theta^{(L)} \, , \, \mu^{(L)} \in (0,1] \qquad (۴-۱)$$

۴-۱- ساختار شبکه پیشنهادی

ساختار ارائه شده برای شبکه، در شکل ۴-۱ نشان داده شده است. همانطورکه از شکل پیداست، ساختار شبکه دارای دو بخش اصلی می‌باشد. بخش اول مربوط به لایه ورودی، ابرجعبه‌ها و نرون‌های کلاس می‌باشد و از دو طبقه شبیه FMNN (شکل ۲-۱۶-الف) تشکیل شده است. هر کدام از این دوطبقه به طور مستقل خروجی‌هایی تولید می‌کنند. بخش دوم شبکه، مربوط به انتخاب خروجی نهایی از بین خروجی طبقات می‌باشد. به طوری که خروجی یک طبقه به خروجی نهایی شبکه منتقل شود.



شکل ۴-۱: ساختار شبکه پیشنهادی

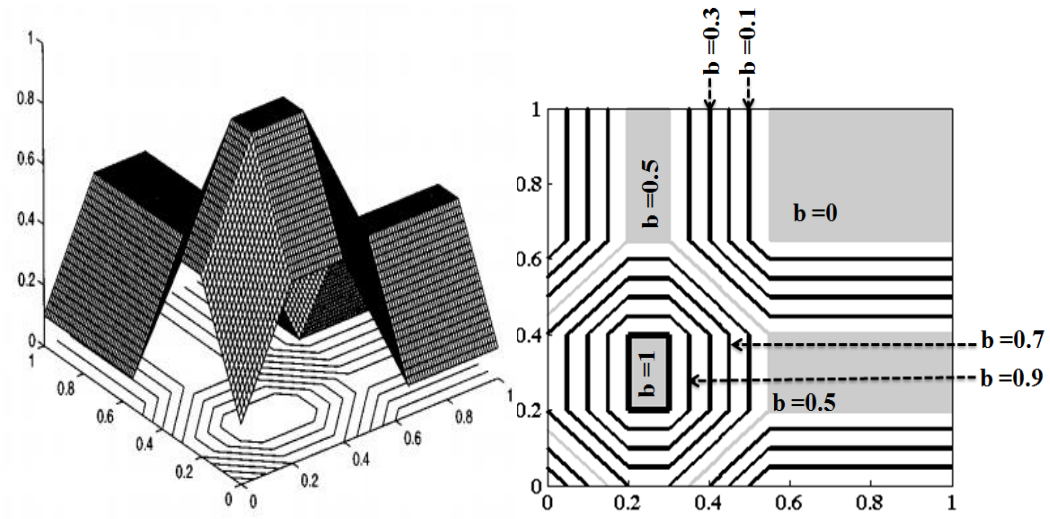
لایه اول شبکه با نرون‌های x_i لایه ورودی بوده و مسئول پخش ورودی روی نرون‌های لایه بعدی می‌باشند. تعداد نرون‌های این لایه به تعداد ابعاد داده ورودی (n) می‌باشد.

لایه دوم شبکه مربوط به نرون‌های ابرجعبه می‌باشد. نرون‌های این لایه با $B_j^{(L)}$ نشان داده شده‌اند که L شماره طبقه‌ای است که ابرجعبه متعلق به آن است. شبکه برای نرون‌های این لایه از توابع عضویت

طبیعی استفاده می‌کند به طوری که مقدار عضویت برای نقاط داخل ابرجعبه، یک باشد. برای این منظور ما از تابع رابطه (۴-۲) استفاده کرده‌ایم [۱۴].

$$b_j(X_h, V_j, W_j) = \frac{1}{n}\sum_{i=1}^{n}(1 - f(X_h - W_j, \gamma) - f(V_j - X_h, \gamma))$$

$$f(x,\gamma) = \begin{cases} 1, & \gamma x > 1 \\ \gamma x, & 1 \geq \gamma x \geq 0 \\ 0, & \gamma x < 0 \end{cases} \qquad (۴-۲)$$



(الف) (ب)

شکل ۴-۲: تابع عضویت ابرجعبه، استفاده شده در شبکه پیشنهادی به ازای $\boldsymbol{V} = (\mathbf{0.2}, \mathbf{0.2}), \boldsymbol{W} = (\mathbf{0.3}, \mathbf{0.4}), \boldsymbol{\gamma} = \mathbf{4}$، الف) نمودار سه‌بعدی تابع عضویت و ب) منحنی‌های حاصل از برش مقطعی (الف) [۱۵].

خطوط ارتباطی لایه اول و لایه دوم، در بردارنده نقاط مینیمم و ماکزیمم ابرجعبه‌ها (V, W) می‌باشند. این وزن‌ها در طول پروسه آموزش ایجاد می‌شوند.

نرون‌های لایه سوم شبکه که عمل max را انجام می‌دهند، نرون‌های کلاس می‌باشند. تعداد این نرون‌ها برای هر طبقه برابر با تعداد کلاس‌ها (p) می‌باشد. وزن‌های ارتباطی بین لایه دوم و سوم (u_{kj}) مشابه رابطه (۲-۱۶) می‌باشند. همانطور که دیده می‌شود، سه لایه اول این شکل مشابه FMNN می‌باشد.

بخش دوم شبکه خروجی نهایی را با پردازش خروجی حاصل از دو طبقه، تولید می‌کند. خروجی هر طبقه، اطلاعاتی در مورد اینکه داده تست در چه موقعیتی نسبت به ابرجعبه‌های آن طبقه قرار دارد، بدست می‌دهد. برای مثال اگر هیچ ابرجعبه‌ای از یک طبقه، مقدار عضویت کامل برای داده تست بدست ندهد، همه خروجی‌های آن طبقه مقداری کوچک‌تر از واحد خواهند داشت. اگر فقط یک خروجی طبقه برابر واحد باشد، آنگاه یک یا چند ابرجعبه با برچسب یکسان، داده را در بر گرفته‌اند و در نهایت، اگر بیش از یک خروجی یک شود، داده داخل ناحیه همپوشانی چند ابرجعبه از چند کلاس مختلف واقع شده است. ساختار شبکه به گونه‌ای طراحی شده است که با توجه به وضعیت خروجی‌های طبقات مختلف، فقط خروجی‌های یک طبقه به خروجی نهایی شبکه منتقل می‌شود.

دو نرون میانی با توابع فعالسازی $f^{(1)}$ و$f^{(2)}$، مسئول تشخیص وضعیت خروجی طبقات ۱ و ۲ می‌باشند. خروجی‌های این دو نرون به نرون g وارد می‌شوند و نرون g با توجه به وضعیت ورودی‌هایش، دو خروجی مکمل با مقادیر صفر و یک تولید می‌کند. این خروجی‌ها به لایه چهارم شبکه ($\prod$) منتقل می‌شوند تا در خروجی لایه سوم ضرب شوند. بنابراین خروجی نهایی شبکه، با توجه به وضعیت خروجی‌های نرون g برابر خروجی لایه اول یا دوم شبکه است.

عملکرد نرون‌های f را می‌توان با تابع فعالسازی رابطه (۴-۳) بیان کرد. این رابطه، در واقع تعداد خروجی‌های مساوی با ۱ را شمارش می‌کند.

$$f^{(L)}(c_1^{(L)}, \dots, c_p^{(L)}) = \sum_{i=1}^{p} U(c_i^{(L)} - 1) \qquad (۴-۳)$$

در رابطه بالا، $c_i^{(L)}$ خروجی iام طبقه Lام، p تعداد کلاس‌های خروجی و $U(x)$ تابع پله واحد است. همچنین، تابع فعال‌سازی g توسط رابطه (۴-۴) تعیین می‌شود.

$$g_1(f^{(1)}, f^{(2)}) = U(1 - f^{(1)} - f^{(1)}.f^{(2)})$$
$$g_2(f^{(1)}, f^{(2)}) = 1 - U(1 - f^{(1)} - f^{(1)}.f^{(2)}) \qquad (۴-۴)$$

رابطه بالا، بر حسب اینکه طبقات شبکه در لایه سوم، چه خروجی را تولید کنند، تصمیم می‌گیرد که زوج $(\boldsymbol{g_1}, \boldsymbol{g_2})$ را، برابر $(\mathbf{0}, \mathbf{1})$ یا $(\mathbf{1}, \mathbf{0})$ قرار بدهد. این عمل، جدول ۴-۱ را پیاده‌سازی می‌کند.

جدول ۴-۱: جدول درستی برای تصمیم گیری شبکه پیشنهادی، در نرون $\boldsymbol{g}$

$f^{(1)}$	0	0	0	1	1	1	>1	>1	>1
$f^{(2)}$	0	1	>1	0	1	>1	0	1	>1
g_1	1	1	1	1	0	0	0	0	0
g_2	0	0	0	0	1	1	1	1	1

جدول ۴-۱ را به صورت زیر می‌توان توضیح داد:

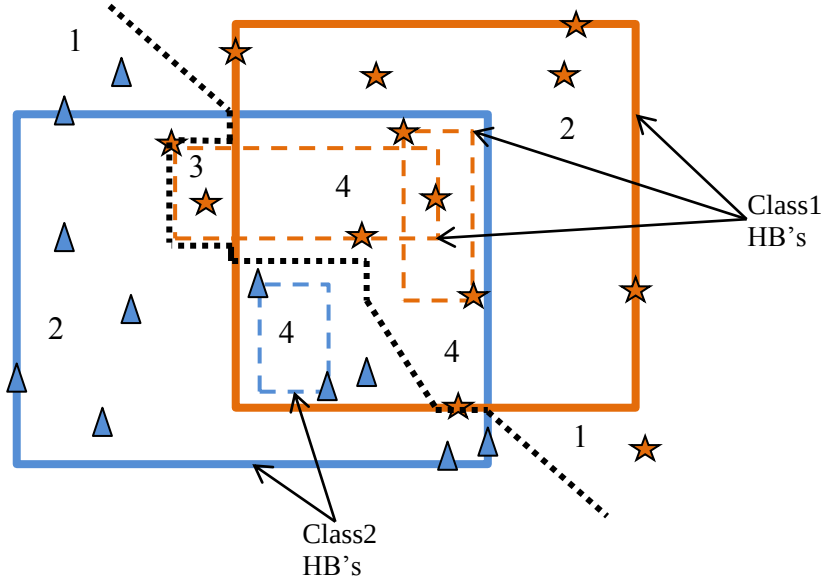
۱) اگر $f^{(1)} = 0$ باشد، در این حالت، داده تست در خارج ابرجعبه‌های طبقه اول قرار دارد. که در این حالت خروجی طبقه اول به خروجی نهایی شبکه منتقل می‌شود. دلیل ایجاد طبقه دوم شبکه این بوده است که، نرون‌های این طبقه بتوانند، نمونه‌هایی که در مرحله آموزش داخل ابرجعبه‌های اشتباه قرار گرفته‌اند را، یاد بگیرند. بنابراین، وظیفه تشخیص نمونه‌های خارج ابرجعبه‌های طبقه ۱، بر عهده طبقه ۲ نمی‌باشد.

۲) اگر $f^{(1)} = 1$ و $f^{(2)} = 0$ باشد، در این حالت، داده تست در داخل ابرجعبه‌های طبقه اول و خارج از ابرجعبه‌های طبقه دوم می‌باشد. یعنی اینکه داده مورد نظر در ناحیه داده‌هایی قرار دارد که در مرحله آموزش، داخل ابرجعبه مناسبی قرار گرفته‌اند و بنابراین توسط طبقه ۲ فراگیری نشده‌اند. که در این حالت نیز، خروجی نهایی شبکه، خروجی طبقه اول می‌باشد.

۳) اگر $f^{(1)} = 1$ و $f^{(2)} = 1$ باشد، در این حالت، داده تست در داخل ابرجعبه‌های یک کلاس از طبقه اول و همچنین، داخل ابرجعبه‌های یک کلاس از طبقه دوم می‌باشد. در این حالت، داده تست در ناحیه داده‌های آموزشی قرار دارند که داخل ابرجعبه نامناسبی افتاده‌اند و توسط نرون‌های لایه ۲ یاد گرفته شده‌اند. بنابراین خروجی نهایی شبکه در این حالت،خروجی طبقه دوم می‌باشد.

۴) اگر $f^{(1)} > 1$ باشد، در این حالت، داده تست در ناحیه همپوشان ابرجعبه‌های طبقه اول واقع می‌باشد. بنابراین در فاز آموزش، داده‌های آموزش واقع در این ناحیه توسط طبقه دوم فرا گرفته شده است. بنابراین خروجی نهایی شبکه در این حالت،خروجی طبقه دوم می‌باشد.

شکل ۴-۳، مثالی ساده از نحوه جداسازی دو کلاس داده دو بعدی، توسط شبکه پیشنهادی را نشان می‌دهد. در این شکل، ابرجعبه‌های نشان‌داده شده با خط‌چین، مربوط به طبقه دوم شبکه می‌باشند. همچنین بر روی چند داده نمونه، شماره یکی از چهار حالت وصف شده در بالا، برای تفهیم بیشتر نواحی مختلف، آورده شده است.



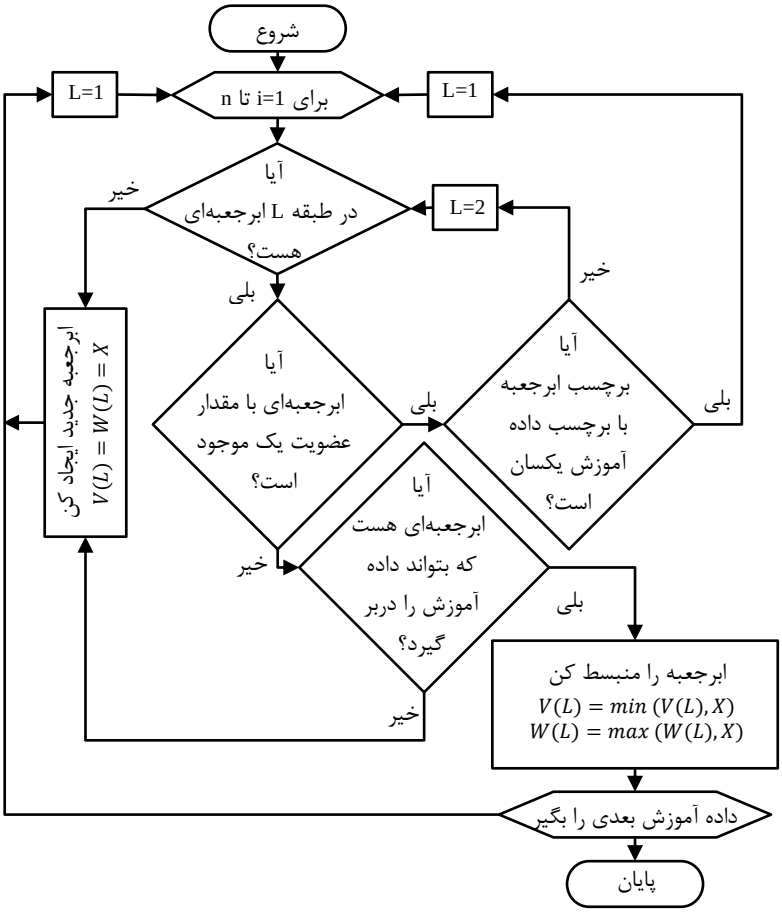
شکل ۴-۳: نحوه جداسازی دو کلاس داده دو بعدی، توسط شبکه پیشنهادی خط‌چین در حالت کلی بیانگر ابرصفحه‌های جدا کننده دو کلاس می‌باشد.

۴-۲- الگوریتم آموزش

شکل ۴-۴ فلوچارت الگوریتم آموزش روش پیشنهادی را نشان می‌دهد. شبکه برای آموزش، الگوریتم FMNN را با حذف مراحل تست همپوشانی و انقباض و اضافه کردن چند شرط برای جابجایی بین طبقات، به کار می‌گیرد.

با ارائه هر داده آموزشی، الگوریتم آموزش ابتدا لایه اول را بررسی می‌کند. اگر داده مورد نظر داخل هیچ ابرجعبه‌ای نباشد، الگوریتم دنبال ابرجعبه مناسب، که بتواند با انبساط، داده جدید را شامل شود می‌گردد. و اگر چنین ابرجعبه‌ای یافت نشد، یک ابرجعبه جدید ایجاد می‌شود. تا اینجا الگوریتم آموزش مشابه الگوریتم آموزش FMNN عمل کرده است. اگر ابرجعبه‌ای در طبقه اول باشد که برای

داده آموزشی جدید مقدار عضویت ۱ بدست دهد، بررسی می‌شود که آیا ابرجعبه متعلق به کلاس داده مورد نظر است یا نه. که در صورت مثبت بودن پاسخ، داده آموزشی بعدی دریافت می‌شود. ولی اگر داده در داخل ابرجعبه‌ای با کلاس مخالف قرار داشته باشد، الگوریتم به طبقه دوم شبکه رفته و در آنجا تمام این مراحل را دوباره انجام می‌دهد. برای هر طبقه، در اولین ورود به طبقه یک ابرجعبه جدید ایجاد می‌شود (مشابه FMNN که در ابتدای الگوریتم یک ابرجعبه ایجاد می‌کند).



شکل ۴-۴: فلوچارت الگوریتم آموزش روش پیشنهادی

۴-۳- نتایج آزمایش‌ها

برای بررسی عملکرد شبکه ارائه شده، دو نوع آزمایش صورت گرفته است. آزمایش اول با داده‌های دست‌ساز و آزمایش دوم با داده‌های واقعی پر ارجاع در مقالات علمی، صورت گرفته است. در تمام آزمایشات ۵۰ درصد داده‌ها، که به صورت تصادفی از کل داده‌ها انتخاب شده‌اند، برای آموزش شبکه و

بقیه داده‌ها برای تست شبکه استفاده شده است. انتخاب داده‌ها به گونه‌ای است، که برای هر کلاس یک داده مشخص، نیمی از داده‌های آن کلاس در آموزش و نیم دیگر در تست شبکه استفاده شده است. آزمایش‌ها برای شبکه‌های FMNN و FMCN و روش پیشنهادی تکرار شده و نتایج در چهار پارامتر درصد خطا روی داده‌های آموزش، درصد خطا روی داده‌های آزمایش، تعداد ابرجعبه‌های تولید شده و زمان آموزش شبکه، آورده شده است. تعداد ابرجعبه‌های تولید شده برای FMCN برابر مجموع ابرجعبه‌های سه بخش CLN، OCN و CCN و برای روش پیشنهادی برابر مجموع ابرجعبه‌های طبقات می‌باشد.

در [۲۳]، مؤلفان عنوان کرده‌اند که تابع عضویت رابطه (۲-۱۷) برای FMCN در همه موارد بهترین نتیجه را تولید نمی‌کند (جایی که تطابق ویژگی‌ها بسیار مهم‌تر از جستجو برای یک اختلاف کمینه است) و بنابراین از رابطه (۴-۲) برای FMCN استفاده کرده‌اند آنها رابطه (۴-۲) را به صورت رابطه (۴-۵) بیان کرده‌اند. رابطه (۴-۵) مشابه رابطه (۲-۱۷) می‌باشد با این تفاوت که در آن به جای مینیمم از میانگین استفاده شده است. رابطه (۴-۵) معادل رابطه (۴-۲) بوده و نتایج یکسانی را به ازای ورودی‌ها و پارامترهای یکسان تولید می‌کند.

$$b_j(X_h, V_j, W_j) = \frac{1}{n}\sum_{i=1}^{n} \min(1 - f(x_{hi} - w_{ji}, \gamma), 1 - f(v_{ji} - x_{hi}, \gamma)) \qquad (۴-۵)$$

با آزمایش روی داده‌های استفاده شده در پایان‌نامه، مشخص شد که FMCN، با تابع عضویت رابطه (۴-۵)، پاسخ بهتری نسبت به استفاده از رابطه (۲-۱۷) تولید می‌کند و بنابراین در آزمایش‌های انجام شده، از این رابطه برای FMCN استفاده شد.

با توجه به اینکه کارایی این نوع شبکه‌ها به ترتیب ارائه ورودی‌ها وابسته است، برای یک ترتیب مشخص از داده‌های ورودی، نتیجه هر سه شبکه محاسبه شده است و این آزمایش ۱۰۰ بار تکرار شده (در کل ۱۰۰ ترتیب مختلف از ارائه ورودی‌ها) و میانگین نتایج آورده شده است. همچنین با توجه به حساس بودن کارایی شبکه به ضریب انبساط θ، این پارامتر را در بازه $[0.05,1]$ با گام ۰.۰۵ تغییر داده و آزمایشات را تکرار کرده‌ایم. در همه آزمایش‌ها مقدار پارامتر γ برابر ۴ در نظر گرفته شده است.

همچنین برای روش پیشنهادی $\theta^{(2)} = 0.25\theta^{(1)}$ در نظر گرفته شده است. همچنین، داده‌های ورودی در بازه $[0,1]$ نرمالیزه شده‌اند. نرمالیزه‌سازی برای هر بعد داده به صورت مجزا صورت گرفته است. برای این منظور، از رابطه (۴-۶) استفاده شده است. در این رابطه، m بیانگر تعداد داده‌ها و n بیانگر تعداد ابعاد می‌باشد.

$$x_{hi}^{normalized} = \frac{x_{hi} - \min\limits_{h=1,\dots,m}(x_{hi})}{\max\limits_{h=1,\dots,m}(x_{hi}) - \min\limits_{h=1,\dots,m}(x_{hi})}\ , i = 1, \dots, n \qquad (۴-۶)$$

آزمایش اول: برای این آزمایش از داده نیم‌دایره که در شکل ۴-۵-الف نشان داده شده، استفاده شده است. این داده شامل ۴۰۰ نقطه می‌باشد که به صورت یکنواخت در $[0,1] \times [0,1]$ توزیع شده‌اند. داده‌هایی که دو شرط $(x-0.5)^2 + (x-0.5)^2 < 0.16$ و $x > y$ را داشته باشند مربوط به کلاس اول و بقیه داده‌ها مربوط به کلاس دوم می‌باشند. در شکل ۴-۵ نواحی ایجاد شده توسط شبکه‌های مختلف، نشان داده شده است. نواحی خاکستری به کلاس ۱ و نواحی سفید به کلاس ۲ منسوب شده‌اند. برای شبکه پیشنهادی، ابرجعبه‌های نشان داده شده با خط‌چین، مربوط به طبقه دوم می‌باشند. برای FMCN، فقط ابرجعبه‌های مربوط به بخش CLN نشان داده شده‌اند. در این شکل‌ها، ابرجعبه‌های قرمز مربوط به کلاس ۱ و ابرجعبه‌های آبی مربوط به کلاس ۲ می‌باشد. نتایج حاصل از این آزمایش در شکل ۴-۶ آورده شده است. همانطور که در این شکل ملاحظه می‌شود، شبکه پیشنهادی نسبت به FMCN با تعداد نرون‌های کمتر، نتایج بهتری را برای داده‌های آموزش و آزمایش، بدست داده است.

آزمایش دوم: در این آزمایش، کارایی شبکه‌ها روی داده یونوسفر[1] سنجیده شده است. این مجموعه شامل ۳۵۱ نمونه ۳۴ بعدی می‌باشد. که ۲۲۵ داده با برچسب خوب و ۱۲۶ داده با برچسب بد، دو کلاس این مجموعه را تشکیل می‌دهند. نتایج این آزمایش در شکل ۴-۷ آمده است. روش پیشنهادی با نرون‌های کم پاسخ بهتری نسبت به سایر روش‌ها، بدست داده است.

1 - Ionosphere

آزمایش سوم: در این آزمایش، کارایی شبکه‌ها روی داده شیشه[1] بررسی شده است. این مجموعه شامل ۲۱۴ نمونه ۹ بعدی می‌باشد. که ۵۱ داده کلاس اول و ۱۶۳ داده کلاس دوم، مجموعه داده را تشکیل می‌دهند. شکل ۴-۸ نتایج این آزمایش را نشان می‌دهد. در این آزمایش، در فاز آموزش، روش پیشنهادی نتایج نزدیک به FMCN، با تعداد نرون‌های کمتر نسبت به آن، حاصل کرده است. در فاز تست، روش پیشنهادی نتایج بهتری از دو روش دیگر ایجادکرده است.

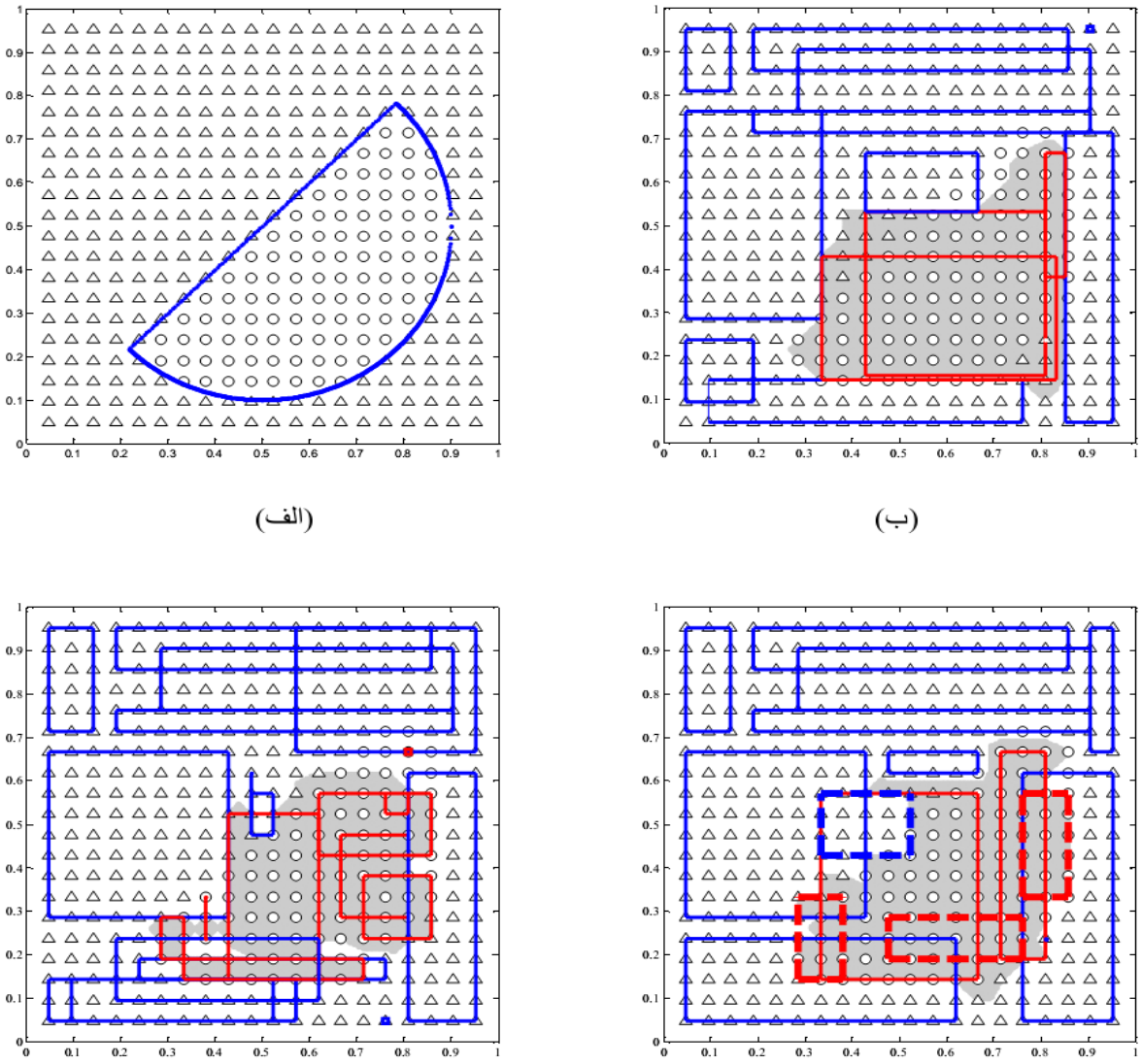
آزمایش چهارم: در این آزمایش، کارایی شبکه‌ها روی داده گل زنبق[2] سنجیده شده است. این داده دارای ۱۵۰ نمونه در سه کلاس مختلف (هر کلاس ۵۰ نمونه)، با طول بردار ویژگی ۴ می‌باشد. در این آزمایش، روش پیشنهادی روی داده‌های آموزش نتایج بهتری از بقیه روش‌ها بدست آورده است. ولی روی داده‌های تست پاسخ خیلی بهتر از FMNN حاصل کرده است. FMCN نسبت به شبکه پیشنهادی پاسخ بهتری روی داده تست حاصل کرده است ولی شبکه پیشنهادی، پاسخ بسیار نزدیک به پاسخ FMCN، با نرون‌های کمتر حاصل کرده است. شکل ۴-۹ نتایج این آزمایش را نشان می‌دهد.

آزمایش پنجم: در این آزمایش، کارایی شبکه‌ها روی داده WDBC سنجیده شده است. این داده دارای ۵۶۹ نمونه ۳۰ بعدی می‌باشد، که ۳۵۷ عدد مربوط به کلاس اول با برچسب خوش‌خیم و ۲۱۲ داده مربوط به کلاس دوم، با برچسب بدخیم می‌باشد. شکل ۴-۱۰ نتایج این آزمایش را نشان می‌دهد. در این آزمایش، FMCN نتایج بهتری از بقیه روش‌ها حاصل کرده است.

آزمایش ششم: در این آزمایش، کارایی شبکه‌ها روی داده شراب[3] سنجیده شده است. این داده دارای ۱۷۸ نمونه ۱۳ بعدی می‌باشد، که ۵۹ عدد مربوط به کلاس اول، ۷۱ داده مربوط به کلاس دوم و ۴۸ داده مربوط به کلاس سوم می‌باشد. شکل ۴-۱۱ نتایج این آزمایش را نشان می‌دهد. در این آزمایش، FMCN نتایج بهتری از بقیه روش‌ها حاصل کرده است. داده‌های استفاده شده در پایان‌نامه، در [۲۴] موجود است.

---

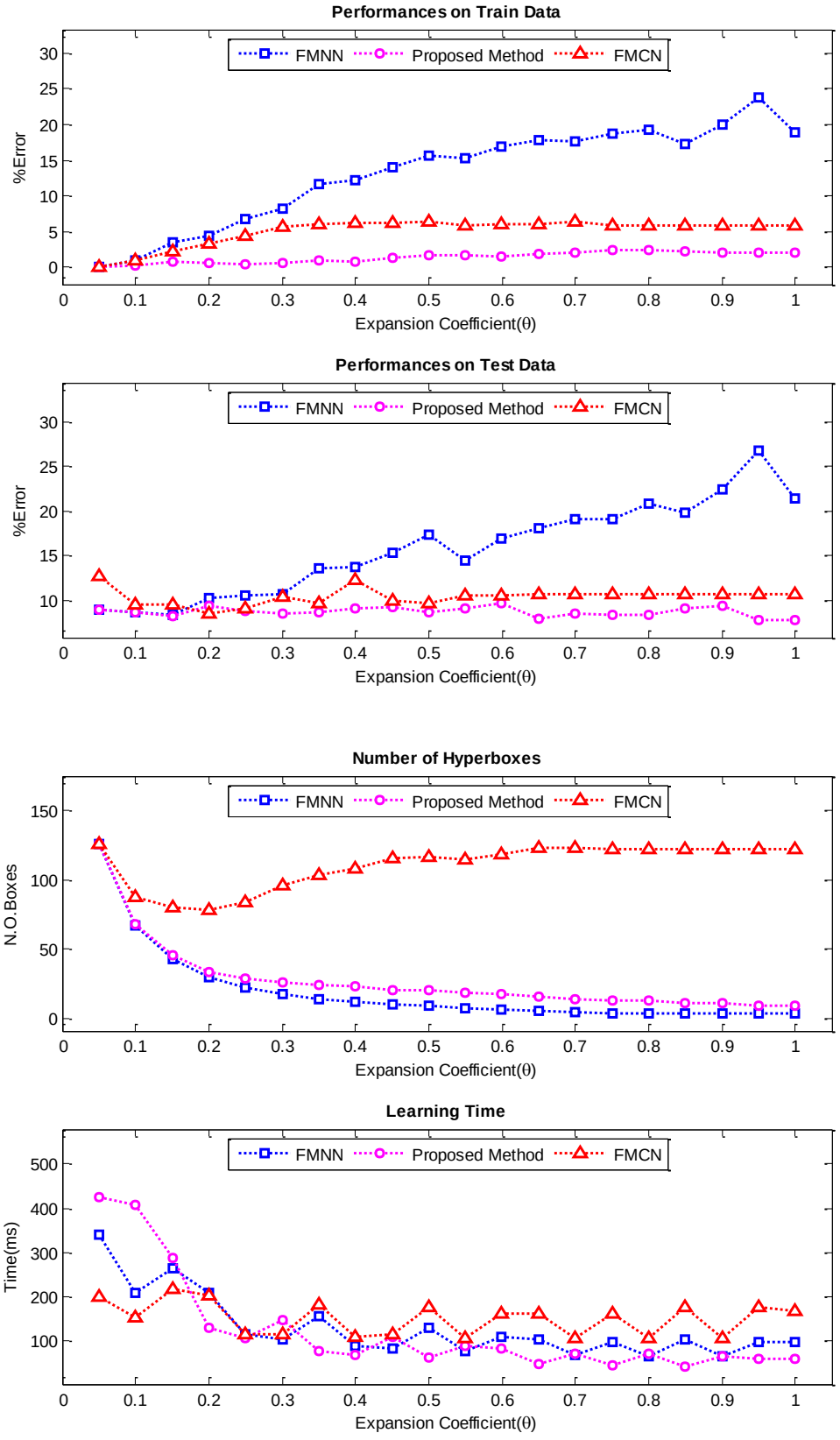
1 - Glass
2 - Iris
3 - Wine



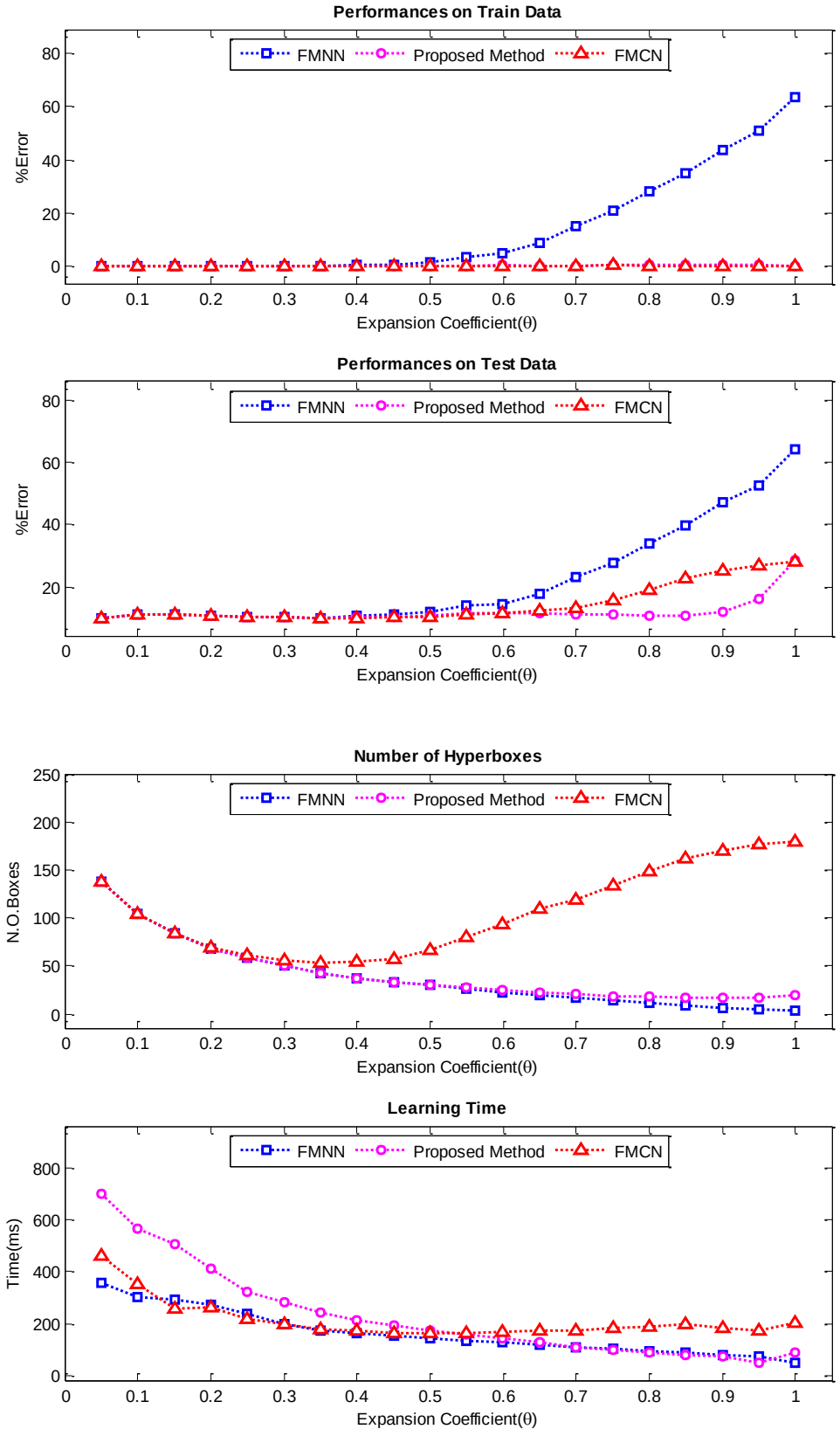
(الف) (ب)

(ج) (د)

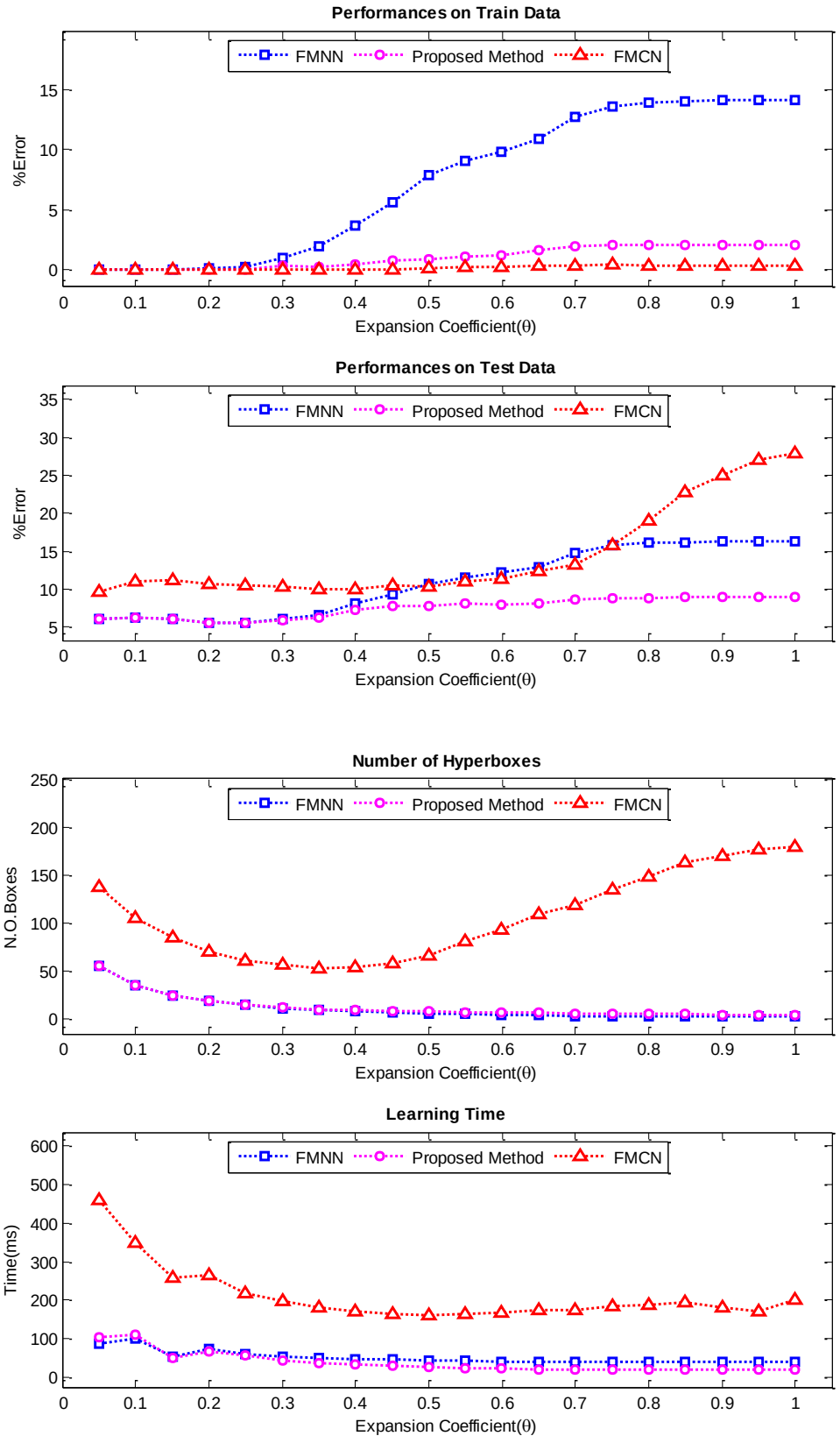
شکل ۴-۵: (الف)داده‌های دو بعدی استفاده شده در آزمایش اول، مرز ایجاد شده بین دو کلاس، توسط شبکه‌های (ب) FMNN، (ج) FMCN و (د)روش پیشنهادی



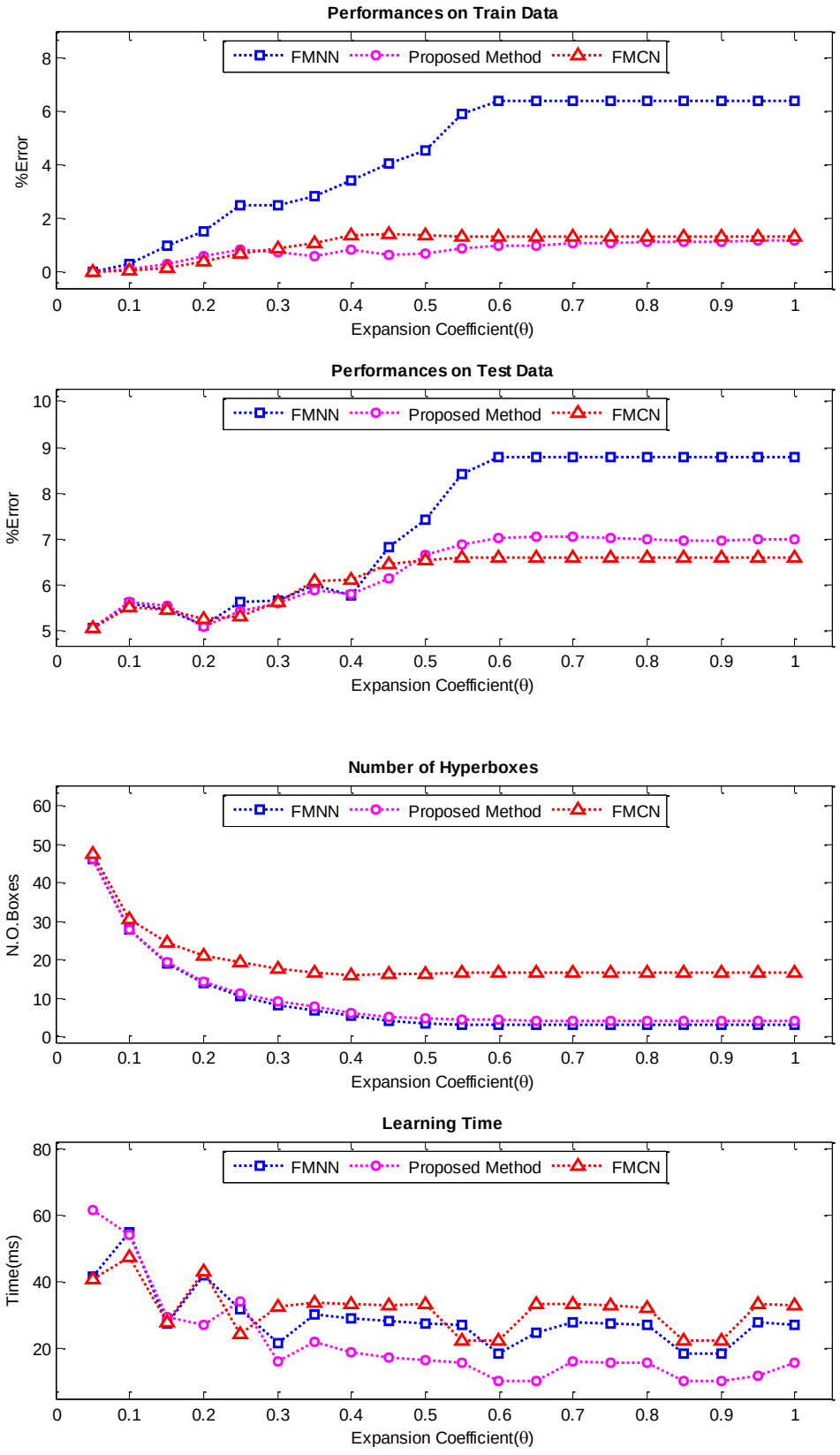
شکل ۴-۶: نتایج حاصل از آزمایش اول روی داده نیم‌دایره



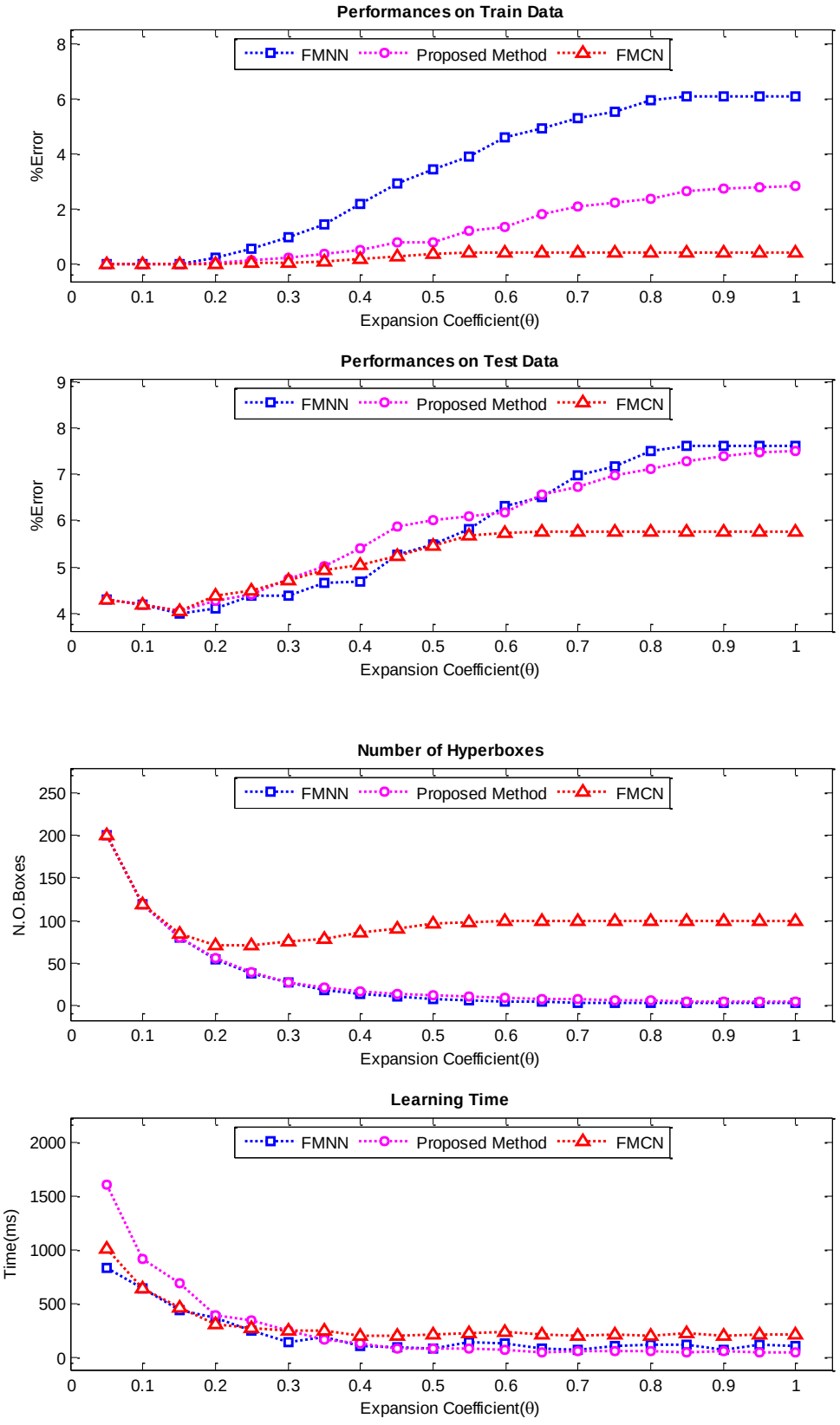
شکل ۴-۷: نتایج حاصل از آزمایش دوم روی داده یونوسفر



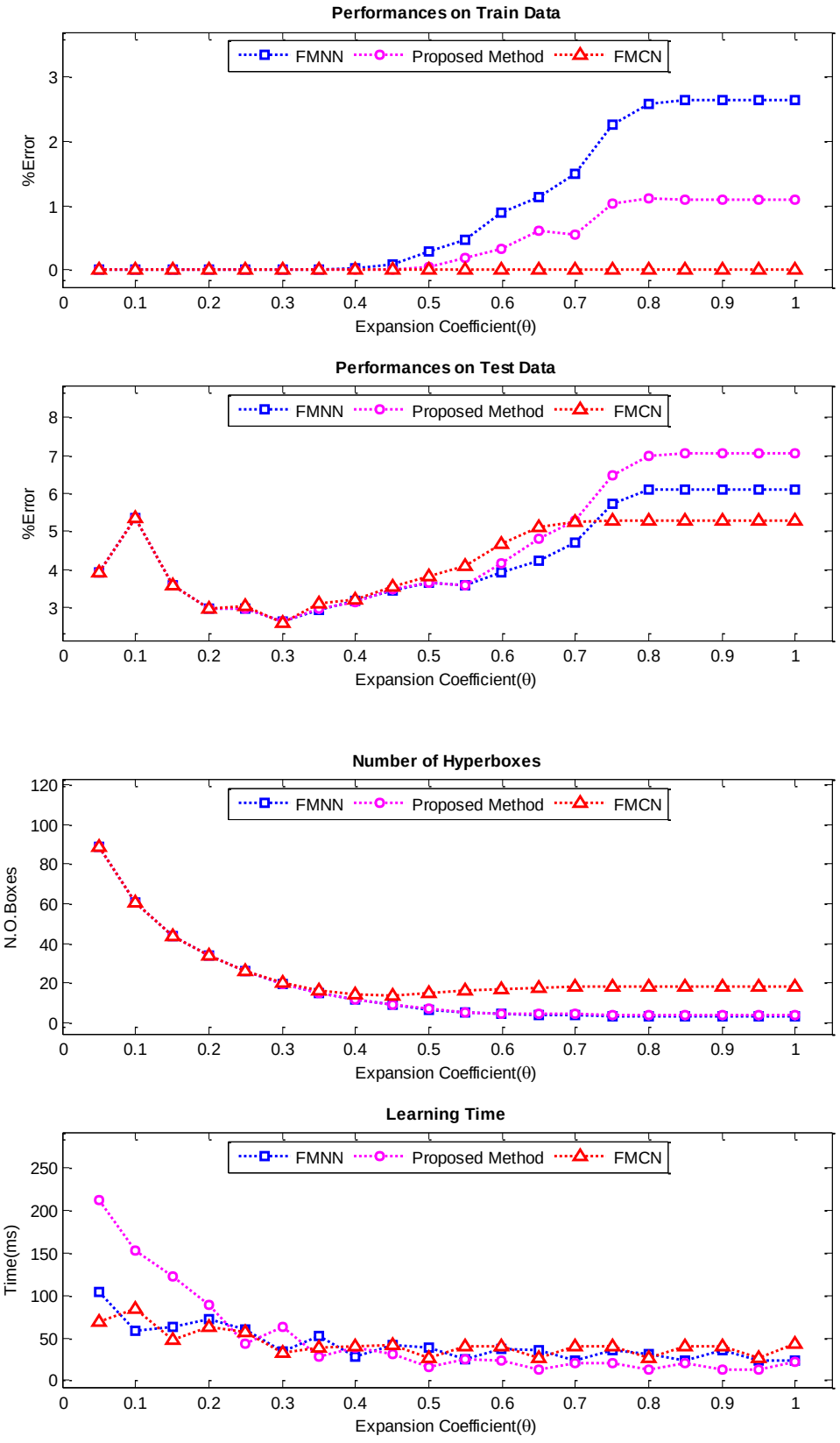
شکل ۴-۸: نتایج حاصل از آزمایش سوم روی داده شیشه



شکل ۴-۹: نتایج حاصل از آزمایش چهارم روی داده گل زنبق



شکل ۴-۱۰: نتایج حاصل از آزمایش پنجم روی داده WDBC



شکل ۴-۱۱: نتایج حاصل از آزمایش ششم روی داده شراب

۴-۴- نتیجه گیری

در این فصل، یک روش جدید برای طبقه بندی الگو، بر پایه ابرجعبه‌های فازی Min-Max ارائه شد. شبکه ارائه شده با یکبار ارائه داده‌های ورودی آموزش می‌بیند. یک الگوریتم ساده برای آموزش شبکه ارائه شد که در مقایسه با شبکه‌ی FMCN، با تعداد نرون‌های کمتر نتایج قابل قبولی تولید می‌کند. یک ساختار دو طبقه که رفتار شبکه را در فاز تست شبیه‌سازی می‌کند ارائه شد. با آزمایش‌های متعدد، کارایی شبکه با روش‌های مشابه مقایسه شد. نتایج آزمایش‌ها بیانگر کارایی خوب شبکه روی داده‌های آموزش و آزمایش می‌باشد. در همه این آزمایش‌ها، به ازای ضرایب انبساط بزرگ، شبکه پیشنهادی، بسیار سریعتر عمل می‌کند. به ازای ضرایب انبساط کوچک، ابرجعبه‌های طبقه دوم، اندازه کوچکی دارد که همین امر، سرعت الگوریتم را کاهش می‌دهد. در دو آزمایش آخری، مربوط به داده‌های WDBC و شراب، شبکه پیشنهادی نسبت به FMCN، نتایج بدتری تولید می‌کند.

جدول ۵-۲ روش‌های عصبی فازی Min-Max، که در پایان‌نامه حاضر بیان شده‌اند را با هم مقایسه می‌کند. نتایج حاصله، از میانگین نتایج شبکه‌ها به ازای ضرایب انبساط مختلف، حاصل شده است (یعنی هر عدد در جدول ۵-۲ میانگین نتایج هر نمودار نشان داده شده در شکل ۴-۶ تا شکل ۴-۱۱ می‌باشد). همانطور که ملاحظه می‌شود، از لحاظ کمترین تعداد ابرجعبه‌های تولیدی، روش FMNN، رتبه نخست را دارا می‌باشد و پس از آن روش ارائه شده قرار می‌گیرد. ولی نتایج روش ارائه شده روی داده‌های آموزش و آزمایش به مراتب بهتر از روش FMNN می‌باشد. روش FMCN و روش ارائه شده، روی داده‌های آموزش و آزمایش نتایج مناسبی تولید می‌کنند. در بعضی داده‌ها روش ارائه شده و در بعضی دیگر روش FMCN نتایج بهتری تولید می‌کند. در جدول ۵-۲ بهترین نتیجه روی هر آزمایش، با ارقام ضخیم‌تر نشان داده شده‌اند.

جدول ۴-۲: مقایسه نتایج حاصله از روش‌های مختلف عصبی فازی Min-Max

داده	شبکه	خطای آموزش	خطای آزمایش	تعداد ابرجعبه	زمان آموزش
نیم دایره	FMNN	۱۳,۱۵	۱۵,۸۳	**۱۹,۷۹**	۱۲۸,۲۰
	FMCN	۹,۲۹	۱۳,۸۷	۳۰۰,۵۵	۲۴۹,۹۹
	Proposed	**۱,۴۰**	**۸,۷۲**	۲۷,۴۶	**۱۲۱,۸۰**
یونوسفر	FMNN	۱۳,۸۹	۲۲,۰۸	**۳۹,۰۲**	**۱۶۲,۵۲**
	FMCN	**۰,۱۹**	۱۴,۵۰	۱۰۵,۹۵	۲۱۱,۴۴
	Proposed	۰,۲۷	**۱۲,۰۱**	۴۲,۳۹	۲۳۱,۳۰
شیشه	FMNN	۷,۳۵	۱۰,۹۴	**۱۰,۷۰**	۵۱,۲۱
	FMCN	**۰,۱۹**	۱۴,۵۰	۱۰۵,۹۵	۲۱۱,۴۴
	Proposed	۱,۰۷	**۷,۵۷**	۱۲,۲۶	**۳۸,۰۱**
گل زنبق	FMNN	۴,۳۰	۷,۳۰	**۸,۸۲**	۲۸,۸۰
	FMCN	۱,۰۵	**۶,۱۷**	۱۹,۶۲	۳۱,۷۲
	Proposed	**۰,۸۱**	۶,۳۴	۹,۷۲	**۲۱,۳۱**
WDBC	FMNN	۳,۳۲	۵,۸۱	**۲۹,۷۳**	**۲۰۹,۹۹**
	FMCN	**۰,۲۷**	**۵,۲۲**	۹۸,۴۶	۲۹۵,۹۳
	Proposed	۱,۲۵	۵,۸۷	۳۱,۹۶	۲۶۱,۵۸
شراب	FMNN	۰,۹۹	۴,۳۶	**۱۷,۵۵**	**۴۲,۳۶**
	FMCN	**۰,۰۰**	**۴,۳۰**	۲۵,۴۸	۴۳,۱۵
	Proposed	۰,۴۱	۴,۷۱	۱۷,۸۷	۴۸,۸۲

فصل پنجم

پیاده‌سازی الگوریتم پیشنهادی

روی FPGA

در این فصل یک طرح سخت‌افزاری، برای پیاده‌سازی شبکه پیشنهادی در پایان‌نامه، ارائه شده است که در ادامه اجزای تشکیل دهنده آن را مورد بررسی قرار می‌دهیم. شبکه مورد نظر برای طبقه‌بندی گل زنبق طراحی شده که دارای بردار ویژگی ۴ بعدی و سه کلاس خروجی می‌باشد. ساختار سخت‌افزاری ارائه شده را به راحتی می‌توان برای طبقه‌بندی سایر داده‌ها تغییر داد. برای کاهش منابع سخت‌افزاری استفاده شده و تأخیر کلی مدار، تغییراتی در تابع عضویت ایجاد شده است که در ادامه بحث می‌شود.

فاز آموزش شبکه با استفاده از نرم‌افزار MATLAB اجرا شده است و پارامترهای مربوط به ابرجعبه‌ها، برای اجرای فاز تست استخراج شده‌اند. این پارامترها عبارتند از: نقاط مینیمم و ماکزیمم ابرجعبه‌ها، برچسب ابرجعبه‌ها و شماره طبقه‌ای که ابرجعبه متعلق به آن است.

۵-۱- شبکه پیشنهادی با تابع عضویت ساده شده

با در نظر گرفتن رابطه (۴-۵) روابط زیر را می‌توانیم بنویسیم:

$$\begin{aligned} b_j(X_h, V_j, W_j) &= \frac{1}{n}\sum_{i=1}^{n} \min(1 - f(x_{hi} - w_{ji}, \gamma), 1 - f(v_{ji} - x_{hi}, \gamma)) \\ &= \tfrac{1}{n}\textstyle\sum_{i=1}^{n}(1 - \max(f(x_{hi} - w_{ji}, \gamma), f(v_{ji} - x_{hi}, \gamma))) \\ &= 1 - \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \max(f(x_{hi} - w_{ji}, \gamma), f(v_{ji} - x_{hi}, \gamma)) \end{aligned} \quad (۱-۵)$$

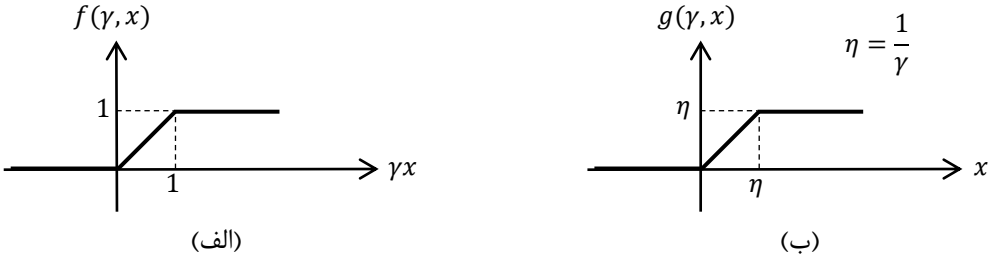
با توجه به صعودی بودن تابع f می‌توان نوشت:

$$\begin{aligned} b_j(X_h, V_j, W_j) &= 1 - \frac{1}{n}\sum_{i=1}^{n} f(\gamma, \max(x_{hi} - w_{ji}, v_{ji} - x_{hi})) \\ &= 1 - \tfrac{1}{n}\textstyle\sum_{i=1}^{n} g(\max(x_{hi} - w_{ji}, v_{ji} - x_{hi})) \end{aligned} \quad (۲-۵)$$

که در آن تابع g به صورت زیر تعریف می‌شود:

$$g(x,\gamma)=\begin{cases}\eta, & x>\eta\\ x, & \eta\geq x\geq 0\\ 0, & x<0\end{cases},\eta=\frac{1}{\gamma} \qquad (۳-۵)$$

اشاره به این نکته می‌تواند مفید باشد که تابع f در رابطه (۲-۱۷) و تابع g در رابطه (۵-۳)، در واقع توابع اشباع[1] می‌باشند. این توابع دارای دو آستانه بالا و پایین می‌باشند. ورودی‌های بزرگ‌تر از آستانه بالا به این آستانه اشباع می‌شوند. به طور مشابه، ورودی‌های کوچک‌تر از آستانه پایین نیز به این آستانه اشباع می‌شوند. ورودی‌های بین دو آستانه بدون تغییر عبور می‌کنند. شکل ۵-۱ این توابع را نمایش می‌دهد.



شکل ۵-۱: الف) تابع $\boldsymbol{f}$ در رابطه (۲-۱۷)، ب) تابع $\boldsymbol{g}$ در رابطه (۵-۳)

از طرفی خروجی یک طبقه، طبق ابطه (۲-۱۵) حاصل می‌شود. این رابطه را به صورت زیر می‌توان بسط داد:

$$\begin{aligned}C_k &= \max_{j=1,\dots,m}(b_j u_{kj})\\ &= \max_{j=1,\dots,m}\left\{\left(1-\frac{1}{n}\sum_{i=1}^{n} g\big(\max(x_{hi}-w_{ji},v_{ji}-x_{hi})\big)\right)u_{kj}\right\}\\ &= \max\left\{\max_{j\in S_{k0}}\left\{\left(1-\frac{1}{n}\sum_{i=1}^{n} g\big(\max(x_{hi}-w_{ji},v_{ji}-x_{hi})\big)\right)u_{kj}\right\},\right.\\ &\qquad \left.\max_{j\in S_{k1}}\left\{\left(1-\frac{1}{n}\sum_{i=1}^{n} g\big(\max(x_{hi}-w_{ji},v_{ji}-x_{hi})\big)\right)u_{kj}\right\}\right\} \qquad (۴-۵)\end{aligned}$$

در رابطه (۵-۴)، S_{k0} مجموعه ابرجعبه‌هایی است که متعلق به کلاس C_k نمی‌باشند و S_{k1} مجموعه ابرجعبه‌هایی هستند که متعلق به کلاس C_k می‌باشد. یعنی:

1 - Saturation

$$u_{kj} = \begin{cases} 0, & b_j \in S_{k0} \\ 1, & b_j \in S_{k1} \end{cases} \quad (۵-۵)$$

با جایگذاری u_{kj} در رابطه بالا، این رابطه به صورت زیر ساده‌سازی می‌شود:

$$C_k = \max\left\{0\ , \max_{j\in S_{k1}}\left\{\left(1 - \frac{1}{n}\sum_{i=1}^{n} g\left(\max\left(x_{hi} - w_{ji}, v_{ji} - x_{hi}\right)\right)\right) \times 1\right\}\right\}$$
$$= \max_{j\in S_{k1}}\left(1 - \frac{1}{n}\sum_{i=1}^{n} g\left(\max\left(x_{hi} - w_{ji}, v_{ji} - x_{hi}\right)\right)\right)$$
$$= 1 - \min_{j\in S_{k1}}\left(\frac{1}{n}\sum_{i=1}^{n} g\left(\max\left(x_{hi} - w_{ji}, v_{ji} - x_{hi}\right)\right)\right) \quad (۶-۵)$$

برای سادگی بیشتر، از ترم دوم رابطه (۵-۶) استفاده می‌کنیم و در خروجی، نرونی که کمترین خروجی را تولید کرده است را به عنوان نرون برنده در نظر می‌گیریم. بنابراین خروجی هر طبقه، به صورت زیر بیان می‌شود:

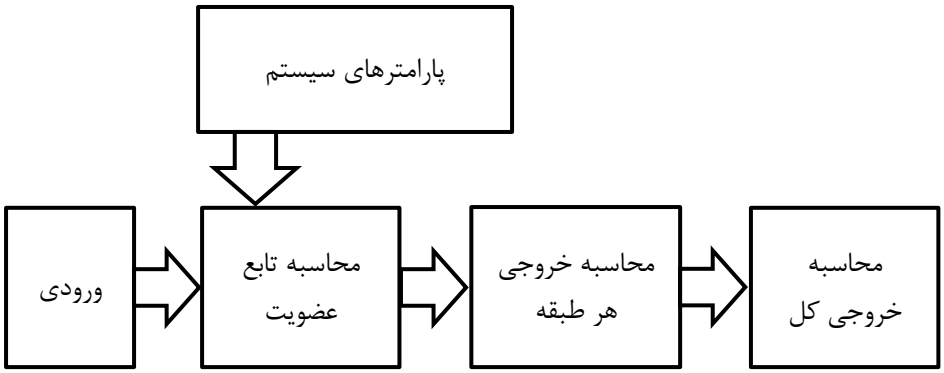
$$C_k = \min_{j\in S_{k1}} \frac{1}{n}\sum_{i=1}^{n} g\left(\max\left(x_{hi} - w_{ji}, v_{ji} - x_{hi}\right)\right) \quad (۷-۵)$$

نتایج شبیه‌سازی نشان داد که روش ساده شده نتایج یکسانی با روش معمولی حاصل می‌کند. فاز آموزش شبکه روی داده گل زنبق چند بار تکرار شد و پارامترهای حاصل از بهترین نتیجه، برای پیاده‌سازی روی FPGA استفاده شده است. در فاز آموزش، مقدار پارامتر $\eta = 0.25$، $\theta^{(1)} = 0.5$ و $\theta^{(2)} = 0.1$ در نظر گرفته شد. پس از آموزش ۷ ابرجعبه تولید شد که ۴ عدد متعلق به طبقه اول و ۳ عدد دیگر متعلق به طبقه دوم می‌باشد. دقت طبقه‌بندی برای داده‌های آموزش ۱۰۰ درصد و برای داده‌های تست ۹۸,۶۶ درصد حاصل شد.

۵–۲– طراحی شبکه پیشنهادی روی FPGA

شکل ۵-۲ بلوک دیاگرام کلی سیستم طراحی شده را نشان می‌دهد. برای بررسی صحت عملکرد سیستم، کل ۱۵۰ داده ورودی را در ۴ حافظه ROM ذخیره کرده‌ایم هر بار که یک سطر از این حافظه‌ها خوانده می‌شود سیستم خروجی مربوطه به آن داده را حساب می‌کند و پس از اتمام عملیات

آدرس بعدی را برای ROM تولید می‌کند تا داده بعدی را مورد آزمون قرار دهد. این کار صرفاً برای تست سیستم می‌باشد و در سیستم‌های واقعی، ورودی از سخت‌افزار مربوط به استخراج ویژگی داده‌های طبیعی (مثل تصویر چهره یا اثر انگشت) حاصل می‌شود.



شکل ۵-۲: بلوک دیاگرام کلی سیستم سخت‌افزاری

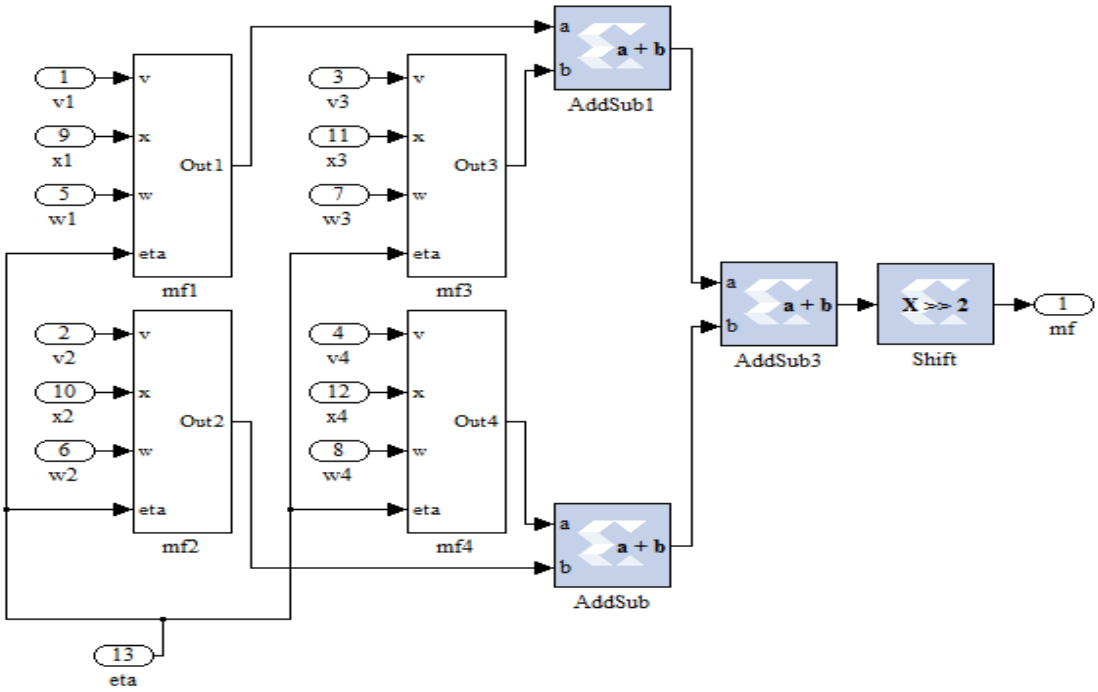
ساختار طراحی شده قادر است در ۱۲ کلاک، خروجی مربوط به یک ورودی را محاسبه نماید. سیستم از ساختار موازی برخوردار بوده و برای محاسبه تابع عضویت، بردار ورودی را به کل گرفته و روی هر ۴ بعد، همزمان محاسبات را انجام می‌دهد. ولی سیستم در کل یک زیر سیستم مربوط به محاسبه تابع عضویت دارد و برای هر ابرجعبه یک تابع عضویت طراحی نشده است. سیستم به جای محاسبه موازی مقدار عضویت هر ابرجعبه، از تکنیک خط لوله در این قسمت استفاده می‌کند و در هر کلاک، اطلاعات مربوط به یک ابرجعبه را از حافظه RAM خوانده و به واحد محاسبه تابع عضویت می‌فرستد. چون ۷ ابرجعبه وجود دارد، ۷ کلاک طول می‌کشد تا محاسبه تابع عضویت برای همه آنها صورت گیرد. ولی این در حالتی امکان‌پذیر است که قسمت‌های دیگر سیستم تأخیر نداشته باشند. با اضافه کردن ۵ تأخیر دیگر که مربوط به سه بلوک محاسباتی می‌باشد، خروجی مربوط به هر ورودی در ۱۲ کلاک حاصل می‌شود.

۵-۲-۱- زیرسیستم تابع عضویت

زیرسیستم تابع عضویت در شکل ۵-۳ نشان داده شده است. این زیر سیستم ۱۳ ورودی دارد که ۸ ورودی مربوط به نقاط مینیمم و ماکزیمم ابرجعبه، ۴ ورودی مربوط به داده تست و ورودی ۱۳ مربوط به پارامتر η می‌باشد که در این طرح برابر ۰,۲۵ است. این زیرسیستم خود از ۴ زیرسیستم مشابه، که هر کدام محاسبات مربوط به یک بعد را انجام می‌دهند، تشکیل شده است. خروجی کلی برابر با میانگین این ۴ بلوک است که در سه بلوک جمع‌کننده بعدی و بلوک شیفت به راست انجام می‌گیرد. در اینجا چون بعد ورودی ۴ بوده است، با استفاده از شیفت، خروجی حاصل شده است. ولی در کل یک تقسیم کننده به جای بلوک شیفت به راست، قرار می‌گیرد.

لازم به ذکر است که برای محاسبه رابطه (۵-۲)، نتیجه حاصله در بلوک بالا باید از مقدار ۱ تفریق شود ولی چون از رابطه (۲-۱۵) برای تعیین خروجی استفاده می‌کنیم این مرحله حذف شده است.

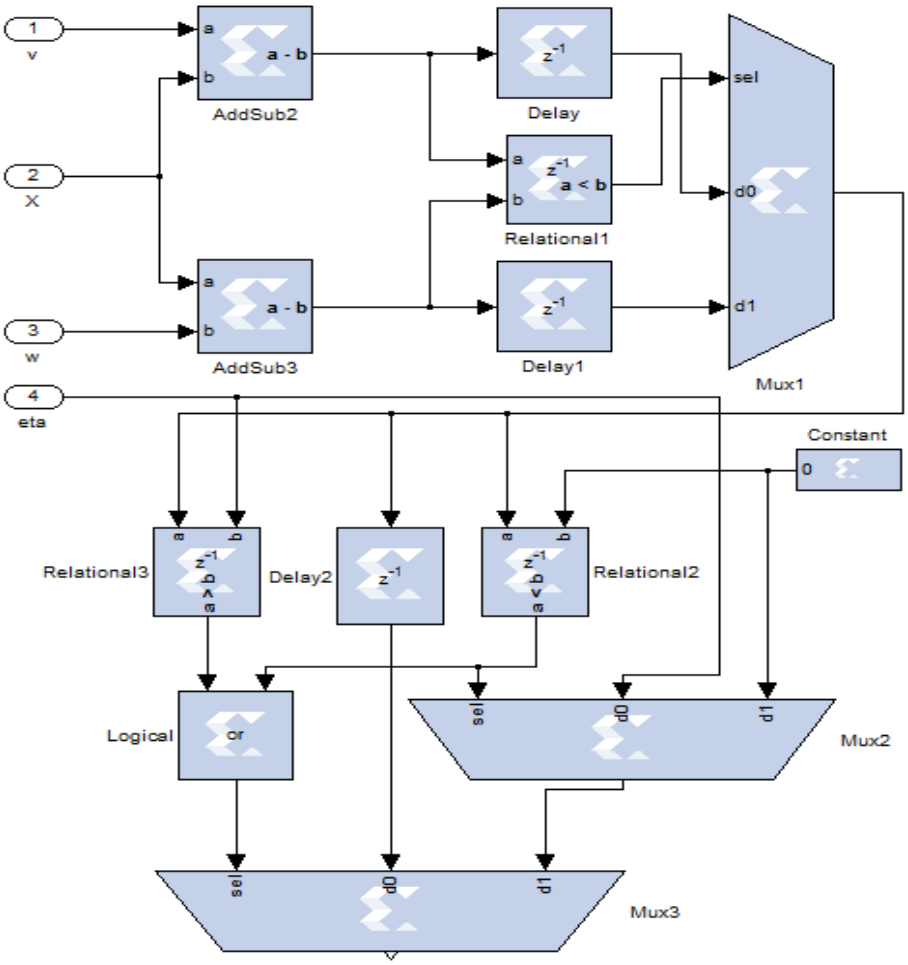
پارامتر η در رابطه (۵-۳) می‌تواند هر مقدار مثبتی داشته باشد. ولی در عمل وقتی η برابر ۱ است، تابع عضویت ابرجعبه‌ای که فقط از یک نقطه تشکیل شده است و در گوشه فضای ورودی (مثلاً نقطه (۱،۱) برای مسأله دو بعدی) قرار دارد، کل فضای مسأله را پوشش می‌دهد. بنابراین ما عموماً از مقادیر η بین صفر و یک استفاده می‌کنیم. بنابراین خروجی بلوک AddSub3 در شکل ۵-۳ حداکثر برابر n (طول بردار ویژگی) می‌باشد. از اینرو برای مقادیر کوچک n با تخصیص تعداد بیت‌های صحیح مناسب برای نمایش این مقادیر، می‌توان بلوک مربوط به تقسیم کننده را نیز حذف کرد. برای این کار لزومی ندارد این نوع کوانتیزه‌سازی در کل شبکه رعایت شود و فقط برای قسمتی که مقایسه بین مقادیر عضویت صورت می‌گیرد این عمل را انجام می‌دهیم.



شکل ۵-۳: زیرسیستم تابع عضویت

ساختار داخلی زیرسیستم‌های mf1 تا mf4 در شکل ۵-۳ به صورت نشان داده شده در شکل ۵-۴ می‌باشد. در این شکل بلوک‌های بالایی (Delay1, Delay, Relational, AddSub3, AddSub3, Mux1) مقدار عبارت $max(X_h - W_j, V_j - X_h)$ در رابطه (۵-۷) را محاسبه می‌کنند. بلوک‌های تأخیری برای کارکرد صحیح سیستم در حالت خط لوله، قرار داده شده‌اند. عملکرد این ۶ بلوک برای محاسبه رابطه گفته شده، به روشنی معلوم است و از توضیح اضافه پرهیز می‌کنیم.

بلوک‌های پایینی در شکل ۵-۴ برای محاسبه تابع g به کار می‌روند. مقادیر آستانه این تابع η و صفر می‌باشد. η جزء ورودی‌های زیرسیستم است و صفر در بلوک Constant1 قرار دارد. بلوک‌های Relational2,3 مقدار ورودی (خروجی بلوک‌های بالایی) را با مقادیر آستانه مقایسه می‌کنند. خروجی این دو باهم OR می‌شوند و به ورودی انتخاب Mux3 متصل می‌شوند. نحوه ایجاد خروجی زیرسیستم از روی شکل مشخص می‌باشد.

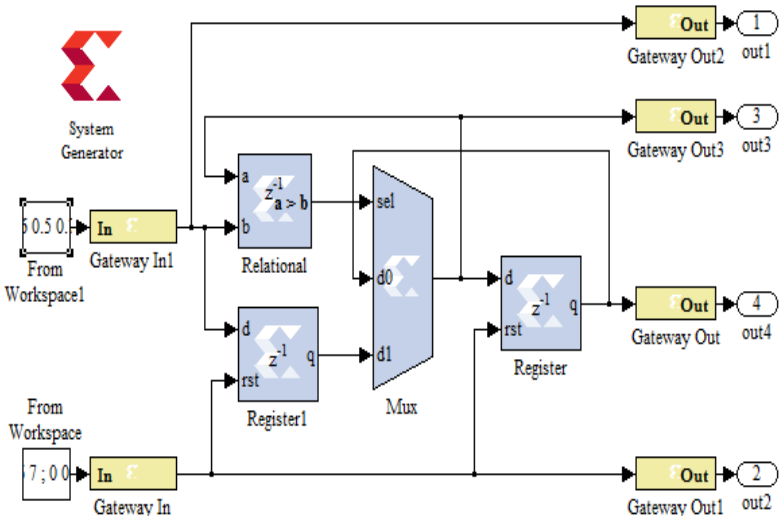


شکل ۵-۴: ساختار داخلی زیرسیستم‌های mf

۵-۲-۲- زیرسیستم محاسبه خروجی یک طبقه

پس از محاسبه تابع عضویت در زیرسیستم مربوطه، مقدار حاصل به یکی از ۶ زیرسیستم محاسبه خروجی طبقه، هدایت می‌شود. این انتخاب توسط دو پارامتر خوانده شده از RAM یعنی شماره طبقه و شماره کلاس خروجی (۶=۲×۳)، صورت می‌گیرد. شکل ۵-۵ ساختار داخلی یکی از این ۶ زیرسیستم را نشان می‌دهد. بلوک‌های تشکیل دهنده این زیرسیستم، بلوک‌های Mux، Relational، Register و Register1 می‌باشند. بقیه بلوک‌ها برای تست زیرسیستم قرار داده شده‌اند و صرفاً به منظور توضیح کارکرد زیرسیستم آورده شده‌اند. نتایج حاصل از اجرای ۱۱ سمپل زمانی در Simulink در شکل ۵-۵-ب نشان داده شده است. این زیر سیستم خروجی خود را با دو تأخیر ایجاد می‌کند. بنابراین برای ایجاد نتایج درست، پس از وارد شدن مقدار عضویت مربوط به آخرین ابرجعبه، زیر

سیستم باید دو کلاک دیگر کار کند. این زیر سیستم مقدار کمینه ورودی‌های متوالی خود را حساب کرده و ذخیره می‌کند. این عمل رابطه (۵-۷) را پیاده‌سازی می‌کند.



(الف)

	1	2	3	4
1	0.6000	0	1.9999	1.9999
2	0.5000	0	0.6000	1.9999
3	0.7000	0	0.5000	0.6000
4	0.2000	0	0.5000	0.5000
5	0.1000	0	0.2000	0.5000
6	0	1	0.1000	0.2000
7	0.3000	0	1.9999	1.9999
8	0.4000	0	0.3000	1.9999
9	0.5000	0	0.3000	0.3000
10	0.6000	0	0.3000	0.3000
11	0.7000	0	0.3000	0.3000

(ب)

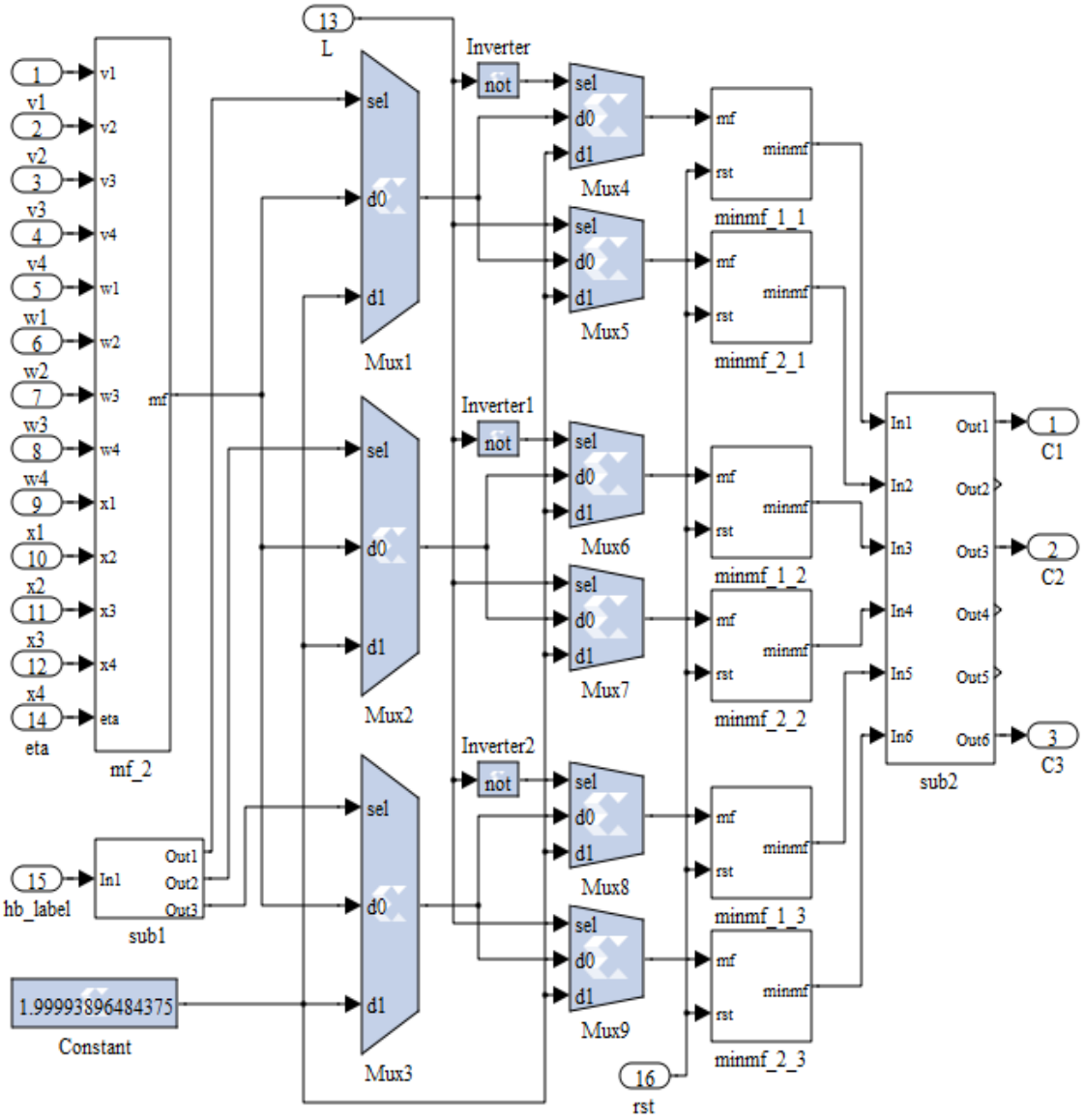
شکل ۵-۵: زیرسیستم محاسبه خروجی یک طبقه

در شکل ۵-۵-الف، بلوک Register مقدار کمینه ورودی زیرسیستم را در خود نگه می‌دارد. در لحظه ریست، مقدار رجیسترها برابر با یک مقدار بزرگ (مثلاً ۲)، قرار می‌گیرد. این مقدار باید از بزرگ‌ترین مقدار عضویت (η) بزرگ‌تر باشد. در شکل ۵-۵-ب، ستون اول مربوط به مقادیر عضویت تولید شده در زیرسیستم تابع عضویت می‌باشد. ستون دوم سیگنال ریست را نشان می‌دهد و ستون چهارم خروجی مدار می‌باشد. هر رجیستر یک تأخیر دارد. بنابراین در لحظه ریست مقدار اولیه رجیستر در خروجی آن ظاهر نمی‌شود و این امر در کلاک بعدی صورت می‌گیرد.

۵-۲-۳- نحوه ارتباط زیرسیستم‌ها

شکل ۵-۶ نحوه ارتباط زیرسیستم‌های معرفی شده و تولید خروجی نهایی را نشان می‌دهد. در این شکل Mux1 تا Mux9، خروجی mf_2 را به یکی از ۶ زیرسیستم minmf منتقل می‌کند. ورودی ۱۳ شماره طبقه را نشان می‌دهد. این ورودی بیت LSB شماره طبقه بوده و برای طبقه اول ۱ و برای

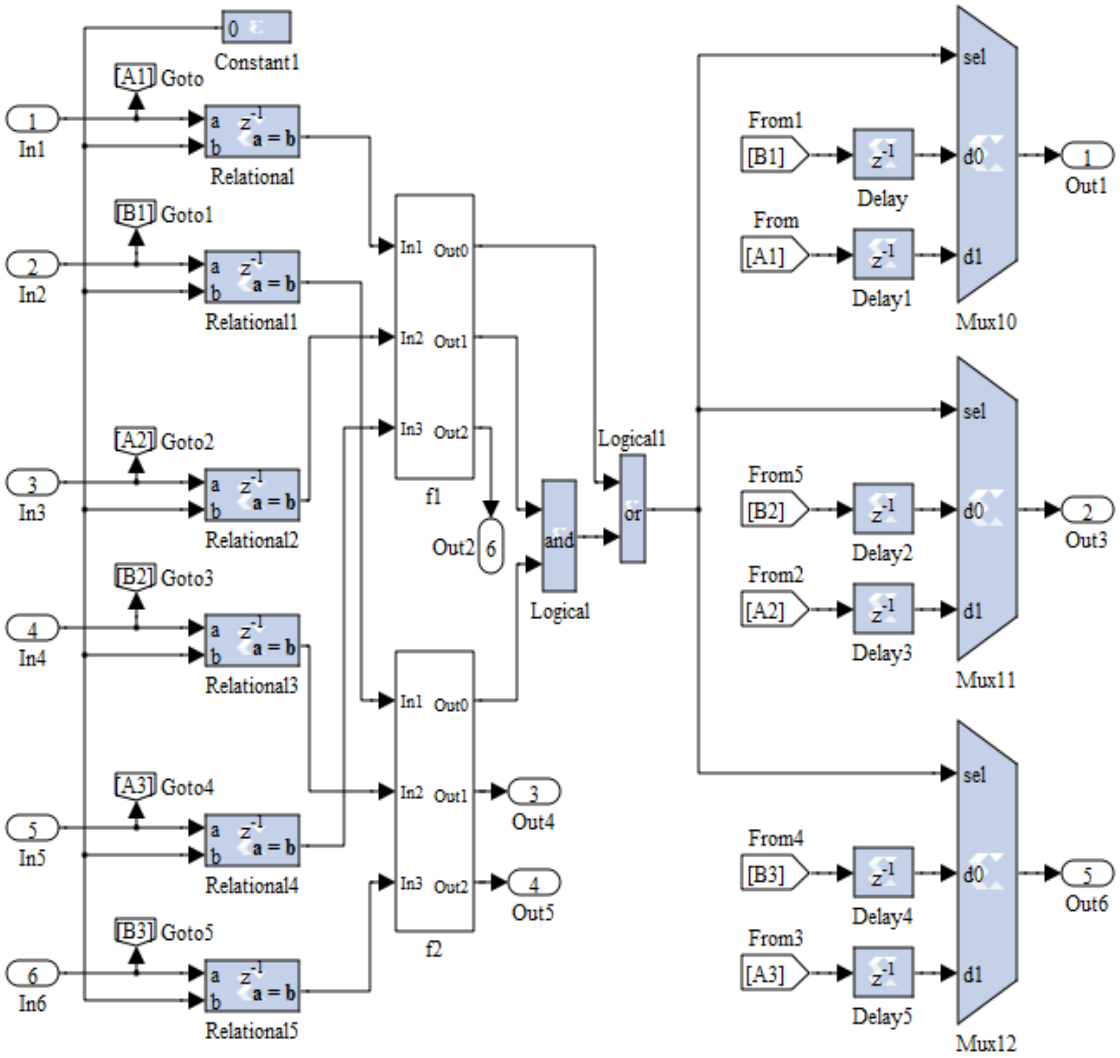
طبقه دوم صفر می‌باشد. ورودی ۱۵ برچسب ابرجعبه را مشخص می‌کند. Sub1 برحسب مقدار برچسب، خروجی mf_2 را به خروجی یکی از مالتی‌پلکسرهای Mux1 تا Mux3 منتقل می‌کند. دو مالتی‌پلکسر دیگر، مقدار Constant را به خروجی خود می‌دهند. Sub2 زیرسیستم محاسبه خروجی کل می‌باشد که خروجی زیرسیستم‌های minmf را می‌گیرد و تصمیم می‌گیرد که خروجی مربوط به کدام طبقه را به خروجی نهایی منتقل کند.



شکل ۵-۶: زیرسیستم کلی محاسبات مربوط به فاز تست شبکه

۵-۲-۴- زیرسیستم محاسبه خروجی کل

شکل ۵-۷ زیرسیستم محاسبه خروجی کل را نشان می‌دهد. این زیرسیستم طبق جدول ۴-۱ تصمیم می‌گیرد که خروجی طبقه ۱ یا ۲ را عبور دهد. در بلوک‌های Relational، هر ۶ ورودی زیرسیستم با صفر مقایسه می‌شوند و نتایج به بلوک‌های f_1 و f_2 منتقل می‌شود. خروجی این دو بلوک ورودی sel بلوک‌های Mux را کنترل می‌کنند.



شکل ۵-۷: زیرسیستم محاسبه خروجی کل

۵-۳- شبیه‌سازی سخت‌افزار طراحی شده

در این بخش، طرح سخت‌افزاری ارائه شده در بخش قبل شبیه سازی شده و نتایج آن با نتایج نرم‌افزاری مقایسه شده است. همچنین بحثی پیرامون نوع کوانتیزاسیون[1] و تعداد بیت‌های مورد استفاده و نوع FPGA استفاده شده، صورت گرفته است.

۵-۳-۱- انتخاب نوع FPGA

خانواده‌های مختلف FPGA، دارای امکانات متفاوتی می‌باشند. همانطورکه در فصل ۳ بیان شد، FPGA ها منابع سخت‌افزاری متعددی در اختیار طراح قرار می‌دهند. برای FPGA های جدید، این منابع می‌تواند شامل هسته‌های میکروکنترلر تعبیه شده نیز باشد که بسته به الگوریتم، می‌تواند مورد استفاده قرار گیرد. هسته‌های تعبیه شده مانند ضرب‌کننده، به تعداد زیاد روی FPGA فراهم شده‌اند که استفاده از آنها باعث طراحی کارا و بهینه می‌شود. از طرفی هر کدام از این امکانات، باعث افزایش قیمت FPGA می‌شود. بنابراین، برای یک طرح، FPGA باید بر حسب نیاز طرح انتخاب شود.

طرح سخت‌افزاری ارائه شده در این فصل، به دلیل سادگی، اجرا شدن در تعداد سیکل کلاک کم و عدم استفاده از عمل ضرب در الگوریتم، نیاز به استفاده از منابع گران قیمت را مرتفع می‌سازد و می‌تواند توسط لاجیک‌های[2] عمومی روی FPGA پیاده‌سازی شود. برای این کار ما از FPGA های خانواده پرکاربرد و متداول Spartan3 شرکت Xilinx استفاده کرده‌ایم.

۵-۳-۲- انتخاب نوع کوانتیزاسیون و تعداد بیت

محاسبات در روی FPGA به دو صورت ممیز ثابت و ممیز شناور قابل پیاده‌سازی می‌باشد. ولی عموماً در پیاده‌سازی‌های FPGA، از نوع داده ممیز ثابت، به خاطر کارایی بالا و مصرف توان و سخت‌افزار کمتر، استفاده می‌شود. یکی دیگر از ویژگی‌های FPGA، قابلیت انتخاب نوع کوانتیزاسیون داده، یعنی تعداد بیت‌های صحیح و اعشاری، می‌باشد. انتخاب تعداد بیت زیاد باعث افزایش دقت

1 - Quantization
2 - Logic

می‌شود، ولی این کار فضای سخت‌افزاری زیادی را می‌طلبد. بنابراین بهتر است تعداد بیت را با توجه به الگوریتمی که می‌خواهیم پیاده‌سازی کنیم، تعیین کنیم.

برای الگوریتم پیاده‌سازی شده در این فصل، ورودی‌ها در بازه [0,1] نرمالیزه شده‌اند. بنابراین نقاط مینیمم و ماکزیمم ابرجعبه‌ها نیز در این بازه قرار می‌گیرند. در فاز تست شبکه، اولین گام محاسبه $(V_j - X_h)$ و $(X_h - W_j)$ می‌باشد (شکل ۵-۴). خروجی این عبارات نیز در بازه $[-1,1]$ واقع می‌شوند. پس از این، تا تولید خروجی نهایی، چند جمع‌کننده وجود دارند که خروجی‌هایشان یک بیت بیشتر از ورودی‌هایشان می‌باشد. این تعداد بیت اضافی تولید شده، در شیفت دهنده بریده می‌شوند. بنابراین برای نشان دادن بخش صحیح، ۲ بیت کفایت می‌کند. برای بخش اعشاری نیز می‌توان چندین آزمایش، با دقت بیت مختلف انجام داده و تعداد بیت مناسب را بدست آورد.

سیستم طراحی شده برای طبقه‌بندی داده گل زنبق استفاده شد و نتایج مربوط به منابع سخت‌افزاری مصرف شده و فرکانس کاری، از شبیه‌سازی‌ها بدست آمد.

۵-۳-۳- نتایج حاصل از شبیه‌سازی‌ها

در این بخش، نتایج حاصل از شبیه سازی با دقت‌های مختلف بیتی، آورده شده است. برای این کار از FPGA، Spartan3 xc3s50-4vq100 استفاده شده است. شبیه‌سازی‌ها به ازای دقت‌های بیتی Q8.6، Q10.8 و Q11.9 صورت گرفته است. به ازای کوانتیزاسیون Q11.9 دقت طبقه‌بندی با FPGA برابر با دقت حاصله از MATLAB می‌باشد، بنابراین برای این کاربرد، نیاز به بیت‌های بیشتری نیست. با کاهش تعداد بیت، خطای طبقه‌بندی افزایش می‌یابد. نتایج حاصل از MATLAB و FPGA در جدول ۵-۱ آورده شده است. ستون مربوط به ماتریس تصمیم‌گیری در این جدول، خطاهای مربوط به هر آزمایش را نشان می‌دهد. در این ماتریس‌ها، سطرها بیانگر کلاس داده و ستون‌ها بیانگر کلاسی است که داده توسط طبقه‌بند به آن منسوب شده است. به عنوان مثال، با دقت Q11.9 یکی از داده‌های کلاس دوم اشتباهاً جزء کلاس سوم محسوب شده است که مشابه نتایج نرم‌افزاری می‌باشد.

جدول ۵-۱: نتایج مربوط به دقت تشخیص و فرکانس کاری برای سخت‌افزار طراحی شده

دقت بیتی	زمان محاسبات	حداکثر فرکانس کاری	درصد خطا	ماتریس تصمیم‌گیری			
					۱	۲	۳
Intel Core i7 MATLAB ممیز شناور	۴۵ ms	۱,۶ GHz	٪۰,۶۶	۱	۵۰	۰	۰
				۲	۰	۴۹	۱
				۳	۰	۰	۵۰
Xilinx Spartan3 xc3s50-4vq100 FPGA ممیز ثابت Q8.6	۲۷,۷۲ us	۶۴,۹۳۹MHz	٪۲		۱	۲	۳
				۱	۵۰	۰	۰
				۲	۰	۴۷	۰
				۳	۰	۳	۵۰
Xilinx Spartan3 xc3s50-4vq100 FPGA ممیز ثابت Q10.8	۲۷,۹۸ us	۶۴,۳۲۱ MHz	٪۱,۳۳		۱	۲	۳
				۱	۵۰	۰	۰
				۲	۰	۴۸	۲
				۳	۰	۰	۵۰
Xilinx Spartan3 xc3s50-4vq100 FPGA ممیز ثابت Q11.9	۲۸,۰۵ us	۶۴,۱۶۴ MHz	٪۰,۶۶		۱	۲	۳
				۱	۵۰	۰	۰
				۲	۰	۴۹	۱
				۳	۰	۰	۵۰

همچنین در جدول فوق، حداکثر فرکانس کاری سیستم مورد نظر و زمانی که این سیستم کل ۱۵۰ داده را طبقه‌بندی کرده است، آورده شده است. برای FPGA زمان محاسبات توسط رابطه زیر محاسبه شده است:

زمان طبقه‌بندی ۱۵۰ داده =۱۵۰× (تعداد سیکل کلاک لازم)۱۲ × (حداکثر فرکانس / ۱)

همانطور که ملاحظه می‌شود، فرکانس کاری کامپیوتر تقریباً ۲۵ برابر فرکانس کاری FPGA است، ولی FPGA محاسبات را تقریباً ۱۶۰۰ برابر سریع‌تر انجام داده است. این امر به دلیل محاسبات موازی در FPGA توجیه پذیر است.

پارامتر دیگری که عموماً برای مقایسه طرح‌های سخت‌افزاری استفاده می‌شود، منابع سخت‌افزاری استفاده شده توسط طرح است. اساساً با افزایش تعداد بیت، منابع مصرفی افزایش می‌یابد. جدول ۵-۲ منابع مصرفی FPGA برای طرح مورد نظر را، به ازای دقت‌های مختلف بیتی نشان می‌دهد.

جدول ۵-۲: تعداد منابع استفاده شده در FPGA برای سخت‌افزار طراحی شده

منابع موجود	تعداد موجود	Q8.6		Q10.8		Q11.9	
		تعداد استفاده شده	درصد استفاده شده	تعداد استفاده شده	درصد استفاده شده	تعداد استفاده شده	درصد استفاده شده
Slices	۷۶۸	۳۷۰	٪۴۸	۴۶۸	٪۶۰	۵۰۳	٪۶۵
Slice Flip Flops	۱۵۳۶	۴۳۰	٪۲۷	۵۲۲	٪۳۳	۵۶۸	٪۳۶
4 input LUTs	۱۵۳۶	۵۷۹	٪۳۷	۷۱۱	٪۴۶	۷۷۷	٪۵۰
logic		۵۰۸		۶۲۴		۶۸۲	
Shift registers		۴		۴		۴	
RAMs		۶۷		۸۳		۹۱	
IOs		۳۵		۴۳		۴۷	
Bounded IOBs	۶۳	۳۴	٪۵۳	۴۲	٪۶۶	۴۶	٪۷۳
GCLKs	۸	۱	٪۱۲	۱	٪۱۲	۱	٪۱۲
BRAMs	۴	۴	٪۱۰۰	۴	٪۱۰۰	۴	٪۱۰۰

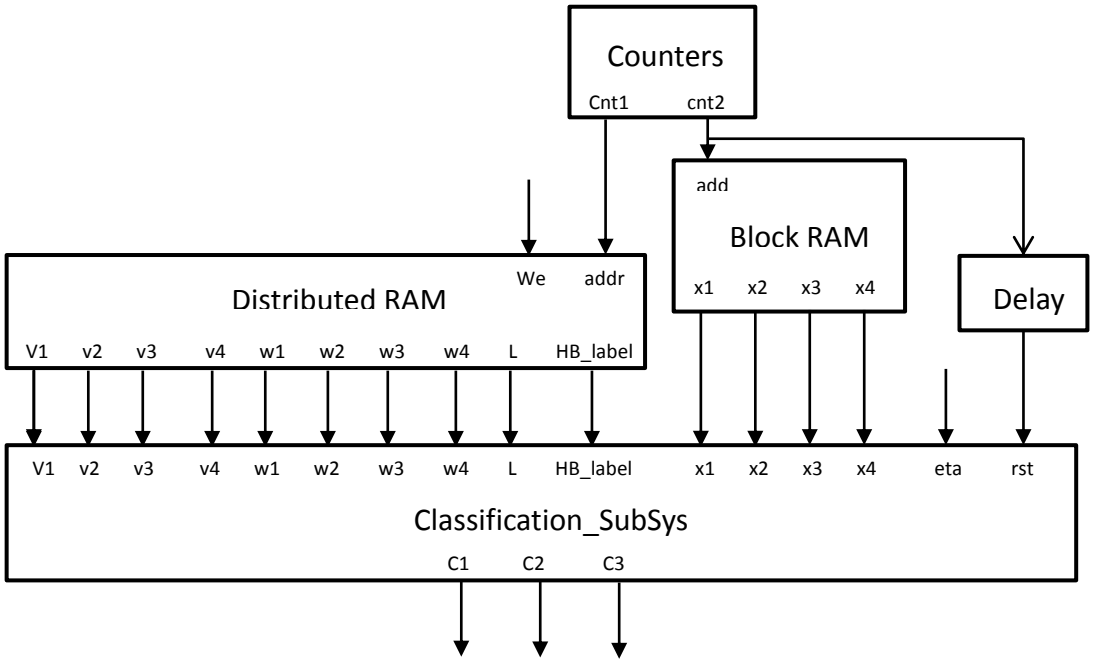
همانطور که از این جدول مشخص است، منابع استفاده شده با افزایش بیت‌ها افزایش یافته است. این امر لزوم انتخاب مناسب تعداد بیت‌ها را مشخص می‌کند.

منابع موجود در روی FPGA، در جدول ۵-۲ آمده است. همانطور که ملاحظه می‌شود، این منابع شامل منابع عمومی می‌باشد و منابع تعبیه شده (بلوک‌های DSP48 در System Generator) در آن استفاده نشده است. این امر نشان می‌دهد که این نوع طبقه‌بند بسیار ساده و به‌صرفه می‌باشد. منابع موجود در FPGA استفاده شده، شامل اسلایس‌ها، فلیپ‌فلاپ‌ها، جداول جستجوی ۴ ورودی، منابع ورودی/خروجی (IOs) و RAM های بلوکی می‌باشد.

LUT ها می‌توانند به صورت لاجیک، شیفت‌رجیستر و RAM پیکربندی شوند. نوع دیگر RAM در روی FPGA، BRAM می‌باشد. در این طرح پارامترهای ابرجعبه‌ها تعداد کمی دارند و از RAM توزیع شده (پیکربندی شده توسط LUT) برای ذخیره آنها استفاده شده است. BRAM ها نیز برای ذخیره ورودی‌ها (به منظور تست شبکه) استفاده شده‌اند.

۵-۴- نتیجه‌گیری

در این فصل یک تابع عضویت ساده شده، برای شبکه ارائه شده در فصل قبل، ارائه شد. شبکه ساده شده، توسط نرم‌افزار MATLAB روی داده گل زنبق آموزش داده شد و پارامترهای ابرجعبه‌ها برای پیاده‌سازی سخت‌افزاری استخراج شد. سپس یک طرح سخت‌افزاری برای اجرای فاز تست شبکه، طراحی شد، شکل ۵-۸ بلوک دیاگرام کلی سیستم طراحی شده را نشان می‌دهد.



شکل ۵-۸: بلوک دیاگرام کلی سیستم طراحی شده

طرح مورد نظر با دقت بیت‌های مختلف روی FPGA، Spartan3 xc3s50-4vq100 آزموده شد. به ازای کوانتیزاسیون Q11.9، دقت طبقه‌بندی روی FPGA برابر دقت حاصل از نرم افزار می‌باشد. این در حالی است که FPGA با فرکانس کاری 64.164 MHz، که ۲۵ برابر کمتر از فرکانس کاری پردازنده کامپیوتر است، عملیات مورد نظر را تقریباً ۱۶۰۰ برابر سریع‌تر از نرم‌افزار انجام داد. این نتایج بیانگر کارایی خوب این ادوات، در کاربردهای با قابلیت موازی کاری بالا، می‌باشد. همچنین منابع سخت‌افزاری استفاده شده برای پیاده‌سازی طرح مورد نظر، استخراج شد. منابع استفاده شده بسیار اندک می‌باشد و اگر قرار باشد طرح مورد نظر، روی یک FPGA بزرگ در کنار سایر بلوک‌های محاسباتی قرار بگیرد، فضای بسیار ناچیزی را اشغال می‌کند.

سخت‌افزار مورد نظر، هم از لحاظ دقت طبقه‌بندی و هم از نظر فرکانس کاری، نتایج بهتری را نسبت به روش آورده شده در بخش ۳-۴ حاصل کرد.

فصل ششم

نتیجه گیری

و

پیشنهاد برای کارهای آینده

در این فصل، مرور کوتاهی بر مطالب ارائه شده در فصل‌های قبل بیان شده است. همچنین پیشنهاداتی برای کارهای بیشتر، ارائه شده است.

۶-۱- خلاصه و جمع‌بندی

در این پایان‌نامه، پس از معرفی مفاهیم مقدماتی لازم در فصول نخست، دسته خاصی از شبکه‌های عصبی فازی به نام شبکه‌های عصبی فازی Min-Max معرفی شده و کارهای انجام شده در این زمینه، مورد مطالعه قرار گرفت.

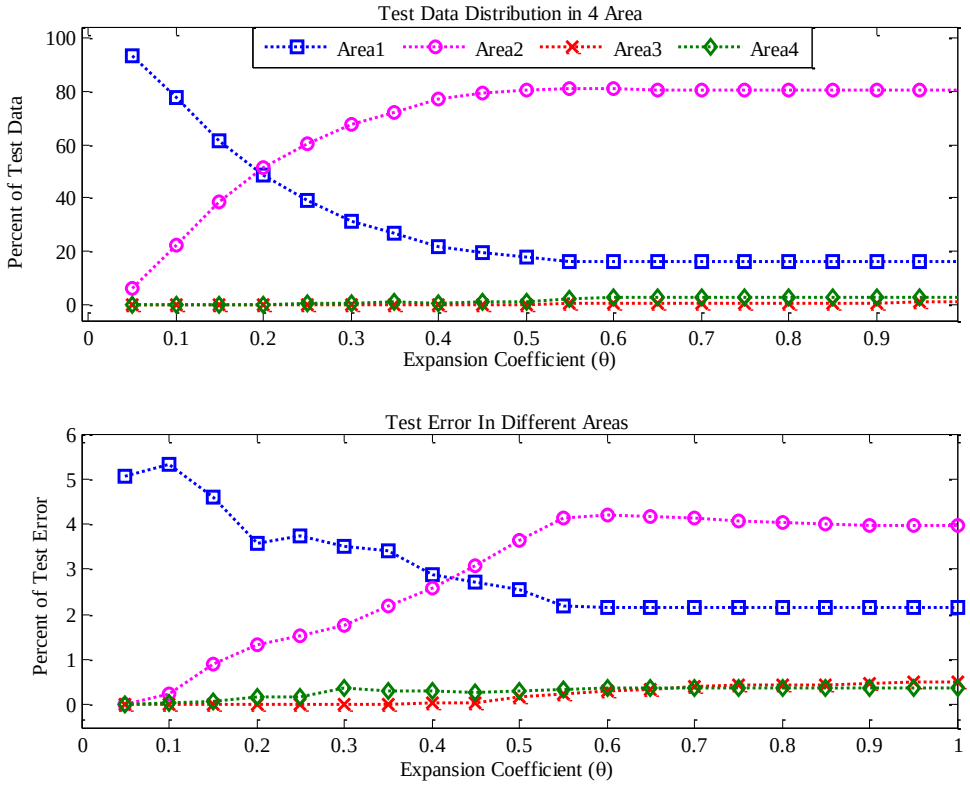
یک شبکه جدید برپایه ابرجعبه‌های فازی Min-Max ارائه شد و کارایی آن با روش‌های قبلی مقایسه شد. ویژگی بارز شبکه ارائه شده این است که در فاز آموزش، ابرجعبه‌های کمتری نسبت به FMCN، تولید می‌کند. شبکه پیشنهادی در فاز تست، روی داده‌های نیم‌دایره، یونوسفر و شیشه نتایج بهتری از بقیه شبکه‌ها حاصل کرد. در فاز آموزش، شبکه پیشنهادی روی داده‌های نیم‌دایره و گل زنبق، نتایج بهتری را نسبت به شبکه‌های دیگر حاصل کرد.

تابع عضویت شبکه پیشنهادی برای پیاده‌سازی روی FPGA، ساده سازی شد و پیاده‌سازی سخت‌افزاری فاز تست شبکه پیشنهادی روی FPGA، برای طبقه‌بندی داده گل زنبق صورت گرفت. از آنجا که ساختار شبکه ساده می‌باشد، نیاز به منابع تعبیه شده مرتفع می‌شود. بنابراین طرح مورد نظر، روی FPGA های معمول و پرکاربرد خانواده Spartan3 پیاده‌سازی شد. طرح سخت‌افزاری برای اجرای فاز تست شبکه، طراحی شد و با دقت بیت‌های مختلف روی FPGA، Spartan3 xc3s50-4vq100 آزموده شد. به ازای کوانتیزاسیون Q11.9، دقت طبقه‌بندی روی FPGA برابر دقت حاصل از نرم افزار حاصل شد. این در حالی است که FPGA با فرکانس کاری 64.164 MHz، که ۰،۰۴ فرکانس کاری کامپیوتر است، عملیات مورد نظر را ۱۶۰۰ برابر سریع‌تر از نرم‌افزار انجام داد.

۶-۲- پیشنهاد برای کارهای آینده

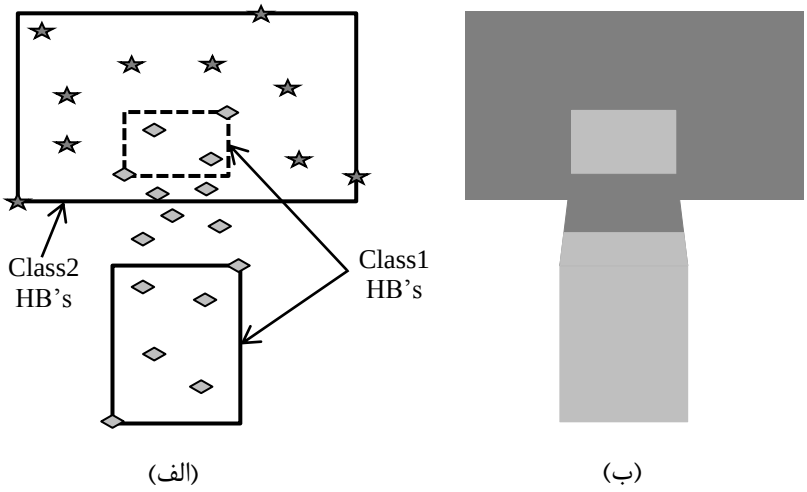
تحقیقات بیشتری روی این موضوع، هم در بخش ساختار شبکه و هم در بخش پیاده‌سازی آن می‌توان انجام داد. در زیر چند پیشنهاد در این زمینه ارائه شده است:

۱) به منظور بهبود شبکه‌های از این نوع، باید بررسی کنیم که شبکه مورد نظر در کدام نواحی فضای ورودی، مرتکب خطاهای زیادی می‌شود. یک مطالعه آماری روی داده گل زنبق نشان داد که در شبکه پیشنهادی، به ازای مقادیر کوچک θ، درصد زیادی از خطای کل، به ازای داده‌های واقع شده در خارج ابرجعبه‌ها صورت می‌گیرد. برای مقادیر بزرگ θ، هنوز بخش قابل توجهی از خطاها در ناحیه خارج ابرجعبه‌ها صورت می‌گیرد و بخش بزرگی از خطای کل نیز مربوط به نواحی داخل ابرجعبه‌های طبقه اول و خارج ابرجعبه‌های طبقه دوم (ناحیه ۲) می‌باشد. چهار ناحیه مربوطه در شکل ۴-۳ در فصل ۴ نشان داده شدند. شکل ۶-۱ توزیع داده‌های تست و خطای تست شبکه پیشنهادی در نواحی چهارگانه برای داده گل زنبق را نشان می‌دهد.



شکل ۶-۱: توزیع داده‌های تست و خطای تست شبکه پیشنهادی در نواحی چهارگانه برای داده گل زنبق

شکل ۶-۲ توضیح بیشتری در این مورد ارائه می‌دهد. در شکل ۶-۲-الف، کلاس ۱ با داده‌های لوزی شکل، دارای دو ابرجعبه یکی در طبقه اول (مستطیل بزرگ) و دیگری در طبقه دوم (مستطیل کوچک خط‌چین) می‌باشد. کلاس ۲ نیز با داده‌های ستاره شکل، دارای یک ابرجعبه می‌باشد. شکل ۶-۲-ب مرزهای تصمیم‌گیری شبکه پیشنهادی را نشان می‌دهد. همانطورکه ملاحظه می‌شود، قسمتی از فضای بین دو ابرجعبه کلاس ۱ اشتباه تشخیص داده شده است.



شکل ۶-۲: نحوه تصمیم‌گیری شبکه پیشنهادی (الف) داده‌ها و ابرجعبه‌های دو کلاس ۱ و۲ (ب) مرزهای تصمیم‌گیری شبکه پیشنهادی

رفع مشکل فوق می‌تواند دقت این طبقه‌بند را افزایش دهد، لذا پیشنهاد می‌شود با یافتن روش مناسبی برای تشخیص این نواحی و استفاده از توابع عضویت مناسب، روی حل مشکل فوق تحقیق شود.

همانطور که در فصل‌های قبل ملاحظه شد، تابع عضویت نقش بسزایی در کارایی شبکه‌های Min-Max ایفا می‌کند. بنابراین برای بهبود این شبکه‌ها، یک راه می‌تواند یافتن تابع عضویت مناسب باشد. با آزمایشی مشابه آزمایش بالا می‌توان ضعف شبکه‌های مختلف را شناسایی کرده و برای بهبود آنها چاره‌جویی کرد.

۲) شبکه ارائه شده را می‌توان برای پردازش ورودی‌های دانه‌ای[1] توسعه داد. داده‌های دانه‌ای به چند طریق می‌توانند نشان داده شوند: ابرجعبه قطعی، ابرجعبه فازی، مجموعه‌های فازی نوع-۲[2] و مجموعه‌های ناهموار[3] [۲۰].

۳) در بخش پیاده‌سازی سخت افزاری شبکه، با توجه به اینکه پروسه آموزش شبکه پیشنهادی ساده می‌باشد، می‌توان فاز آموزش شبکه را نیز روی FPGA پیاده‌سازی نمود. همچنین می‌توان سخت‌افزار شبکه را طوری طراحی نمود که داده‌های با طول بردار ویژگی متفاوت را پشتیبانی کند.

با بهبود روش‌های عصبی فازی Min-Max و پیاده‌سازی سخت‌افزاری این روش‌ها، می‌توان در نهایت یک تراشه طبقه‌بند، که بتواند فاز آموزش و تست محدوده وسیعی از داده‌ها را اجرا کند، طراحی نمود.

1 - Granular
2 - Type-2 Fuzzy Sets
3 - Rough Set

فهرست مراجع

[1] R. Full´er, Neural Fuzzy Systems, Turku: Abo Akademi University, 1995.

[2] D. Nauck, "neuro-fuzzy systems: review and prospects," in *Fifth European Congress on Intelligent Techniques and Soft Computing (EUFIT'97)*, Aachen, Sep. 8-11, 1997.

[3] G. C. a. S. Grossberg, "A massively parallel architecture for a self-organizing neural pattern recognition machine," in *Comput. Vis., Graphics Image Understanding, vol. 37, pp. 54–115*, 1987.

[4] P. K. Simpson, "Fuzzy min-max neural network-Part I: Classification," *IEEE Trans. Neural Netw*, vol. 3, p. 776–786, Sep. 1992.

[5] I. S. J. Misra, "Artificial neural networks in hardware: A survey of two decades of progress," *Neurocomputing*, vol. 74, p. 239–255, 2010.

[6] "System Generator for DSP User Guide," july 6, 2011.

[7] L. A. Zadeh, "Fuzzy sets," *Inf. Control*, vol. 8, p. 338–353, 1965.

[۸] لی وانگ، ترجمه محمد تشنه لب، سیستم‌های فازی و کنترل فازی، دانشگاه صنعتی خواجه نصیرالدین طوسی

[9] Leszek Rutkowski, Krzysztof Cpalka, "Flexible Neuro-Fuzzy Systems," *IEEE TRANSACTIONS ON NEURAL NETWORKS*, Vols. 14, NO. 3, MAY 2003.

[10] A. Abraham, "Adaptation of Fuzzy Inference System Using Neural Learning," in *Fuzzy Systems Engineering: Theory and Practice*, New York, Spr.-Verlag, 2005, pp. 53-83.

[11] J.-S. R. Jang, "ANFIS : Adaptive-Network-Based Fuzzy Inference System," *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS*, vol. 23, May/June 1993.

[12] Detlef Nauck, Ulrike Nauck, and Rudolf Kruse, "Generating Classification Rules with the neuro-fuzzy system NEFCLASS," in *Proc. Biennial Conference of the North American fuzzy Information Processing Society NAFIPS'96*, Berkeley, June 1996.

[13] F. M. F. Mascioli and G. Martinelli, "A constructive approach to neuro-fuzzy networks," *Signal Process*, Vols. 64, no. 3, p. 347–358, Feb.1998.

[14] K. Simpson, "Fuzzy min-max neural network—Part II: Clustering," *IEEE Trans. Fuzzy Syst*, Vols. 1, no. 1, p. 32–45, Feb. 1993.

[15] B. Gabrys and A. Bargiela, "General fuzzy min-max neural network for clustering and classification," *IEEE Trans. Neural Netw*, Vols. 11, no. 3, p. 769–783, May 2000.

[16] A. V. Nandedkar, and P. K. Biswas, "A Fuzzy Min-Max Neural Network Classifier With Compensatory Neuron Architecture," *IEEE trans. Neural netw*, Vols. 18, no. 1, january. 2007.

[17] S. Abe and M. S. Lan, "A method for fuzzy rules extraction directly from numerical data and its application to pattern classification," *IEEE Trans. Fuzzy Syst*, Vols. 3, no. 1, p. 18–28, Feb. 1995.

[18] M. Meneganti, F. S. Saviello, and R. Tagliaferri, "Fuzzy neural networks for classification and detection of anomalies," *IEEE Trans. Neural Netw*, Vols. 9, no. 5, p. 848–861, Sep. 1998.

[19] A. Rizzi, M. Panella, and F. M. F. Mascioli, "Adaptive resolution min-max classifiers," *IEEE Trans. Neural Netw*, Vols. 13, no. 2, p. 402–414, Mar. 2002.

[20] Abhijeet V. Nandedkar and Prabir K. Biswas, "A Granular Reflex Fuzzy Min–Max Neural Network for Classification," *IEEE TRANSACTIONS ON NEURAL NETWORKS*, Vols. 20, NO. 7, JULY 2009

[21] Syed Manzoor Qasim, Shuja Ahmad Abbasi and Bandar Almashary, "A Review of FPGA-Based Design Methodology and Optimization Techniques for Efficient Hardware Realization of Computation Intensive Algorithms, " *2009 IEEE*

[22] Y. Liang, S. Q. Fan, and D. M. Jin, "The Hardware Implementation of A Multi-resolution Combined Fuzzy Min-Max Classifier Chip," in *First International Conference on Innovative Computing, Information and Control (ICICIC'06)*, 2006.

[23] A.V. Nandedkar and P.K. Biswas, "Object Recognition Using Reflex Fuzzy Min-Max Neural Network with Floating Neurons," in *ICVGIP 2006, LNCS 4338, pp. 597–609*, Springer-Verlag Berlin Heidelberg, 2006.

[24] "http://www.ics.uci.edu/~mlearn/databases," [Online].

واژه‌نامه فارسی به انگلیسی

ابرجعبه	Hyperbox
اتصال به‌روز شده بر ثانیه	Connection-Updates-Per-Second
اتصالات بر ثانیه	Connections-Per-Second
اتصالات ثابت	Hardwired
ادوات شارژ خازنی	Charge Coupled Devices
ارزیابی محافظت شده	Guarded Evaluation
استلزام فازی	Fuzzy Implication
استلزام ممدانی	Mamdani Implication
استلزام منطقی	Logical Implication
استنتاج	Inference
اسلایس	Slice
اشباع	Saturation
اصل توسعه زاده	Zade’s Expansion Principle
الگوریتم آموزشی	Training Algorithm
الگوریتم کمترین مربعات	Least Square Method
الگوریتم‌های ژنتیک	Genetic Algorithms
انتساب‌های غلط	Misclassification
انرژی سیناپسی	Synaptic Energy
آرایه گیت برنامه‌پذیر در محل	Field Programmable Gate Array
با نظارت	Supervised
بازدارنده	Inhibition
بازده	Throughput
بازشناسی کاراکتر نوری	Optical Character Recognition
بازگشتی	Recursive
بدون نظارت	Unsupervised
برچسب	Label
برخط	Online
برش درخت کلاک	Clock Tree Paring
برون‌خط	Offline
برون‌دهی	Fan-Out
بلادرنگ	Real Time
بلوک‌های منطقی برنامه‌پذیر	Configurable Logic Block
بیش یادگیری	Over Fitting
پارامتر کنترل فازی کنندگی	Fuzziness Control Parameter
پایگاه داده‌های	Data Base
پایگاه قواعد فازی	Fuzzy Rule Base
پس انتشار خطا	Back Propagation
پسخور	Backward
پیاده‌سازی	Implementation
پیشخور	Forward
تابع فعال‌سازی	Activation Function
تاکاشی-سوگنو	Takagi-Sugeno
تالی	Consequent
تأخیر	Latency
تجمیع	Aggregation
تراشه‌عصبی	Neurochip
تراشه‌عصبی خاص منظوره	Special-Purpose Neurochip
تراشه‌عصبی همه منظوره	General-Purpose Neurochip
ترکیبی	Hybrid
تعمیم	Generalization
تماماً متصل	Fully Connected
تنزل مطبوع	Graceful Degradation
جانمایی	Placement
جدول جستجو	Look-Up Table
جزئی	Partial
چیپ در حلقه	Chip-In-The-Loop
حافظه‌های انجمنی فازی	Fuzzy Associative Memories
خارج- چیپ	Off-Chip
خط لوله	Pipeline

خوشه‌یابی	Clustering
دانه‌ای	Granular
درجه خودکار بودن	Automation Degree
رباتیک خودمختار	Autonomous Robotics
رشته‌بیت	Bit Stream
رم بلوکی	Block RAM
رمزگذاری ماشین حالت محدود	Finite State Machine Encoding
رمزگشایی	Decode
روی- چیپ	On-Chip
ریست	Reset
سازنده	Constructive
سنتز منطقی	Logical Synthesis
سیستم استنتاج عصبی- فازی تطبیقی	Adaptive Neuro-Fuzzy Inference System
سیستم‌های استنتاج عصبی-فازی	Neuro-Fuzzy Inference Systems
سیستم‌های استنتاج فازی	Fuzzy Inference Systems
شبکه عصبی سخت‌افزاری	Hardware Neural Network
شبکه عصبی فازی Min-Max	Fuzzy Min-Max Neural Network
شبکه عصبی فازی منظم	Regular Fuzzy Neural Network
شبکه عصبی فازی مین-ماکس با نرون‌های جبران‌ساز	Fuzzy Min-Max Neural Network with Compensatory Neurons
شبکه عصبی فازی مین-ماکس عمومی	General Fuzzy Min-Max Neural Network
شبکه عصبی فازی ترکیبی	Hybrid Fuzzy Neural Network
شبکه عصبی منظم	Regular Neural Network
شبکه عصبی ترکیبی	Hybrid Neural Network
شبکه‌های عصبی فازی	Fuzzy Neural Network
شبکه‌های عصبی مصنوعی	Artifical Neural Networks
شبیه‌سازی رفتاری	Behavioral Simulation
شراب	Wine

شمول	Containment
شمول جزئی	Partial Containment
شمول کامل	Full Containment
شیشه	Glass
ضریب انبساط	Expansion Coefficient
طبقه‌بند	Classifier
طبقه‌بند بادقت تطبیقی	Adaptive Resolution Classifier
طبقه‌بند عصبی فازی مین-ماکس چند-دقته ترکیبی	Multi-resolution Combined Fuzzy Min-Max Classifier
طبقه‌بندی	Classification
طبقه‌بندی عصبی فازی	Neuro-Fuzzy Classification
طراحی آسنکرون	Asynchronous Design
عمومی	Generic
غیرفازی‌ساز	Defuzzifier
فازی‌ساز	Fuzzifier
فعال‌سازی	Activation
قابل پیکربندی مجدد	Reconfigurable
قابلیت اطمینان	Reliability
قابلیت کاسکد شدن	Cascadability
قابلیت مسیریابی	Routability
قدرت آتش	Firing Strong
قطع کلاک	Clock Gating
قطعه‌بندی تصویر	Image Segmentation
قطعی	Crisp
قواعد	Rules
قواعد اگر-آنگاه فازی	Fuzzy If-Then Rules
کارایی	Performance
کامپیوتر میزبان	Host Computer
کد گری	Gray Coding
کلیدزنی	Switching
کلین-دینس	Kleene-Dienes
کوانتیزاسیون	Quantization

گرادیان نزولی	Gradient Descent
گل زنبق	Iris
لچ‌های شفاف	Transparent Latches
متغیر زبانی	Linguistic Variable
مجموعه فازی	Fuzzy Set
مجموعه‌های فازی نوع-۲	Type-۲ Fuzzy Sets
مجموعه‌های ناهموار	Rough Set
محاسبات نرم	Soft Computing
مدار مجتمع با کاربرد خاص	Application Specific Integrated Circuit
مدل طبقه‌بندی	Classification Model
مساله طبقه‌بندی	Classification Problem
مسیریابی	Routing
مشارکتی	Cooperative
مقدار زبانی	Linguistic Value
مقدم	Antecedent
ممدانی	Mamdani
ممیز ثابت	Fixed Point
ممیز شناور	Floating Point
منطق فازی	Fuzzy Logic
منطقی	Logical
موتور ضرب-انباره	Multiply-Accumulate Engine
مولتی پلکس تقسیم زمانی	Time-Division Multiplexing
نتلیست	Netlist
نرون فازی	Fuzzy Neuron
نرون‌های جبران شمول	Containment Compensation Neurons
نرون‌های جبران همپوشانی	Overlap Compensation Neurons
نرون‌های طبقه‌بند	Classifying Neurons
نقشه‌های خود سازمانده	Self Organizing Maps
نگاشت تکنولوژی	Technology Mapping
همزمان	Concurrent
وات بر ثانیه برای هر اتصال	Watt Per Connection-Per-Second
یکبارگذر	One Pass Through
یکپارچه	Integrated
یونوسفر	Ionosphere

واژه‌نامه انگلیسی به فارسی

Activation	فعال‌سازی
Activation Function	تابع فعال‌سازی
Adaptive Neuro-Fuzzy Inference System	سیستم استنتاج عصبی-فازی تطبیقی
Adaptive Resolution Classifier	طبقه‌بند بادقت تطبیقی
Aggregation	تجمیع
Antecedent	مقدم
Application Specific Integrated Circuit	مدار مجتمع با کاربرد خاص
Artifical Neural Networks	شبکه‌های عصبی مصنوعی
Asynchronous Design	طراحی آسنکرون
Automation Degree	درجه خودکار بودن
Autonomous Robotics	رباتیک خودمختار
Back Propagation	پس انتشار خطا
Backward	پسخور
Behavioral Simulation	شبیه‌سازی رفتاری
Bit Stream	رشته‌بیت
Block RAM	رم بلوکی
Cascadability	قابلیت کاسکد شدن
Charge Coupled Devices	ادوات شارژ خازنی
Chip-In-The-Loop	چیپ در حلقه
Classification	طبقه‌بندی
Classification Model	مدل طبقه‌بندی
Classification Problem	مساله طبقه‌بندی
Classifier	طبقه‌بند
Classifying Neurons	نرون‌های طبقه‌بند
Clock Gating	قطع کلاک
Clock Tree Paring	برش درخت کلاک
Clustering	خوشه‌یابی
Concurrent	همزمان
Configurable Logic Block	بلوک‌های منطقی برنامه‌پذیر
Connections-Per-Second	اتصالات بر ثانیه
Connection-Updates-Per-Second	اتصال به‌روز شده بر ثانیه
Consequent	تالی
Constructive	سازنده
Containment	شمول
Containment Compensation Neurons	نرون‌های جبران شمول
Cooperative	مشارکتی
Crisp	قطعی
Data Base	پایگاه داده‌های
Decode	رمزگشایی
Defuzzifier	غیرفازی‌ساز
Expansion Coefficient	ضریب انبساط
Fan-Out	برون‌دهی
Field Programmable Gate Array	آرایه گیت برنامه‌پذیر در محل
Finite State Machine Encoding	رمزگذاری ماشین حالت محدود
Firing Strong	قدرت آتش
Fixed Point	ممیز ثابت
Floating Point	ممیز شناور
Forward	پیشخور
Full Containment	شمول کامل
Fully Connected	تماماً متصل
Fuzzifier	فازی‌ساز
Fuzziness Control Parameter	پارامتر کنترل فازی کنندگی
Fuzzy Associative Memories	حافظه‌های انجمنی فازی
Fuzzy If-Then Rules	قواعد اگر-آنگاه فازی
Fuzzy Implication	استلزام فازی
Fuzzy Inference Systems	سیستم‌های استنتاج فازی
Fuzzy Logic	منطق فازی
Fuzzy Min-Max Neural Network	شبکه عصبی فازی Min-Max

Least Square Method	الگوریتم کمترین مربعات
Linguistic Value	مقدار زبانی
Linguistic Variable	متغیر زبانی
Logical	منطقی
Logical Implication	استلزام منطقی
Logical Synthesis	سنتز منطقی
Look-Up Table	جدول جستجو
Mamdani	ممدانی
Mamdani Implication	استلزام ممدانی
Misclassification	انتساب‌های غلط
Multiply-Accumulate Engine	موتور ضرب-انباره
Multi-resolution Combined Fuzzy Min-Max Classifier	طبقه‌بند عصبی فازی مین-ماکس چند-دقته ترکیبی
Netlist	نتلیست
Neurochip	تراشه‌عصبی
Neuro-Fuzzy Classification	طبقه‌بندی عصبی فازی
Neuro-Fuzzy Inference Systems	سیستم‌های استنتاج عصبی-فازی
Off-Chip	خارج- چیپ
Offline	برون‌خط
On-Chip	روی- چیپ
One Pass Through	یکبارگذر
Online	برخط
Optical Character Recognition	بازشناسی کاراکتر نوری
Over Fitting	بیش یادگیری
Overlap Compensation Neurons	نرون‌های جبران همپوشانی
Partial	جزئی
Partial Containment	شمول جزئی
Performance	کارایی
Pipeline	خط لوله
Placement	جانمایی
Quantization	کوانتیزاسیون
Real Time	بلادرنگ
Reconfigurable	قابل پیکربندی مجدد
Fuzzy Min-Max Neural Network with Compensatory Neurons	شبکه عصبی فازی مین-ماکس با نرون‌های جبران‌ساز
Fuzzy Neural Network	شبکه‌های عصبی فازی
Fuzzy Neuron	نرون فازی
Fuzzy Rule Base	پایگاه قواعد فازی
Fuzzy Set	مجموعه فازی
General Fuzzy Min-Max Neural Network	شبکه عصبی فازی مین-ماکس عمومی
Generalization	تعمیم
General-Purpose Neurochip	تراشه‌عصبی همه منظوره
Generic	عمومی
Genetic Algorithms	الگوریتم‌های ژنتیک
Glass	شیشه
Graceful Degradation	تنزل مطبوع
Gradient Descent	گرادیان نزولی
Granular	دانه‌ای
Gray Coding	کد گری
Guarded Evaluation	ارزیابی محافظت شده
Hardware Neural Network	شبکه عصبی سخت‌افزاری
Hardwired	اتصالات ثابت
Host Computer	کامپیوتر میزبان
Hybrid	ترکیبی
Hybrid Fuzzy Neural Network	شبکه عصبی فازی ترکیبی
Hybrid Neural Network	شبکه عصبی ترکیبی
Hyperbox	ابرجعبه
Image Segmentation	قطعه‌بندی تصویر
Implementation	پیاده‌سازی
Inference	استنتاج
Inhibition	بازدارنده
Integrated	یکپارچه
Ionosphere	یونوسفر
Iris	گل زنبق
Kleene-Dienes	کلین-دینس
Label	برچسب
Latency	تأخیر

Antecedent	مقدم
Application Specific Integrated Circuit	مدار مجتمع با کاربرد خاص
Artifical Neural Networks	شبکه‌های عصبی مصنوعی
Asynchronous Design	طراحی آسنکرون
Automation Degree	درجه خودکار بودن
Autonomous Robotics	رباتیک خودمختار
Back Propagation	پس انتشار خطا
Backward	پسخور
Behavioral Simulation	شبیه‌سازی رفتاری
Bit Stream	رشته‌بیت
Block RAM	رم بلوکی
Cascadability	قابلیت کاسکد شدن
Charge Coupled Devices	ادوات شارژ خازنی
Chip-In-The-Loop	چیپ در حلقه
Classification	طبقه‌بندی
Classification Model	مدل طبقه‌بندی
Classification Problem	مساله طبقه‌بندی
Recursive	بازگشتی
Regular Fuzzy Neural Network	شبکه عصبی فازی منظم
Regular Neural Network	شبکه عصبی منظم
Reliability	قابلیت اطمینان
Reset	ریست
Rough Set	مجموعه‌های ناهموار
Routability	قابلیت مسیریابی
Routing	مسیریابی
Rules	قواعد
Saturation	اشباع
Self Organizing Maps	نقشه‌های خود سازمانده
Slice	اسلایس
Soft Computing	محاسبات نرم
Special-Purpose Neurochip	تراشه‌عصبی خاص منظوره
Supervised	با نظارت
Switching	کلیدزنی
Synaptic Energy	انرژی سیناپسی
Takagi-Sugeno	تاکاشی-سوگنو
Technology Mapping	نگاشت تکنولوژی
Throughput	بازده
Time-Division Multiplexing	مولتی پلکس تقسیم زمانی
Training Algorithm	الگوریتم آموزشی
Transparent Latches	لچ‌های شفاف
Type-۲ Fuzzy Sets	مجموعه‌های فازی نوع-۲
Unsupervised	بدون نظارت
Watt Per Connection-Per-Second	وات بر ثانیه برای هر اتصال
Wine	شراب
Zade’s Expansion Principle	اصل توسعه زاده
Activation	فعال‌سازی
Activation Function	تابع فعال‌سازی
Adaptive Neuro-Fuzzy Inference System	سیستم استنتاج عصبی-فازی تطبیقی
Adaptive Resolution Classifier	طبقه‌بند بادقت تطبیقی
Aggregation	تجمیع

Abstract: Fuzzy min-max neural networks, are relatively new techniques in classification and clustering applications. Special feature of these networks, one-pass learning algorithm, make them ideal for hardware implementation and real time applications.

A new algorithm based on fuzzy min-max hyper-boxes is presented. Such as FMNN, this network uses aggregation of fuzzy hyper-boxes to define class boundaries, but the overlap test and contraction process has been removed from standard learning algorithm of FMNN, making our method simpler and faster.

The proposed network uses two sets of hyper-boxes with different expansion coefficients, so the network can use smaller hyper-boxes in the class boundaries. Computer simulations show a good performance of the network. In most cases, the proposed method produces better results than previous methods with lower number of neurons.

In this thesis, a hardware implementation of the proposed network is also realized on FPGA. The membership function of hyper-boxes is simplified, so that the resulting layout does not use multipliers. Furthermore pipeline technique and parallel computing is constructed to increase the speed of computation.

Key words: Classification, Fuzzy Neural Network, Fuzzy Min-Max Hyper-box, FPGA.

Shahrood University of Technology

FPGA Implementation of Fuzzy Neural Networks for Classification Applications

Thesis

Submitted in Partial Fulfillment of the
Requirements for the Degree of Master of Science (M.Sc.)
in Electrical Engineering, Digital Electronics

Department of Electrical Engineering and Robotics
Shahrood University of Technology

By:

Farid Farajzadeh

Supervisor:

Dr. Ali Soleimani

Advisor:

Dr. Hossein Khosravi

Feb. 2013