پژوهش‌های نوین در تصمیم‌گیری
دوره 4، شماره 3، پاییز 1398

توسعه شبکه عصبی تصمیم مبتنی بر الگوریتم ژنتیک برای ارزیابی ارجحیات در مسائل تصمیم‌گیری چندهدفه

محدثه نادرشاهی[1*]، اعظم دخت صفی صمغ آبادی[2]، رضا توکلی مقدم[3]

1- مربی، گروه مهندسی صنایع، دانشگاه پیام نور، تهران، ایران
2- استادیار، گروه مهندسی صنایع، دانشگاه پیام نور، تهران، ایران
3- استاد، دانشکده مهندسی صنایع، پردیس دانشکده های فنی، دانشگاه تهران، تهران، ایران

دریافت: 98/1/28 پذیرش: 98/5/26

چکیده

بکارگیری شبکه‌های عصبی در تخمین و توصیف ساختار ارجحیت‌های تصمیم‌گیرنده، در حل مسائل تصمیم‌گیری چندهدفه در سال‌های اخیر بسیار مورد توجه قرار گرفته است. شبکه عصبی تصمیم رویکردی نوین برای تخمین تابع مطوبیت تصمیم‌گیرنده در مسایل چندهدفه است. توسعه و بهبود روش‌های آموزش این نوع از شبکه‌ها، یافتن راه حل مرجح در مسایل چندهدفه، به خصوص مسایل با ابعاد بزرگ را تسهیل می‌نماید. در این مقاله، به منظور غلبه بر مشکلات روش‌های آموزشی مبتنی بر گرادیان و با هدف افزایش کارآیی شبکه عصبی تصمیم روش آموزشی آن توسعه داده شده است و از الگوریتم ژنتیک برای آموزش این شبکه عصبی استفاده می‌شود. برای تنظیم پارامترهای شبکه عصبی تابع هزینه بهبود یافته‌ای پیشنهادی می‌شود و بر اساس این تابع هزینه پارامترهای شبکه عصبی بهینه‌سازی می‌شوند. رویکرد پیشنهادی در حل چندین مثال کاربردی بکارگرفته شده‌است که نتایج نشان می‌دهند که رویکرد پیشنهادی روشی کارآ به منظور تخمین تابع مطلوبیت –به‌خصوص غیرخطی- در حل مسائل تصمیم‌گیری چندهدفه می‌باشد. همچنین رویکرد پیشنهادی در تخمین توابع مطلوبیت مسائل چندهدفه گسسته نیز قابلیت بکارگیری دارد.

کلمات کلیدی: آموزش شبکه عصبی تصمیم، الگوریتم ژنتیک، مسائل تصمیم گیری چندهدفه، تابع مطلوبیت.

---

*نویسنده مسئول مقاله: E-mail: ????????

1- مقدمه

نحوه آموزش یک شبکه عصبی در کارآیی آن تأثیر بسیار زیادی دارد. از این رو روش‌های مختلفی در این زمینه ارائه و استفاده شده‌اند. در آموزش یک شبکه عصبی، پارامترهای شبکه به گونه‌ای تعیین می‌شوند که شبکه، قدرت تعمیم قابل‌قبولی داشته باشد. توانایی الگوریتم‌های تکاملی در حل مسائل بهینه‌سازی سبب ترغیب پژوهش‌گران به استفاده از این دسته از الگوریتم‌ها برای طراحی و آموزش شبکه‌های عصبی شده است [1]. الگوریتم‌های تکاملی از روش‌های جستجوی تصادفی مبتنی بر جمعیت محسوب می‌شوند [2]. اساس این الگوریتم‌ها که از تکامل طبیعی موجودات زنده الهام گرفته است، انتخاب متناسب با برازندگی افراد جمعیت می‌باشد. در یک الگوریتم تکاملی، جمعیتی از راه‌حل های نامزد با استفاده از عملگرهای ژنتیکی و در طی نسل‌های متوالی تکامل می‌یابند، در شرایطی که احتمال گیرافتادن در دام نقاط بهینه محلی پایین است.

در تحلیل مسائل تصمیم‌گیری چندهدفه[1] (MODM) به دلیل وجود توابع چندگانه متناقض، اغلب مطلوب است که به طور صریح ساختار کامل ارجحیت‌های تصمیم‌گیرنده[2] (DM) از طریق یک مدل تصمیم تجویزی مانند تابع مطلوبیت[3] بدست آید. در این صورت، ادامه فرآیند حل در مسائل MODM بسیار ساده‌تر می‌شود. اما ارزیابی ساختار ارجحیت‌های DM از طریق تابع مطلوبیت به سادگی میسر نمی‌باشد و در بسیاری از مسایل دنیای واقعی ملزوم به برقراری مفروضاتی می‌باشد که کاربرد این روش‌ها را محدود ساخته است. علاوه بر این از نقطه نظر عملی نیز، روش هایی که از اطلاعات اولیه استفاده می‌کنند، بار زیادی بر دوش DM از جهات مختلف وارد می‌سازند [3].

از آنجاکه ساختار ارجحیت‌های تصمیم‌گیرنده در عمل ممکن است بسیار پیچیده باشند، روش‌ها و ابزارهای توانمندی که قادر به اخذ این ساختارها و هدایت فرآیند حل در یافتن ارجح‌ترین جواب باشند، نیاز می باشد [4]. از طرفی در سال‌های اخیر محققین توجه زیادی به کاربرد شبکه‌های عصبی مصنوعی در مسائل مختلف داشته‌اند. شبکه عصبی ابزار مدل‌سازی داده قوی هستند که قادر به تخمین و نمایش رابطه‌های پیچیده ورودی و خروجی می‌باشند [5]. یک نوع شبکه عصبی مصنوعی پیشرو[4] (FFANN) که بسیار مورد استفاده قرار گرفته است، شبکه‌های عصبی پرسپترون چند لایه (MLP) می‌باشد. شبکه‌عصبی پرسپترون می‌تواند یک لایه یا چند لایه باشد. در یک شبکه عصبی چند لایه پرسپترون[5] (MLP) گره‌های

1. Multi-Objective Decision Making
2. Decision Maker
3. Utility Function
4. Feed Forward Artificial Neural Network
5. Multiple Layer Perceptron

عصبی در یک MLP در لایه‌های سلسله مراتبی چندگانه ورودی، لایه (های) پنهان و لایه خروجی سازماندهی شده‌اند. با استفاده از یک الگوریتم آموزشی، MLP قادر به نمایش روابط پیچیده ورودی و خروجی می‌باشد. ثابت شده‌است که هر تابع پیوسته با یک شبکه پرسپترون سه لایه قابل برآورد و تخمین است [6]. این خصوصیت، شبکه‌عصبی را ابزاری قوی و نویدبخش جهت توصیف تابع مطلوبیت ساخته‌است. در اغلب رویکردهای مبتنی بر شبکه‌های عصبی در این حوزه از یک FFANN استفاده شده است. چن و لین [7] یک شبکه عصبی به نام شبکه عصبی تصمیم[1] (DNN) پیشنهاد کردند که ساختار ویژه این شبکه عصبی، محدودیت روش‌های مبتنی بر شبکه‌های عصبی در تخمین تابع مطلوبیت را مرتفع ساخته‌است. این شبکه از تکنیک‌های ارزیابی غیرمستقیم ارجحیت‌ها در آموزش شبکه عصبی بهره گرفته است، بنابراین ظرفیت یادگیری شبکه افزایش یافته است. علاوه بر این مزیت دیگر آن است که در این رویکرد حجم مجموعه داده آموزشی با بکارگیری روش‌های ارزیابی غیردقیق افزایش یافته، لذا شرایط برای تصمیم‌گیرندگان تسهیل شده است [8].

همانطور که درابتدای این بخش ذکر شد، نحوه آموزش یک شبکه عصبی در کارآیی آن تأثیر بسیار زیادی دارد. یکی از متداول‌ترین روش‌ها برای آموزش شبکه‌های عصبی، الگوریتم‌های مبتنی بر گرادیان است. الگوریتم پس انتشار خطا یکی از نمونه‌های پر کاربرد این نوع الگوریتم ها است که در نسخه‌های مختلفی استفاده شده است [9]. یکی از معایب روش‌های مبتنی بر گرادیان، افتادن در دام نقاط بهینه محلی تابع خطا است [10]. همچنین اگر تابع خطا چندبعدی بوده یا مشتق‌پذیر نباشد، روش پس انتشار خطا، نقطه بهینه عمومی را نمی‌تواند پیدا کند. از طرفی استفاده از الگوریتم‌های تکاملی برای یافتن مجموعه پارامترهای شبکه‌های عصبی درواقع راهی برای غلبه بر مشکلات روش‌های آموزش مبتنی بر گرادیان است. همچنین، این الگوریتم‌ها فضای جواب را بدون نیاز به اطلاعات گرادیان جستجو می‌کنند [10] و از این‌رو بسیار سودمند و قابل استفاده هستند [11].

از آنجا که الگوریتم آموزشی موجود برای DNN نیز مبتنی بر گرادیان است، در این مقاله با هدف توسعه DNN به‌منظور تخمین توابع مطلوبیت خطی و به‌خصوص غیرخطی و پیچیده در مسایل چند هدفه، روش آموزشی مبتنی بر الگوریتم ژنتیک ارایه شده است.

در بخش بعدی مقاله ابتدا به مرور آموزش شبکه‌های عصبی با الگوریتم‌های تکاملی پرداخته شده‌است و بخش 3، به معرفی شبکه عصبی پیشنهادی مبتنی بر الگوریتم ژنتیک به منظور تخمین توابع مطلوبیت در مسائل MODM اختصاص یافته‌است. در بخش4 روش پیشنهادی برای آموزش DNN بر مبنای الگوریتم ژنتیک شرح داده شده‌است. در آخر

1. Decision Neural Network

کارایی روش پیشنهادی در حل مسئله تصمیم گیری چند هدفه با توابع مطلوبیت خطی و غیرخطی ارزیابی می‌شود.

2- آموزش شبکه‌های عصبی با الگوریتم‌های تکاملی

آموزش یک شبکه عصبی معمولاً بر اساس کمینه‌سازی یک تابع خطا مانند میانگین مربعات خطا بین مقدار مورد انتظار و خروجی واقعی شبکه به ازای همه نمونه‌های آموزشی صورت می‌گیرد. اغلب الگوریتم‌های آموزشی مانند پس انتشار خطا برمبنای نزول گرادیان است. روش پس انتشار خطا در موارد متعددی با موفقیت به‌کاربرده شده است [13-12] و [9]، اما این روش به علت استفاده از اطلاعات گرادیان دارای معایب مختلفی است [14] و [10]. همانطور که در بخش مقدمه نیز ذکر شد، افتادن در دام یک نقطه کمینه محلی تابع خطا یکی از معایب این روش است. همچنین اگر تابع خطا چندبعدی یا مشتق ناپذیر نباشد، روش پس انتشار خطا نمی‌تواند کمینه عمومی را پیدا کند. این در حالی است که الگوریتم‌های تکاملی به‌صورت عمومی جستجو می‌کنند و بدون نیاز به اطلاعات گرادیان، مجموعه جواب های نزدیک بهینه را پیدا می‌کند. در کاربردهای متعددی برای آموزش شبکه عصبی از الگوریتم های تکاملی استفاده شده است [15] و [10].

مقایسه‌های متعددی بین آموزش مبتنی بر گرادیان و الگوریتم‌های تکاملی صورت گرفته‌است. بعضی از آن‌ها الگوریتم‌های تکاملی را سریع‌تر دانسته‌اند [18-16] و در برخی از مقالات روش پس انتشار خطا را سریع تر و کارآتر گزارش کرده‌اند [21-19]. بر این اساس می توان نتیجه گرفت که کارایی و سرعت این الگوریتم‌ها به کاربرد مورد نظر بستگی دارد. همچنین اختلاف نتایج گزارش شده می‌تواند به دلیل متفاوت بودن الگوریتم‌های استفاده شده باشد. به عنوان مثال ممکن است است در یک آزمایش از یک الگوریتم پس انتشار سریع با یک نسخه کند الگوریتم تکاملی مقایسه شده باشد.

به‌طور کلی الگوریتم‌های ارائه شده برای تکامل شبکه‌های عصبی را می توان به سه دسته تقسیم‌بندی کرد. دسته نخست که روش پیشنهادی این مقاله نیز جز این دسته محسوب می‌شود، به روش‌هایی اختصاص دارد که در آن‌ها معماری شبکه از قبل مشخص بوده و فقط اوزان اتصالات شبکه برای تخمین خروجی شبکه بهینه می‌شوند. در این راستا می‌توان به استفاده از الگوریتم ژنتیک برای یافتن مجموعه اوزان بهینه اشاره کرد. نمایش وزن‌ها به صورت باینری و به‌کارگیری الگوریتم ژنتیک استاندارد از قدیمی‌ترین کارهای این دسته محسوب می‌شوند [15]. از جمله مزیت‌های نمایش باینری، سادگی و عمومیت آن است. بدون نیاز به طراحی عمل‌گرهای پیچیده می‌توان از عمل‌گرهای ساده ادغام و جهش

بیتی استفاده کرد. از طرفی یکی از مسائل اصلی آموزش شبکه‌های عصبی با الگ.وریتم‌های تکاملی، مشکل جایگشت می‌باشد [23-22]. این مسأله از چند به یک بودن نگاشت ژنوتایپ به فنوتایپ به‌وجود می‌آید. به این معنی که ترتیب‌های عصب های یک شبکه می‌تواند به شبکه‌های مختلف با عملکرد یکسان منجر شود. بعضی نیز از بردار اعداد حقیقی برای بازنمایی اوزان استفاده کرده‌اند [24]. در این تحقیقات رهیافت آموزش به وسیله تکامل بسیار سریع‌تر از روش پس انتشار خطا مسأله مورد نظر را حل کرده‌است. استراتژی تکاملی و برنامه نویسی تکاملی از معمول‌ترین روش‌ها برای تکامل بردارهای حقیقی محسوب می‌شود [1]. در سال‌های اخیر همچنین الگوریتم‌های مختلفی برای تکامل بردارهایی از اعداد حقیقی ارائه شده‌است [25] که در زمینه تکامل وزن‌های شبکه‌های عصبی نیز مورد استفاده قرار گرفته‌اند [26].

دسته دوم الگوریتم‌های تکاملی ارائه شده در این حوزه، به روش‌هایی اختصاص دارد که ساختار معماری شبکه را تعیین می کنند. طرح معماری در کارایی و قدرت پردازش شبکه تأثیر مستقیم دارد. اگر تعداد نرون‌های شبکه و ارتباطات بسیار کم باشد، ممکن است شبکه هیچ‌گاه همگرا نشود. از طرفی، اگر پیچیدگی شبکه عصبی بسیار زیاد باشد، امکان یادگیری بیش از حد[1] (بیش برازش) آن وجود دارد [26]. چنانچه معماری شبکه توسط فرد خبره طراحی شود، گریزی از سعی و خطا نخواهد بود.

در دسته سوم نیز طراحی معماری شبکه و بهینه سازی پارامترهای آن همزمان صورت می‌گیرد. با توجه به توانایی‌های الگوریتم‌های تکاملی برای آموزش شبکه‌های عصبی، لزوم طراحی معماری که به طور معمول توسط فرد خبره و با سعی و خطا انجام می‌شود، بهترین رویکرد می‌باشد.

3- شبکه عصبی تصمیم مبتنی بر الگوریتم ژنتیک[2] به منظور تخمین توابع مطلوبیت

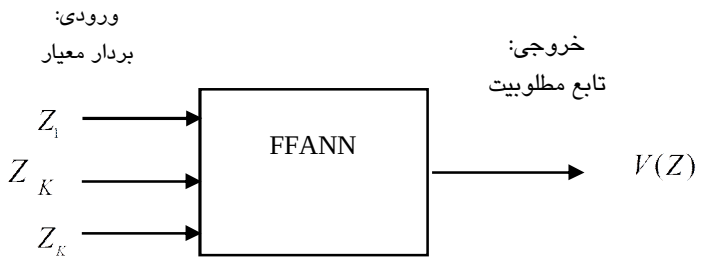
ایده اصلی استفاده از شبکه‌های عصبی برای حل مسائل MODM از آنجا ناشی می‌شود که ارجحیت‌های DM دارای توابعی ضمنی است، تعیین این توابع کار دشواری است. از طرفی یک شبکه عصبی پیشرو سه لایه (لایه‌های ورودی، پنهان و خروجی) با تعداد عصب کافی در لایه پنهان می‌تواند هر تابع پیوسته را تخمین بزند [6]. بنابراین اساس رویکردهای مبتنی بر

1. Over-fitting
2. Genetic Algorithm

شبکه‌های عصبی برای حل مسائل تصمیم‌گیری چندهدفه، مدل‌سازی ساختار ارجحیت‌های DM با استفاده از یک شبکه عصبی پیشرو مانند شکل (1) می‌باشد.

در این شبکه تعداد ورودی‌ها، برابر با تعداد توابع هدف می‌باشد و شبکه تنها یک خروجی دارد. شبکه تعدادی نمونه آموزشی دریافت می‌کند. هر داده آموزشی شامل ورودی و خروجی متناظر برای آن است. در اینجا ورودی شامل مقادیر توابع هدف به ازای یک راه حل موثر و خروجی میزان مطلوبیت آن راه حل از نظر DM است. پس از آموزش، این شبکه عصبی قادر خواهد بود هر تابع مطلوبیت مورد نظر را تخمین بزند.

بکارگیری شبکه‌های عصبی در تخمین توابع مطلوبیت در مسائل MODM چند مزیت دارد. اول اینکه در این رویکرد، لازم نسیت فرض شود که تابع مطلوبیت ساختار خاصی دارد. بررسی شروط استقلال متقابل، استقلال از مکمل و استقلال جمع پذیری قبل از حل مساله ضروری نیست. همچنین در این رویکرد، شبکه عصبی زمانی که اطلاعات کسب شده از DM تکمیل‌تر شود، قادر به تطبیق و بهبود نمایش ساختار ارجحیت‌ها می‌باشد.



شکل 1 ساختار شبکه عصبی پیشرو در تخمین تابع مطلوبیت در مسایل MODM

استفاده از شبکه‌های عصبی به تخمین تابع مطلوبیت تصمیم‌گیرنده قدمت طولانی ندارد. با این حال در چند دهه گذشته تحقیقاتی در این زمینه شکل گرفته است [30-27]، [7] و [4-3].

اگر چه عملکرد رویکردهای مبتنی بر شبکه‌های عصبی پیشرو در این زمینه خوشایند بوده‌است اما نقص‌هایی در این زمینه وجود دارد: موضوع اول اینکه DM اغلب ارجحیت‌های خود را به صورت غیرمستقیم ارزیابی و آن‌ها را غیردقیق بیان می‌دارد این در حالی‌است که برای آموزش شبکه عصبی مقادیر صریح و دقیق نیاز است. از طرفی دلیل استفاده از شبکه عصبی در اخذ ارجحیت‌های تصمیم‌گیرنده، اجتناب از هرگونه فرضیات قبلی و بیشینه کردن انعطاف پذیری این فرآیند می‌باشد [7]. موضوع دوم این است که در

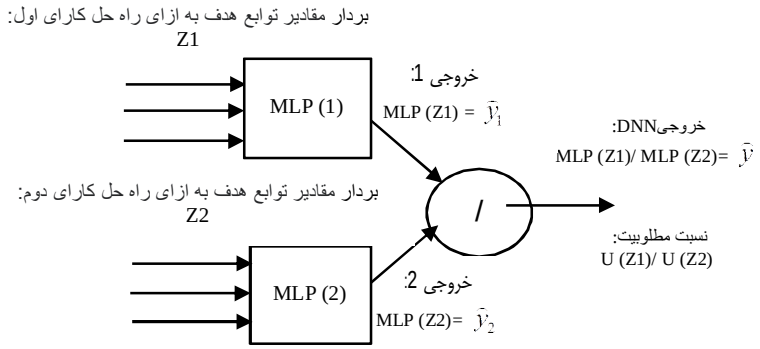
رویکردهای موجود پس از به‌کارگیری مقایسات زوجی، با استفاده از روش‌هایی مانند روش فرایند تحلیلی سلسه‌مراتبی[1] (AHP) که برای محاسبه میزان مطلوبیت یک راه حل استفاده می‌شود. در اینصورت از اطلاعات دریافتی به‌طور کامل استفاده نمی‌شود و این موضوع منجر به اتلاف اطلاعات ارجحیت‌های DM می‌شود. نکته دیگر آن است که نمونه‌های بیشتر در صحت و دقت بیشتر یک شبکه عصبی موثر است. گرفتن نمونه‌های بیشتر برای آموزش بهتر شبکه عصبی نیز نیاز به مقایسات زوجی بسیار زیادی دارد که این خود عملی نیست و در صورت امکان بار زیادی بر دوش تصمیم‌گیرندگان وارد می‌شود. لذا استفاده مستقیم از قضاوت‌های زوجی تصمیم‌گیرندگان علاوه بر حفظ اطلاعات مورد نظر DM، تعداد داده‌های آموزشی بیشتری ایجاد می‌نماید. در میان رویکردهای مبتنی بر شبکه‌های عصبی به منظور تخمین تابع مطلوبیت در مسائل MODM، در سال 2004 شبکه عصبی تصمیم[2] (DNN) معرفی شد [7] که با قابلیت‌های خود مشکلات عمده رویکردهای قبلی را مرتفع ساخته است که در ادامه این شبکه کامل‌تر معرفی شده است.

3-1- شبکه عصبی تصمیم

ساختار DNN در شکل (2) نشان داده شده است. این شبکه، شامل دو شبکه چند لایه پیشرو[3] با ساختار کاملاً یکسان است. خروجی این دو شبکه به یک گره نهایی منتقل می‌گردد. خروجی این گره حاصل تقسیم خروجی‌های این دو شبکه پیشرو با ساختار همانند است. این نکته باعث می گردد که اطلاعات مربوط به مقایسات زوجی راه کارها که توسط تصمیم گیرنده ارائه می گردد، به راحتی برای این شبکه قابل استفاده گردد. زیرا هر گونه پیش پردازش اطلاعات مربوط به مقایسات زوجی، برای یافتن ارزش راه کارها، علاوه بر صرف وقت می‌تواند بر ساختار اطلاعات و آموزش شبکه عصبی تأثیر بگذارد. علاوه بر این، در این روش جداول مقایسات زوجی **که** بصورت ناقص تکمیل شده باشد، نیز می تواند در آموزش این شبکه بکارگرفته شود. مزیت دیگر این شبکه، تعداد داده های آموزشی است که در هر مرحله مقایسه زوجی توسط تصمیم‌گیران به‌دست می‌آید. در حالتی‌که از شبکه های FFANN استفاده شود، تعداد داده‌های آموزشی برابر k خواهد بود، اما در حالتی که از شبکه‌های DNN استفاده نماییم، تعداد داده‌های آموزشی برابر با $k(k\text{-}1)/2$ خواهد بود. در این حالت تعداد مراجعات به تصمیم گیرندگان کمتر خواهد شد و کیفیت آموزش افزایش خواهد یافت [7].

---

1. Analytical Hierarchy Process
2. Decision Neural Network
3. Multi Layer Perceptron



شکل 2 ساختار DNN

از آنجا که هر شبکه عصبی پیشرو سه لایه (لایه های ورودی، میانی و خروجی) با تعداد عصب کافی در لایه پنهان می‌تواند هر تابع پیوسته را تخمین بزند [6]، در رویکردهای مبتنی بر شبکه‌های عصبی برای حل مسائل تصمیم‌گیری چندهدفه پس از آموزش شبکه عصبی توسط داده‌های آموزشی، شبکه قادر خواهد بود که تابع مطلوبیت مسئله را تخمین بزند. بنابراین در DNN به ازای دریافت ورودی‌های Z_1 و Z_2 (دو راه‌حل کارا بر حسب مقادیر اهداف)، نسبت مطلوبیت[1] راه حل اول به راه‌حل دوم تولید می‌شود [8].

فرض می‌کنیم زیر شبکه‌های MLP درDNN دارای دولایه (لایه پنهان و لایه خروجی) باشند، در این‌صورت اگر n تعداد ورودی‌های شبکه عصبی MLP، n_h تعداد نرون‌ها در لایه پنهان، W^h ماتریس وزن‌های لایه پنهان، f_h تابع فعال‌ساز لایه پنهان، O^h بردار خروجی لایه پنهان، W^o بردار وزن های لایه خروجی، f_o تابع فعال‌ساز لایه خروجی باشد، به منظور محاسبه خروجی نهایی گام‌های لازم به صورت زیر می‌باشند:

گام اول: ورودی توابع فعال‌ساز در لایه پنهان برای هر جزء MLP، به ازای ورودی های Z_1 و Z_2 به صورت رابطه (1) بدست می‌آید:

$$net^h_{i\,1} = \sum_{j=1}^{n} W^h_j Z_1 \quad i = 1,\ldots,n_h$$
$$net^h_{i\,2} = \sum_{j=1}^{n} W^h_j Z_2 \quad i = 1,\ldots,n_h \qquad (1)$$

1. Utility

گام دوم: خروجی لایه پنهان با عبور دادن بردارهای $net^h_{i\,1}$ و $net^h_{i\,2}$ از توابع فعال‌ساز بدست می‌آید:

$$\begin{aligned} o^h_{i\,1} &= f_h\left(net^h_{i\,1}\right) \qquad i=1,\ldots,n_h \\ o^h_{i\,2} &= f_h\left(net^h_{i\,2}\right) \qquad i=1,\ldots,n_h \end{aligned} \tag{2}$$

گام سوم: ورودی توابع فعال‌ساز در لایه خروجی به صورت رابطه (3) بدست می‌آید:

$$\begin{aligned} net^o{}_1 &= \sum_{i=1}^{n_h} W^o_i o^h_{i\,1} \\ net^o{}_2 &= \sum_{i=1}^{n_h} W^o o^h_{i\,2} \end{aligned} \tag{3}$$

گام چهارم: خروجی شبکه‌های عصبی MLP در DNN، به صورت رابطه (4) بدست می‌آید:

$$\begin{aligned} \hat{y}_1 &= f_o\left(net^o{}_1\right) \\ \hat{y}_2 &= f_o\left(net^o{}_2\right) \end{aligned} \tag{4}$$

گام پنجم: در نهایت خروجی شبکه عصبی DNN به صورت (5) بدست می‌آید:

$$\hat{y} = \frac{\hat{y}_1}{\hat{y}_2} \tag{5}$$

3-2- ساختار شبکه‌های MLP در DNN

همانطور که ذکر شد، پیچیدگی بیش از حد یک شبکه عصبی، ممکن است به یادگیری بیش از حد آن منجر شود که این امر سبب کاهش قدرت تعمیم شبکه می‌شود. از طرفی پیاده‌سازی شبکه‌های ساده‌تر با سهولت بیشتری انجام می‌شود. هچنین استخراج دانش از شبکه‌های عصی آموزش دیده با افزایش تعداد نرون ها مشکل‌تر می‌شود. همچنین همانطور که در بخش های قبلی نیز ذکر شد، ثابت شده است هر شبکه عصبی پیشرو سه لایه (لایه های ورودی، میانی و خروجی) با تعداد عصب کافی در لایه پنهان می‌تواند هر تابع پیوسته را تخمین بزند. بنابراین در روش پیشنهادی این مقاله برای توسعه و بهبود آموزش DNN، مانند

رویکردهای پیشین از شبکه های پیشرو در ساختارDNN استفاده شده است و پس از آزمون بر روی تعداد لایه های پنهان و تعداد نرون ها، یک لایه پنهان با سه نرون در شبکه‌های MLP در DNN در نظر گرفته شده‌است. توابع فعال‌ساز برای لایه پنهان یکسان و از تابع سیگموئید تک قطبی استفاده شده‌است. در لایه خروجی نیز تابع فعال‌ساز خطی کارگرفته شده است. تعداد ورودی های هر زیر شبکه MLP برابر با تعداد توابع هدف مسأله می‌باشد. از آنجا که خروجی این شبکه مقدار تخمین زده مطلوبیت به ازای هر راه حل اولیه کارآ را نشان میدهند، هر یک از شبکه‌های MLP درDNN دارای یک خروجی می‌باشند.

3-3- آموزش شبکه عصبی تصمیم با استفاده از الگوریتم ژنتیک

از آنجا که DNN دارای ساختار خاصی است، در حین آموزش باید ساختار و پارامترهای دو شبکه پیشرو ANN1 و ANN2 به‌طور یکسان حفظ شود. چن و لین [7] یک الگوریتم آموزشی مبتنی بر مشتقات مرتبه اول برای آموزش DNN، با روش BP که متناسب با ساختار ویژه DNN می‌باشد، ارایه نمودند. روش پس انتشار خطا[1] (BP)، متداول‌ترین روش آموزشی در آموزش شبکه های عصبی پیشرو و یک الگوریتم مبتنی بر مشتقات مرتبه اول است که از سال 1986 برای آموزش این شبکه‌ها معرفی شده است [31]. این روش به کارآیی پایین آموزشی‌اش نیز معروف است. با وجود راهکارهای ارائه شده جهت افزایش سرعت آموزش در الگوریتم BP، مانند نرخ‌های یادگیری پویا یا مومنتم برای افزایش سرعت همگرایی آن، اما در کاربردهای عملی از کارآیی لازم برخوردار نشده‌است. از طرفی با استفاده از الگوریتم‌های تکاملی در آموزش شبکه‌های عصبی، مشکلات روش‌های آموزش مبتنی بر گرادیان برطرف می گردد.

به منظور بکارگیری الگوریتم ژنتیک برای آموزش شبکه عصبی، در ابتدا یک نمایش مناسب برای این شبکه ها باید انتخاب شوند تا به شکل کروموزوم ذخیره شوند [32]. از آنجایی‌که در این پژوهش ساختار شبکه عصبی از پیش تعین شده و ثابت در نظر گرفته شده‌است، کدبندی هر گونه اطلاعات مربوط به معماری شبکه عصبی به صورت کروموزوم لازم نیست. بنابراین تنها لازم است که وزن های شبکه عصبی به شکل کروموزوم کدبندی شوند. برای سهولت می‌توان از بایاس‌ها صرف نظر کرد. علاوه براین لازم است یک تابع برازندگی مخصوص مسئله تعریف شود. در این بخش نحوه کدبندی

1. Back Propagation Error

مسئله و تعریف تابع برازندگی و همچنین تعیین پارامترهای مربوط به الگوریتم ژنتیک بکار گرفته شده در این پژوهش شرح داده شده است.

نوع کد کردن مسئله

برای کد کردن مسئله از متغیرهای حقیقی استفاده می شود. در هر کروموزم ژن‌ها، پارامترهای شبکه عصبی MLP در ساختار DNN می‌باشند. با توجه به اینکه از بایاس صرف نظر شد، تعداد ژن‌های هر کروموزم برابر با تعداد وزن‌های شبکه عصبی MLP می‌باشد. از آنجایی‌که تعداد ورودی‌های MLP برابر با n و تعداد نرون‌ها در لایه میانی برابر با n_h در نظر گرفته شد، تعداد ژن‌ها در هر کروموزم $n \times n_h + n_h$ خواهد بود، که $n \times n_h$ تعداد وزن‌ها در لایه اول و n_h تعداد وزن‌ها در لایه خروجی می‌باشد. از آنجا که تعداد پارامترهای شبکه به اندازه کافی در نظر گرفته شده است، می‌توان از بایاس صرف نظر کرد.

تابع برازندگی[1]

معیار رتبه بندی کروموزم‌ها تابع هزینه زیر می‌باشد که بر اساس خطای بین خروجی مطلوب و خروجی بدست آمده از شبکه، به ازای همه داده‌های آموزشی تعریف می‌شود:

$$E = \sum_{k=1}^{N} \left(y_k - \hat{y}_k \right)^2 \qquad (6)$$

که در آن N تعداد کل داده‌های آموزشی، y_k خروجی مطلوب و $\hat{y}_k$ خروجی DNN به ازای k-امین داده آموزشی می‌باشد.

عملگر تقاطع

برای این عملیات، با توجه به اینکه نوع کدینگ مسئله، متغیرهای حقیقی در نظر گرفته شده است، یک عدد تصادفی تولید شده و بر اساس این عدد تصادفی کروموزم های جدید تولید می‌شوند. به عنوان مثال اگر کرموزم های والدین را P_1 و P_2 در نظر بگیریم، کرومزم‌های جدید C_1 و C_2 به صورت زیر تولید می شوند:

1. Fitness function

$$C_1 = \lambda P_1 + (1-\lambda) P_2$$
$$C_2 = (1-\lambda) P_1 + \lambda P_2 \quad (7)$$

که در آن، λ یک عدد تصادفی بین صفر ویک می‌باشد.

انتخاب والدین

نحوه انتخاب والدین به صورت تصادفی با رویکرد حفظ نخبگان می‌باشد.

شرط توقف الگوریتم

برای توقف الگوریتم دو شرط حداکثر تعداد نسل‌ها و کرانی برای مقدار تابع هزینه بهترین کروموزم در نظر گرفته می‌شود.

4- مثال‌های عددی نمونه (نتایج عددی)

در این بخش چندین مثال کاربردی برای نشان دادن کارایی روش پیشنهادی ارائه می‌گردد. در دو مثال اول قابلیت تخمین تابع ارزیابی مطلوبیت با استفاده از روش پیشنهادی نشان داده شده و در مثال سوم شبکه عصبی DNN در ارزیابی ارجحیت‌های تصمیم‌گیرنده در حل یک مسئله تصمیم‌گیری چندهدفه گسسته به کار برده شده است.

4-1- مثال اول: تابع مطلوبیت خطی

در مثال اول، روش پیشنهادی برای تخمین تابع مطلوبیت زیر به کار برده می‌شود:

$$v(z_1, z_2) = 0.85 z_1 + 0.15 z_2 \quad (8)$$

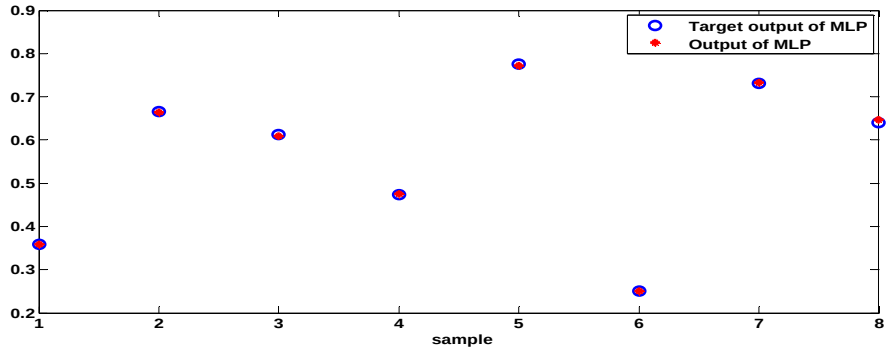
که در آن z_1, z_2 اعداد تصادفی بین صفر و یک هستند. ابتدا هشت جواب اولیه تولید کرده و با استفاده از این هشت جواب به تعداد $8(8-1)/2 = 28$ داده آموزشی برای شبکه عصبی بدست می‌آوریم. با استفاده از این داده‌های آموزشی، وزن‌های شبکه عصبی با روش GA که در بخش قبل مطرح گردید، تنظیم شده‌است. جمعیت اولیه 2000 و تعداد نسل‌ها 200 می‌باشد. نتایج عددی در جدول‌های 1 و 2 و نتایج گرافیکی در شکل‌های (3) تا (5) نشان داده شده است. همانطور که دیده می‌شود روش پیشنهادی نتایج بسیار مطلوبی داشته است.

جدول 1 نتایج تخمین تابع مطلوبیت خطی

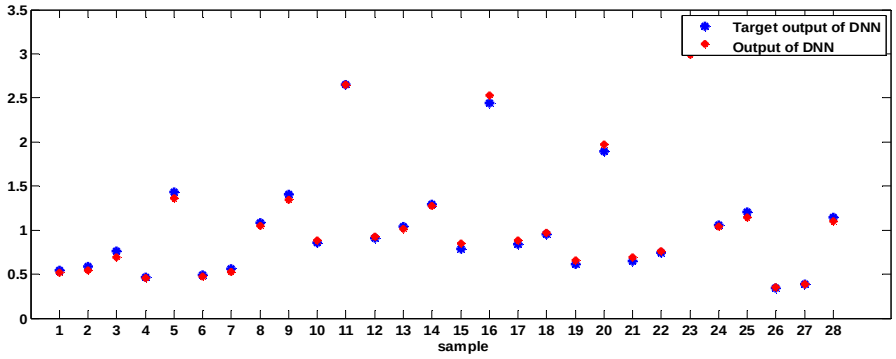
ورودی MLP		خروجی مطلوب $v(z)$	خروجیMLP
Z_1	Z_2		
0,7583	0,2891	0,3595	0,3587
0,2933	0,7315	0,6658	0,6636
0,0152	0,7178	0,6124	0,6087
0,3562	0,4954	0,4745	0,4747
0,0358	0,9059	0,7754	0,7722
0,5438	0,1994	0,2510	0,2501
0,0550	0,8501	0,7309	0,7327
0,0876	0,7384	0,6408	0,6467

جدول 2 نتایج به ازای داده‌های آموزشی تولید شده از طریق مقایسات داده‌های اولیه در مثال اول

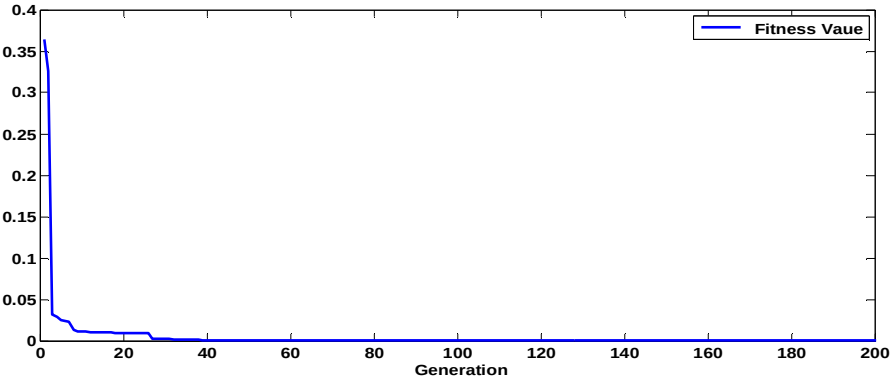
ورودی MLP(1)		ورودی MLP(2)		خروجی مطلوب DNN	خروجی DNN
0,7583	0,2891	0,2933	0,7315	0,5399	0,5405
0,7583	0,2891	0,0152	0,7178	0,5870	0,5893
0,7583	0,2891	0,3562	0,4954	0,7576	0,7555
0,7583	0,2891	0,0358	0,9059	0,4636	0,4645
0,7583	0,2891	0,5438	0,1994	1,4320	1,4340
0,7583	0,2891	0,0550	0,8501	0,4919	0,4895
0,7583	0,2891	0,0876	0,7384	0,5610	0,5546
0,2933	0,7315	0,0152	0,7178	1,0872	1,0903
0,2933	0,7315	0,3562	0,4954	1,4032	1,3979
0,2933	0,7315	0,0358	0,9059	0,8587	0,8594
0,2933	0,7315	0,5438	0,1994	2,6523	2,6533
0,2933	0,7315	0,0550	0,8501	0,9110	0,9057
0,2933	0,7315	0,0876	0,7384	1,0391	1,0261
0,0152	0,7178	0,3562	0,4954	1,2907	1,2822
0,0152	0,7178	0,0358	0,9059	0,7898	0,7883
0,0152	0,7178	0,5438	0,1994	2,4396	2,4336
0,0152	0,7178	0,0550	0,8501	0,8379	0,8307
0,0152	0,7178	0,0876	0,7384	0,9557	0,9411
0,3562	0,4954	0,0358	0,9059	0,6120	0,6148
0,3562	0,4954	0,5438	0,1994	1,8902	1,8981
0,3562	0,4954	0,0550	0,8501	0,6492	0,6479
0,3562	0,4954	0,0876	0,7384	0,7405	0,7340
0,0358	0,9059	0,5438	0,1994	3,0888	3,0873
0,0358	0,9059	0,0550	0,8501	1,0609	1,0538
0,0358	0,9059	0,0876	0,7384	1,2101	1,1939
0,5438	0,1994	0,0550	0,8501	0,3435	0,3413
0,5438	0,1994	0,0876	0,7384	0,3918	0,3867
0,0550	0,8501	0,0876	0,7384	1,1406	1,1329



شکل 3 نمودار مقایسه خروجی مطلوب با خروجی MLP در مثال اول



شکل 4 نمودار مقایسه خروجی مطلوب با خروجیDNN در مثال اول



شکل 5 مقدار تابع هزینه برای بهترین جواب در نسل های مختلف در مثال اول

به منظور مقایسه رویکرد مبتنی بر الگوریتم ژنتیک با روش مبتنی بر گردیان، نتایج میانگین مربعات خطا (MSE) در هر تکراری از اجرای این دو روشبه به ازای همه داده های آموزشی، همچنین جمع ماکزیمم خطاها (ME) در جدول (3) آورده شده‌است تاعملکرد بهتر الگوریتم ژنتیک نشان داده شود. برای مثال تعداد تکرار اجرای الگوریتم آموزشی را 10 در نظر گرفته شده است.

جدول 3 مقایسه خطاهای حاصل از اجرای روش آموزش الگوریتم ژنتیک با روش گرادیان نزولی درمثال اول پس از 10 تکرار

روش آموزش	MSE										ME*
	تکرار1	تکرار2	تکرار3	تکرار4	تکرار5	تکرار6	تکرار7	تکرار8	تکرار9	تکرار10	
گرادیان نزولی	0,3687	0,3661	0,3635	0,3610	0,3584	0,3584	0,3559	0,3533	0,3508	0,3482	10,4027
الگوریتم ژنتیک	0,3334	0,1359	0,0135	0,0135	0,0030	0,0030	0,0005	0,0004	0,0003	0,0002	5,5465

$*ME = \left|MLP(z)K - v(z)\right| / v(z), \quad K = \sum v(z) / \sum MLP(z)$

4-2- مثال دوم: تابع مطلوبیت غیرخطی

در این مثال، تابع مطلوبیت یک تابع غیرخطی به صورت زیر در نظر گرفته می‌شود:

$$v = L - \left(\sum \lambda_i \left(z_i^* - z \right)^p \right)^{\frac{1}{p}} \qquad (9)$$

که در آن، $p = 4, L = 1, \lambda_1 = 0.220, \lambda_2 = 0.472, \lambda_3 = 0.308$ و $z^* = (1,1,1)^T$.

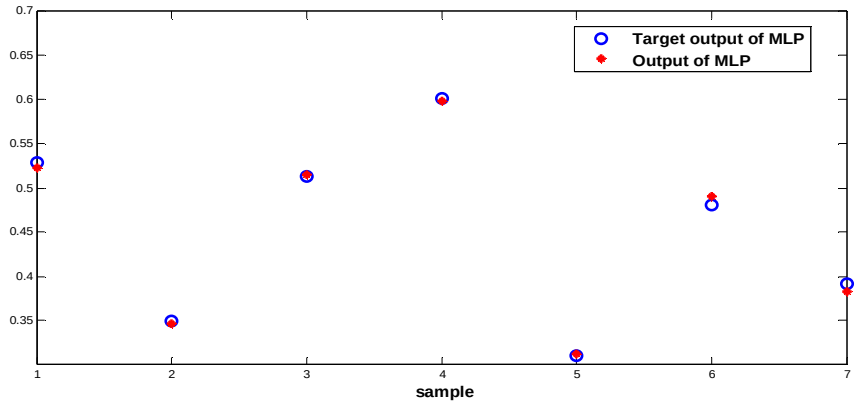
مشابه مثال اول، هفت جواب اولیه بدست آورده و با 21 نمونه آموزشی پارامترهای شبکه عصبی را با روش GA تنظیم می‌کنیم. در این مثال جمعیت اولیه را 2000 و تعداد نسل-ها را 100 در نظر گرفته‌ایم. نتایج عددی در جداول (4) و (5)آورده شده است. همچنین نتایج گرافیکی در شکل های (6) تا (8) نشان داده شده است.

جدول 4 نتایج تخمین تابع مطلوبیت غیرخطی با روش GA

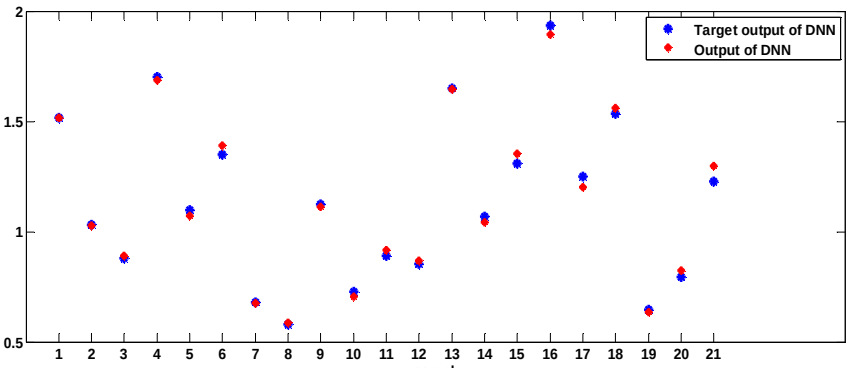
ورودی MLP			خروجی مطلوب	خروجی MLP
Z_1	Z_2	Z_3		
0,5991	0,4798	0,5812	0,5288	0,5228
0,5336	0,2311	0,9855	0,3490	0,3465
0,9023	0,8243	0,2903	0,5129	0,5156
0,4689	0,7961	0,9785	0,6011	0,5980
0,1569	0,9601	0,2483	0,3106	0,3119
0,3113	0,7462	0,7168	0,4808	0,4906
0,2354	0,8952	0,3828	0,3913	0,3826

جدول 5 نتایج به ازای داده‌های آموزشی تولید شده از طریق مقایسات داده‌های اولیه در مثال دوم

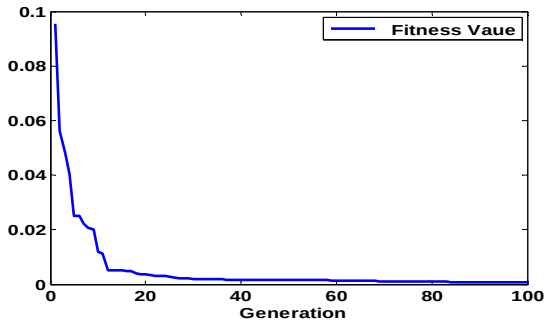
ورودی MLP(1)			ورودی MLP(2)			خروجی مطلوب DNN	خروجی DNN
0,5991	0,4798	0,5812	0,5336	0,2311	0,9855	1,5149	1,5088
0,5991	0,4798	0,5812	0,9023	0,8243	0,2903	1,0310	1,0140
0,5991	0,4798	0,5812	0,4689	0,7961	0,9785	0,8797	0,8741
0,5991	0,4798	0,5812	0,1569	0,9601	0,2483	1,7024	1,6759
0,5991	0,4798	0,5812	0,3113	0,7462	0,7168	1,0998	1,0655
0,5991	0,4798	0,5812	0,2354	0,8952	0,3828	1,3512	1,3664
0,5336	0,2311	0,9855	0,9023	0,8243	0,2903	0,6805	0,6720
0,5336	0,2311	0,9855	0,4689	0,7961	0,9785	0,5807	0,5793
0,5336	0,2311	0,9855	0,1569	0,9601	0,2483	1,1237	1,1108
0,5336	0,2311	0,9855	0,3113	0,7462	0,7168	0,7260	0,7062
0,5336	0,2311	0,9855	0,2354	0,8952	0,3828	0,8919	0,9057
0,9023	0,8243	0,2903	0,4689	0,7961	0,9785	0,8533	0,8621
0,9023	0,8243	0,2903	0,1569	0,9601	0,2483	1,6513	1,6528
0,9023	0,8243	0,2903	0,3113	0,7462	0,7168	1,0668	1,0509
0,9023	0,8243	0,2903	0,2354	0,8952	0,3828	1,3107	1,3476
0,4689	0,7961	0,9785	0,1569	0,9601	0,2483	1,9351	1,9173
0,4689	0,7961	0,9785	0,3113	0,7462	0,7168	1,2502	1,2190
0,4689	0,7961	0,9785	0,2354	0,8952	0,3828	1,5360	1,5632
0,1569	0,9601	0,2483	0,3113	0,7462	0,7168	0,6460	0,6358
0,1569	0,9601	0,2483	0,2354	0,8952	0,3828	0,7937	0,8153
0,3113	0,7462	0,7168	0,2354	0,8952	0,3828	1,2286	1,2824



شکل 6 نمودار مقایسه خروجی مطلوب با خروجی MLP در مثال دوم



شکل 7 نمودار مقایسه خروجی مطلوب با خروجی DNN در مثال دوم



شکل 8 مقدار تابع هزینه برای بهترین جواب در نسل های مختلف در مثال دوم

مشابه مثال قبل، به منظور مقایسه رویکرد مبتنی بر الگوریتم ژنتیک با روش مبتنی بر گردیان، نتایج میانگین مربعات خطا (MSE) در هر تکراری از اجرای این دو روشبه به ازای همه داده های آموزشی، همچنین جمع ماکزیمم خطاها (ME) در جدول (6) آورده شده‌است تاعملکرد بهتر الگوریتم ژنتیک نشان داده شود. تعداد تکرار اجرای الگوریتم آموزشی 10 در نظر گرفته شده‌است که این تعداد اجرا برای الگوریتم مبتنی بر گرادیان بسیار نامناسب می باشد و جمع خطا به ازای این تعداد تکرار بسیار زیاد می باشد.

جدول 6 مقایسه خطاهای حاصل از اجرای روش آموزش الگوریتم ژنتیک با روش گرادیان نزولی درمثال دوم پس از 10 تکرار

روش آموزش	MSE										ME
	تکرار1	تکرار2	تکرار3	تکرار4	تکرار5	تکرار6	تکرار7	تکرار8	تکرار9	تکرار10	
گرادیان نزولی	0,1191	0,1190	0,1189	0,1188	0,1188	0,1187	0,1186	0,1185	0,1184	0,1184	126,6999
الگوریتم ژنتیک	0,0989	0,0712	0,0565	0,0490	0,0490	0,0421	0,0346	0,0346	0,0194	0,0193	14,6887

5-3- مثال سوم- مسئله چندهدفه گسسته

برای نشان دادن قابلیت الگوریتم های فراابتکاری در حل مسائل گسسته، در این مثال یک مسئله تصمیم‌گری چندهدفه عددصحیح در نظر گرفته شده‌است، تا کاربرد مؤثر شبکه عصبی DNN مبتنی بر GA، در حل این دسته از مسائل نشان داده شود.

$$
\begin{aligned}
&\max z_1 = 2x_2 + 5x_3 + 5x_4 - 2x_5 + 5x_6 \\
&\max z_2 = -x_1 - 2x_2 + 4x_5 - x_6 \\
&\max z_3 = 5x_1 + 3x_2 - 2x_3 - x_5 - x_6 \\
&\textit{subject to} \\
&7x_4 + 2x_5 + 6x_6 \le 28 \qquad (10) \\
&3x_1 + 4x_6 \le 23 \\
&4x_1 + 4x_3 + x_4 \le 23 \\
&x_2 + 6x_3 + 7x_4 + 4x_6 \le 23 \\
&2x_1 + 5x_2 + 5x_3 + 5x_4 + 8x_5 \le 29 \\
&x_j \,\text{int}\, eger,\ 1 \le j \le 6
\end{aligned}
$$

تابع مطلوبیت به صورت زیر می‌باشد:

$$v = 50 - \left(\sum_{i=1}^{3} \lambda_i \left(z_i^{\max} - z_i \right)^4 \right)^{\frac{1}{4}} \tag{11}$$

که در آن، $\lambda_1 = 0.319, \lambda_2 = 0.416, \lambda_3 = 0.265$.

در این مثال راه حل‌های ایدآل و ضد-ایدآل به صورت زیر می‌باشند:

$$z^{maz} = (30,12,34),$$
$$V(z^{\max}) = 50.000$$

$$z^{\min} = (-6,-16,-8),$$
$$V(z^{\min}) = 35.1093$$

سپس برای نرمال سازی راه‌حل‌ها و تابع مطلوبیت به صورت زیر عمل می‌کنیم:

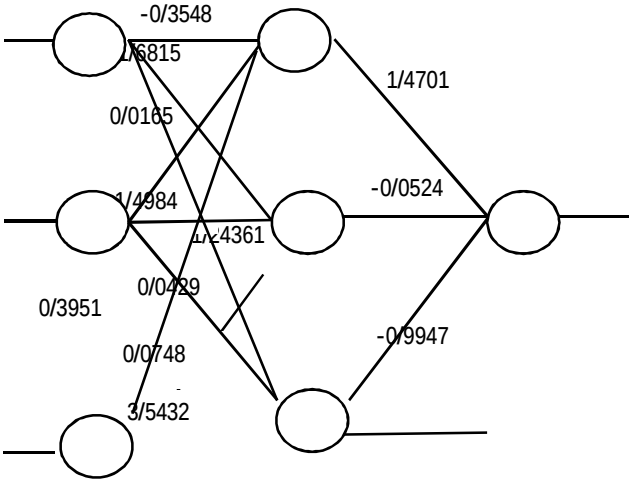
$$z_i' = \frac{z_i - z_i^{\min}}{z_i^{\max} - z_i^{\min}}$$
$$V'(z) = \frac{V(z) - V(z^{\min})}{V(z^{maz}) - V(z^{\min})} \tag{12}$$

برای حل این مسئله به کمک روش پیشنهادی، هفت جواب اولیه متفاوت با استفاده از نرم افزار گمز و با روشL-P متریک از این مسئله به صورت جدول (7) بدست آورده شده‌است. با استفاده از این جواب‌های اولیه مشابه مثال‌های اول و دوم، تابع ارزیابی مطلوبیت (11) با شبکه عصبی تخمین زده می‌شود. سپس مسئله به صورت مدل ریاضی (13) بازنویسی می‌شود که با حل این مسئله بهینه‌سازی، جواب نهایی را می‌توان بدست آورد. در این مثال در الگوریتم ژنتیک، جمعیت اولیه 2000 و تعداد نسل‌ها 150 در نظر گرفته شده است. در ساختار DNN، شبکه‌های MLP سه لایه در نظر گرفته شده اند. تعداد نرون ها در لایه اول این شبکه‌ها برابر تعداد اهداف یعنی 3 می باشد. همچنین تعداد نرون‌ها در لایه پنهان برابر 3 و در لایه خروجی برابر با 1 در نظر گرفته شده است. توابع فعال‌ساز برای لایه پنهان یکسان و از تابع سیگموئید تک قطبی استفاده شده است. در لایه خروجی نیز تابع فعال‌ساز خطی بکارگرفته شده است. وزن‌های نهایی شبکه MLP در DNN شکل 7 آورده شده است. به طور میانگین زمان آموزش این شبکه با روش پیشنهادی، با تعداد تکرار 150و میانگین خطای

0,0017، حدود 53 ثانیه به طول انجامید. همچنین نتایج گرافیکی در شکل‌های 8 و9 کارآیی روش پیشنهادی را نشان می‌دهند.

جدول 7 جواب‌های اولیه برای مسئله مثال سوم

راه حل	تابع هدف اول		تابع هدف دوم		تابع هدف سوم		$V'(Z)$
	Z_1	Z_1'	Z_2	Z_2'	Z_3	Z_3'	
1	6	0,3333	-11	0,1786	34	1	0,2998
2	16	0,6111	-13	0,1071	32	0,9524	0,2957
3	22	0,7778	-9	0,25	26	0,8095	0,4118
4	21	0,7500	2	0,6429	2	0,2381	0,4389
5	11	0,4722	2	0,6429	16	0,5714	0,5454
6	21	0,7500	4	0,7143	-8	0	0,2754
7	18	0,6667	-3	0,4643	22	0,7143	0,5614



شکل 7 وزن‌ها نهایی شبکهMLP

$$\max \ \mathrm{MLP}(z')$$

$$z_1' = \frac{z_1 + 6}{30 + 6}$$

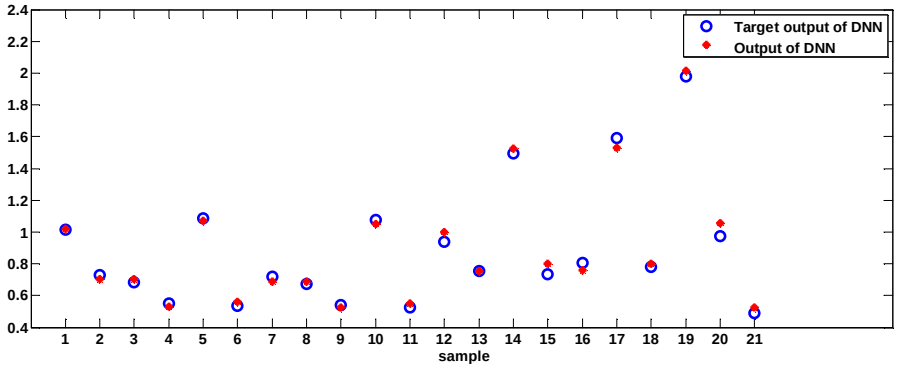
$$z_2' = \frac{z_2 + 16}{12 + 16}$$

$$z_3' = \frac{z_3 + 8}{34 + 8}$$

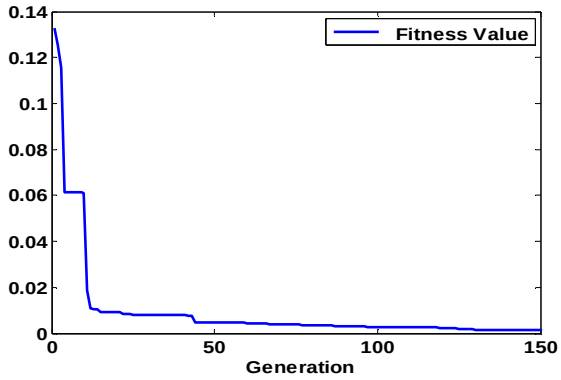
$$z_1 = 2x_2 + 5x_3 + 5x_4 - 2x_5 + 5x_6$$
$$z_2 = -x_1 - 2x_2 + 4x_5 - x_6$$
$$z_3 = 5x_1 + 3x_2 - 2x_3 - x_5 - x_6$$

subject to:

$$\begin{aligned} &7x_4 + 2x_5 + 6x_6 \le 28 \\ &3x_1 + 4x_6 \le 23 \\ &4x_1 + 4x_3 + x_4 \le 23 \\ &x_2 + 6x_3 + 7x_4 + 4x_6 \le 23 \\ &2x_1 + 5x_2 + 5x_3 + 5x_4 + 8x_5 \le 29 \\ &x_j \ \text{integer}, 1 \le j \le 6 \end{aligned} \qquad (13)$$



شکل 8 نمودار خروجی واقعی به همراه خروجی تخمین زده شده در مثال سوم



شکل 9 مقدار تابع هزینه برای بهترین جواب در نسل های مختلف در مثال سوم

5- نتیجه‌گیری

در حل مسائل چند هدفه در شرایط واقعی، تخمین تابع مطلوبیت با روش‌های سنتی همواره کاری پیچیده و همراه با خطا می‌باشد. با توجه به تأیید محققین در توانایی شبکه‌های عصبی مصنوعی در تخمین توابع پیچیده و غیرخطی، بکارگیری شبکه‌های عصبی تصمیم برای حل مسائل چندهدفه پیشنهاد می‌شود. مطابق مثال اول، اگر به‌طور پیش فرض ساختار ارجحیت تصمیم‌گیرنده با یک تابع خطی قابل نمایش باشد، در این‌صورت ممکن است اصلا لزومی به استفاده از شبکه عصبی نباشد. این در حالی‌است که یکی از مزایای مدل‌های شبکه عصبی توانایی خودسازماندهی است. با وجود این خصوصیت شبکه‌های عصبی، ساختار ارجحیات تصمیم‌گیرنده، بر اساس اطلاعات ترجیحی وی بدون تحقیق و تایید درباره شکل تابع مطلوبیت، ارزیابی می‌گردد. به عبارتی فرآیند ارزیابی ساختار ارجحیت های تصمیم‌گیرنده در شبکه‌های عصبی به‌طورذاتی مستقل از هر گونه پیش فرض درباره تابع مطلوبیت می‌باشد. بنابراین، هر چقدر فضاهای تصمیم‌گیری پیچیده‌تر یا ساختار ارجحیات تصمیم‌گیرنده پیچیده‌تر باشد، رویکرد شبکه عصبی تصمیم بسیار موثرتر خواهد بود. از آنجا که توابع مطلوبیت غیرخطی انعطاف پذیرتر و تطبیق پذیرتر از نوع خطی آن هستند، اغلب فرض می‌شود که تابع مطلوبیت تصمیم‌گیرنده غیرخطی می‌باشد. لذا توانایی تخمین توابع مطلوبیت غیرخطی در رویکرد بکارگرفته شده در حل مسائل MODM، حائز اهمیت است. که نتایج مثال‌های ارائه شده در این حوزه حاکی از کارآیی رویکرد پیشنهادی در این خصوص می‌باشد. لازم به تاکید است که از آنجا که مدل های شبکه عصبی مدل های غیرخطی هستند، تخمین توابع غیرخطی پیچیده تر به مراتب سریع‌تر توابع خطی صورت می‌گیرد. همچنین لازم

به ذکر است که با بکارگیری شبکه عصبی تصمیم علاوه بر بهره‌مندی از مزایای بالقوه شبکه‌های عصبی در تخمین توابع غیرخطی و پیچیده، مشکلات رویکردهای پیشین مبتنی بر شبکه‌های عصبی در حل مسائل MODM نیز برطرف می‌گردد. به‌طوری که با تعداد اندک جواب اولیه، تعداد قابل توجهی نمونه آموزشی برای آموزش شبکه فراهم می‌گردد. به‌گونه ای که در حل مسائل واقعی با تعداد مراجعات بسیار کم به DM، فرآیند ارزیابی ارجحیات موثرتر انجام می‌گردد.

در مثال‌های نمونه یک تابع مطلوبیت مشخص، فرض گردید. ممکن است این سوال مطرح شود که در عمل، تابع مطلوبیت تصمیم گیرنده از قبل مشخص نیست. لازم به ذکر است در واقع تابع مطلوبیت فرضی در این مسئله نقش تصمیم‌گیرنده را ایفا می‌کند. به عبارتی چون در حل مسئله تصمیم گیرنده حضور ندارد، یک تابع مطلوبیت مفروض شده است. در واقع به ازای اینکه مطلوبیت جواب های موثر اولیه از تصمیم‌گیرنده پرسیده شود، از طریق تابع مطلوبیت فرضی محاسبه می‌گردد. پس از محاسبه مطلوبیت‌های جواب های اولیه، مقایسات زوجی انجام می‌شود. این معادل این است که در عمل یک تصمیم‌گیرنده واقعی وجود دارد که ارزیابی راه‌حل های ارائه شده به وی را، به صورت مقایسات زوجی انجام می‌دهد. بنابراین این فرض خدشه‌ای به کلیت مسئله وارد نمی‌سازد. علاوه براین فرض وجود تابع مطلوبیت مزیتی دیگر نیز دارد و آن برای محک و مقایسه جواب بدست آمده از حل مسئله با جواب بهینه تابع مطلوبیت فرضی است که در مثال های حل شده، در جداول نشان‌دهنده خروجی‌های شبکه، این مقایسه قابل مشاهده است.

در این تحقیق، با هدف توسعه و بهبود کارایی شبکه عصبی تصمیم، رویکردی نوین مبتنی بر الگوریتم ژنتیک برای آموزش این شبکه ارایه گردید تا مشکلات بکارگیری روش‌های آموزشی مبتنی بر گرادیان برطرف گردد. رویکرد ارایه شده برای حل مثال‌های عددی با تابع مطلوبیت خطی و غیر خطی به‌کار گرفته شد. نتایج عددی حاکی از آن است که رویکرد پیشنهادی قادر است در مسائل کاربردی از جمله برنامه ریزی عدد صحیح به‌طور کارآمد مورد استفاده قرار بگیرد.

به‌طور کلی در تحلیل و نتیجه‌گیری این پژوهش، در رابطه با رویکرد بکارگیری DNN مبتنی بر الگوریتم ژنتیک در ارزیابی ساختار ارجحیت‌های DM قابل توجه است:

- کاربرد گسترده: به دلیل عدم نیاز به هر گونه مفروضات بر ساختار ارجحیات DM (قابلیت تخمین هر نوع تابع مطلوبیت غیرخطی) و قابلیت بکارگیری در مسائل گسسته؛

- استفاده آسان: به دلیل کاهش قابل توجه بار شناختی بر دوش DM، با استفاده شکل ساده تر اطلاعات DM (امکان استفاده مستقیم از مقایسات زوجی DNN)؛
- فرآیند موثر ارزیابی: به دلیل استفاده کامل ازاطلاعات مقایسات زوجی؛
- بهبود فرآیند آموزش شبکه و افزایش کیفیت جواب: به دلیل استفاده از الگوریتم ژنتیک (غلبه بر عمده مشکلات روش های مبتنی بر گرادیان).

6- منابع

[1] Yao, X., Evolving artificial neural network, Proc. IEEE, 1999, vol. 87, pp. 1423-1447.

[2] Mitchell, M., An introduction to genetic algorithm (Complex Adaptive Systems), The MIT Press, 1998.

[3] Wang, J., & Malakooti, B., A feed forward neural network for multiple criteria decision making, Computer & Operations Research, 1992, Vol. 19, No. 2, pp. 151-167.

[4] Sun, M., Stam, A., Steuer, R. E., Interactive multiple objective programming problems using Tchebycheff programs and artificial neural networks, Computer & Operations Research, 2000, Vol. 27, No.7, PP.601-620.

[5] Haykin, S., Neural networks: A comprehensive foundation. New York: Macmillan, 1994.

[6] Hecht-Nielsen, R., Theory of the backpropagation neural networks, International Joint Conference on Neural Networks, 1989, pp. 593–611.

[7] Chen, J., Lin, S., A neural network approach - decision neural network (DNN) for preference assessment, IEEE Transaction on Systems, Man and Cybernetics net, 2004, vol. 34, No. 2, pp.219-225.

[8] Chen, J. & Lin, S., An interactive neural network-based approach for solving multiple criteria decision-making problems, Decision support systems, 2003, vol. 36, pp. 137-146.

[9] Lang, K. J., Waibel, A.H., Hinton, G.E., A Time-delay neural network architecture for isolated word recognition, Neural Network, 1990, vol. 3, pp. 23-43.

[10] Whitley, D., Starkweather, T., Bogart, C., Genetic algorithm and neural networks: optimizing connections and connectivity, Parallel Computing, 1990, vol. 14, pp. 347-361.

[11] Stanley, K. O., Miikkulainen, R., Evolving neural network through augmenting topologies, Evolutionary computation, 2002, vol. 10, pp. 99-127.

[12] Fels, S. S., Hinton, G. E., Glove-talk: A neural network interface between a data-glove and a speech synthesizer, Neural Networks, IEEE Transactions on Neural Network, 1993, vol. 4, pp.2-8.

[13] Knerr, S., Personnaz, L., Dreyfus, G., Handwritten digit recognition by neural networks with single-layer training, Neural Networks, IEEE Transactions on Neural Network, 1992, vol. 3, pp.962-968.

[14] Sutton, R. S., Two problems with backpropagation and other steepest-descent learning procedures for networks, in proceeding of the Eighth Annual Conference of the Cognitive Science Society, 1986, pp.823-831.

[15] Whitley, D., The GENITOR algorithm and selection pressure: why rank-based allocation of reproductive trials is best, In: Proceedings of the 3th International Conference on Genetic Algorithms, 1989, PP. 116-123.

[16] Porto, V. W., Fogel, D. B., Alternative neural network training methods, IEEE Expert, 1995, vol. 10, pp. 16-22.

[17] Bartlett, P., Downs, T., Training a neural network with a genetic algorithm: Department of Electrical. University of Queensland, 1990.

[18] Sexton, R., S., Dorsey, R., E., Johnson, J., D., Toward global optimization of neural networks: a comparison of the genetic algorithm and the backpropagation, Decision Support Systems, 1998, vol. 22, pp.171-185.

[19] Topchy, A.P., lebedko, O. A., Neural network training by means of cooperative evolutionary search, Nuclear Instrument and Method in Physics Research

Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 1997, vol. 389.

[20] Kinnebrock, W., Accelerating the standard backpropagation method using a genetic approach, Neurocomputing, 1994, vol. 6, pp.583-588.

[21] Chen, Y. M.,OConnell, R. M. Active power line conditioner with a neural network control, Industry Application, IEEE Transaction on, 1997, vol. 33, pp. 1131-1136.

[22] Belew, R. K., McInerney, N. N. schraudolph, Evolving networks: using the genetic algorithm with connectionist learning, in Proc, Second Conference on Artificial Life, 1991, pp.511-547.

[23] Hancock, P. J. B., Genetic Algorithms and permutation problems: a comparison of recombination operators for neural net structure specification, in Proceeding of the International Workshop on Combination of Genetic Algorithms and Neural Networks (COGANN-92), 1992,PP.108-122.

[24] Montana, D. Davis, L., Training feed forward neural networks using genetic algorithms, In: Proceedings of the 11th International Joint Conference on Artificial Intelligence, 1989, pp. 762-767.

[25] Deb, K., Anand, A., Joshi, D., A computationally efficient evolutionary algorithm for real-parameter optimization, Evolutionary Computation, 2002, vol. 10, pp. 371-395.

[26] Cantu-paz, E., Kamath, C., An empirical comparison of combinations of evolutionary algorithms and neural network for classification problems, IEEE Transaction on Systems, Man and Cybernetics, Part B: Cybernetics, 2005, vol. 35, pp. 915-927.

[27] Malakooti, B., Zhou, Y., Feed-forward artificial neural networks for solving discrete multiple criteria decision making problems, Management Science, 1994, Vol. 40, No. 11, pp. 1542–1560.

[28] Sun, M., Stam, A., Steuer, R. E., Solving multiple objective programming problems using feed-forward artificial neural networks: The interactive FFANN procedure, Management Science, 1996, Vol. 42, No.6, pp. 835-849.

[29] Malakooti, B., Zhou, Y., Approximating polynomial functions by feedforward artificial neural networks: Capacity, analysis, and design, Applied Mathematics and Computation, 1998, Vol. 90, No.1, pp. 27-51.

[30] QU, Li Li, Chen, Yan, An interactive integrated MCDM based on FFAN and application in the selection of logistic center location, 14th International Conference on Management Science & Engineering, 2007.

[31] Rumelhart, D. E., Geoffrey, E. H., Walliams, R.J., Learning representations by back-propagating errors, Nature, 1986, Vol. 323, pp.533-536,.

[32] Mahajan, R., Kaur, G., Neural network using genetic algorithm, International Journal of computer applications, 2013,vol. 77, pp.6-11.