فشرده‌سازی اطّلاعات متغیر با زمان با استفاده از کد هافمن

حسن فرسی[1] و پوریا اعتضادی فر[2]

[1و2] دانشکده مهندسی برق و کامپیوتر، دانشگاه بیرجند، بیرجند، ایران

چکیده

در این مقاله با استفاده از شبکهٔ عصبی مصنوعی، تابعی بر منحنی چگالی احتمال، رشته‌اطّلاعات افراز می‌شود. این امر باعث می‌شود که منحنی چگالی احتمال رشته‌اطّلاعات به‌وسیلهٔ تعدای پارامتر تقریب زده شده و از آن به‌منظور حافظه‌دارکردن منحنی چگالی احتمال و نیز جهت فشرده‌سازی اطّلاعات با روش هافمن استفاده می‌شود. تفاوت بین روش پیشنهادی و روش عمومی الگوریتم هافمن در این است که در روش پیشنهادی به جهت حافظه‌دارکردن منحنی چگالی احتمال، می‌توان چندین‌بار از الگوریتم هافمن برای کدکردن رشته‌اطّلاعات استفاده کرد و در هر بار کدکردن اطّلاعات، تابع چگالی احتمال را تقریب‌زده و اطّلاعات مربوط به تابع چگالی احتمال را به انتهای رشتهٔ مورد نظر اضافه کرد. بنابراین علی‌رغم متغیربودن تابع چگالی احتمال در هر مرحله، الگوریتم هافمن با استفاده از روش پیشنهادی قابل پیاده‌سازی است. همچنین دو الگوریتم برای کدکردن و دیکدکردن رشته‌اطّلاعات پیشنهاد شده است. درانتها درصد فشرده سازی روش پیشنهادی با دو روش FDR Code وGolomb مقایسه می‌گردد.

واژگان کلیدی: تابع چگالی احتمال متغیر با زمان - شبکه عصبی - فشرده سازی اطّلاعات - حافظه‌دارکردن رشته‌اطّلاعات-کد هافمن.

۱- مقدّمه

در علوم رایانه و تئوری اطّلاعات، الگوریتم هافمن یک الگوریتم کدگذاری برای فشرده‌سازی بی اتلاف اطّلاعات است (Huffman, 1952, Proakis, 1995). این تعبیر به استفاده از جدول کد طول متغیر برای کدکردن هر کدام از نشانه‌های مبدأ برمی‌گردد . جدول کد طول متغیر، از روش ویژه‌ای مبتنی بر احتمال وقوع هر کدام از نشان‌های مبدا به‌دست می‌آید. این روش توسط *دیوید هافمن* توسعه‌یافت (Proakis, 1995) .

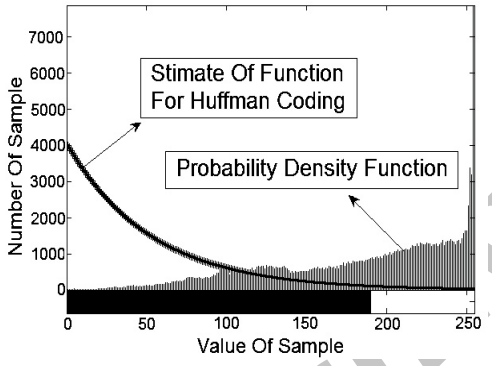
در کدگذاری هافمن، از روشی خاص جهت نحوهٔ نمایش هر نماد استفاده می‌شود. به این روش، کدهای پیشوندی گفته می‌شود. منظور از کدهای پیشوندی آن است که در این روش رشته‌ای که نشان دهندهٔ یک نویسهٔ خاص است، هیچ‌گاه پیشوند رشتهٔ دیگر که نمایان‌گر نویسهٔ دیگراست، نیست. در این روش نویسه‌های پرکاربردتر با طول رشته‌بیت کوتاه‌تری نسبت به نویسه‌هایی که کاربردشان کمتر است، نشان داده می‌شوند (Huffman, 1952) . بنابراین رشته‌بیت ارسالی با استفاده از این روش کوتاه‌تر از حالتی است که به تمام اطّلاعات تعداد بیت برابری اختصاص داده شود.

در بخش ۲ نشان داده می شود که کدگذار هافمن توانایی بالایی در میزان فشرده‌سازی رشته‌اطّلاعات با تابع چگالی احتمال متغیر با زمان را ندارد . در این مقاله با استفاده از حافظه‌دارکردن تابع چگالی احتمال اطّلاعات و شبکهٔ عصبی، می‌توان برای هر رشته تابع چگالی احتمال مربوط به آن رشته را ارسال کرد. همچنین به‌دلیل توانایی در ذخیره‌کردن اطّلاعات تابع چگالی احتمال، می‌توان الگوریتم فشرده‌سازی را چندین بار تکرارکرده و رشته‌اطّلاعات ارسالی را عاری از هرگونه خطا و تلفات بازسازی کرد.

در این مقاله کاستی‌های کد هافمن در کدکردن رشته‌هایی با تابع چگالی متغیر با زمان نشان داده می شوند. در ادامه روش پیشنهادی گام به گام توضیح داده می‌شود . در بخش سه نحوهٔ به‌دست‌آوردن تابع چگالی احتمال برای یک رشته‌بیت با طول دلخواه بررسی و سپس در بخش چهار مفهوم شبکهٔ عصبی به‌صورت مختصر بیان می‌شود. در بخش پنج منظور از حافظه‌دارکردن رشته‌اطّلاعات را توضیح داده و سپس به مزیت حافظه‌دارکردن رشته‌اطّلاعات پرداخته می‌شود. در ادامه به توضیح الگوریتم فشرده‌سازی به‌همراه یک مثال برای واضح‌ترشدن مسأله پرداخته می‌شود. در قسمت پایانی، برای ارزیابی الگوریتم پیشنهادی، درصد فشرده‌سازی روش پیشنهادی با روش‌های FDR (Chandra, et al., 2003) Code و Golomb[1] (Chandra, et al., 2003, Chakrabarty, et al., 2001, Iyengar, et al., 2000, Jas, et al., 1999, Touba, et al., 1998) مقایسه می‌شود .

۲- مشکل کد هافمن

عملکرد کد هافمن (Huffman, 1952, Proakis, 1995, Chandra, et al., 2003, Shi, et al., 2000) بر پایهٔ احتمال رخداد رشته‌هاست. در این روش رشته‌هایی که احتمال رخداد بیشتری دارند با بیت‌های کمتری ارسال می‌شوند، اما این روش کدگذاری محدودیت دارد. محدودیت کدگذاری با روش هافمن این است که باید قبل از ارسال اطّلاعات، تابع چگالی احتمال رخداد نمونه‌ها [2] ، برای داده‌هایی که قرار است ارسال شوند، محاسبه شود. همچنین تابع چگالی احتمال باید در گیرنده نیز ساخته شده باشد، یا اینکه هم‌زمان با داده‌های ارسالی به گیرنده ارسال شود، تا گیرنده بتواند داده‌های دریافتی را به‌درستی بازیابی کند. امّا در مواقعی که تابع چگالی احتمال رخداد نمونه‌ها متغیر با زمان است دیگر کدگذاری به روش هافمن، به‌تنهایی، کارایی مناسبی ندارد؛ زیرا در ابتدا تابع چگالی احتمالی به‌صورت میانگین تمام اطّلاعات ارسالی فرض می‌شود و یا اینکه تابع چگالی احتمال برای اوّلین رشتهٔ تخمین زده شده و بر اساس آن سیستم شروع به کدگذاری و کدگشایی می‌کند. اما برای رشته‌هایی که دارای تابع چگالی احتمال متغیر با زمان هستند، نمی‌توان یک تابع ثابت با زمان را به‌عنوان تابع چگالی احتمال رشته‌اطّلاعات در نظر گرفت. چرا که اگر یک تابع ثابت با زمان به عنوان تابع چگالی احتمال انتخاب گردد، کد هافمن دیگر قادر به فشرده‌سازی اطّلاعات به بهترین نحو نیست. به‌عبارت دیگر پس از گذشت مدت زمانی، تابع چگالی احتمال دچار تغییر شده، در حالی‌که سامانه با توجه به تابع چگالی احتمال اولیه، عملیات فشرده‌سازی را انجام می‌دهد. بنابراین به‌اشتباه اطّلاعاتی که احتمال رخداد آنها کم است، با تعداد بیت کم و اطّلاعاتی را که احتمال رخداد آنها زیاد است با تعداد بیت بیش‌تر کد می‌کند. پس از کدگشایی، به‌دلیل اینکه تابع چگالی احتمال به‌درستی انتخاب نشده، عدم فشرده‌سازی بالای اطّلاعات را به‌همراه داشته است (Chandra, et al., 2003) و در مواردی حتی رشتهٔ کدشدهٔ ارسالی دارای تعداد بیت بیش‌تری نسبت به رشتهٔ کدنشدهٔ اصلی است(شکل ۱ ، جدول ۱).



(شکل ۱): نمونه ای از تخمین نادرست تابع چگالی احتمال به دلیل متغیر بودن هیستوگرام تعداد تکرار نمونه ها در رشته‌اطّلاعات

همان‌طور که در جدول۱ نشان داده شده است، درصورت بروز خطا در تقریب تابع چگالی احتمال نمونه‌ها، تعداد بیت ساخته‌شده با روش هافمن بیشتر از بیت‌های ورودی است. این مشکل زمانی به‌وجود می‌آید که تابع چگالی احتمال اطّلاعات متغیر با زمان باشد. متغیر با زمان‌بودن تابع چگالی احتمال ممکن است پس از مدتی حالتی مانند شکل ۱ را

[1] نتایج گزارش شده برای درصد فشرده‌سازی از (Chandra, 2000, Chakrabarty, 2001) بهتر است به‌دلیل استفاده از روش بهبود الگوریتم غیر مستدل دوباره مرتب‌کردن

[2] در این مقاله تابع چگالی احتمال مربوط به رشته‌هایی با مقادیر [۲۵۵و...و۳و۲و۱و۰] که در مبنای دودویی شامل تک‌بایت اطّلاعات می‌شوند، می‌باشد. این تابع بر اساس تعداد تکرار این ۲۵۶ عدد در یک رشته‌بایت اطّلاعات ساخته می‌شود

به‌وجودآورد. برای حل این موضوع می‌توان در زمان‌های مشخص، تابع چگالی احتمال را برای یک‌سری رشته تخمین‌زده و سپس تابع تخمین‌زده‌شده را به‌همراه رشته‌بیت اطّلاعات به گیرنده ارسال کرد.

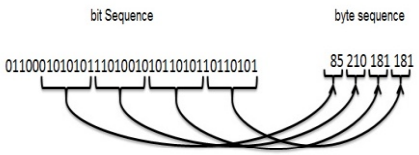
(جدول ۱): افزایش طول رشته‌اطّلاعات پس از کدکردن با الگوریتم هافمن درصورت تخمین نادرست تابع چگالی احتمال با توجه به مثال شکل(۱)

طول رشته‌بیت خروجی پس از کدکردن با روش هافمن با تابع چگالی احتمال با میزان خطای بالا	طول رشته‌بیت ورودی
۲۴۲۶۶۶۶ (bit)	۱۷۲۰۱۹۲ (bit)

در گیرنده نیز ابتدا تابع چگالی احتمال ساخته می‌شود و با استفاده از آن، گیرنده شروع به کدگشایی اطّلاعات می‌کند. اما مشکل بر سر نحوهٔ تخمین تابع چگالی احتمال نمونه‌ها و نحوهٔ ارسال آنهاست. در این مقاله یک روش نوین برای تخمین و ارسال تابع چگالی احتمال بر مبنای کد هافمن و استفاده از شبکهٔ عصبی به‌منظور حافظه‌کردن رشته‌اطّلاعات ارسالی برای افزایش میزان فشردگی ارائه شده است. همچنین سعی شده تا رشته‌بیت‌ها را حافظه‌دار کرد تا بتوان چندین‌بار الگوریتم هافمن را با توابع مختلف چگالی احتمال پیاده‌سازی کرد. در روش پیشنهادی به‌دلیل حافظه‌دارکردن این نوع کدگذاری به‌راحتی می‌توان رشته‌بیت‌ها را در گیرنده بازیابی کرد.

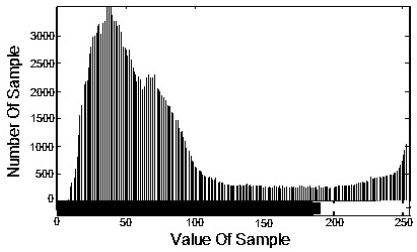
۳-نحوهٔ به‌دست‌آوردن تابع چگالی احتمال برای یک رشته‌بیت با طول دلخواه

برای به‌دست آوردن تابع چگالی احتمال، نیاز به تعداد بیت زیادی است. بدین‌منظور در ابتدا طول مشخصی از بیت‌ها را از رشته‌اطّلاعات جدا کرده یا اینکه سامانه منتظر می‌ماند تا بافر به اندازه طول مورد نظر پر شود. در مرحله بعد، پس از پرشدن بافر توسط بیت‌های رشته‌اطّلاعات، هر هشت بیت از رشتهٔ مورد نظر به مبنای ۱۰ تبدیل‌شده و سپس به بافری دیگر انتقال داده می‌شوند(شکل ۲).



(شکل ۲): نحوه جدا کردن بیت ها از رشته‌اطّلاعات و ساختن رشته‌ای از بایت‌ها

با این کار بافر به جای اعداد ۱ و ۰ اعدادی بین ۰ تا ۲۵۵ را در مبنای ده شامل می‌شود. در مرحلهٔ بعد، تعداد تکرار این ۲۵۶ عدد در رشتهٔ مورد نظر محاسبه می‌شود. بنابراین می‌توان هیستوگرام را بر حسب تعداد تکرار این ۲۵۶ عدد ساخته‌شده در رشتهٔ مورد نظر به‌دست آورد. به‌عنوان مثال، شکل ۳ نمونه‌ای از هیستوگرام یک رشته‌بیت پس از تبدیل را نشان می‌دهد.



(شکل ۳): هیستوگرام تعداد تکرار نمونه ها در یک رشته‌اطّلاعات

پس از این مرحله باید تابع چگالی احتمال را از هیستوگرام (شکل ۳) ساخته شده به‌دست آورد. تابع چگالی احتمال به‌دست آمده از هیستوگرام ساخته شده باید شرط‌های (۱) و (۲) و (۳) را برآورده کند.

$$f_x(x) \geq 0 \qquad (۱)$$

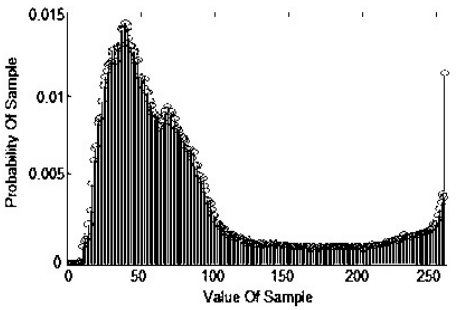
$$f_x(x) \leq 1 \qquad (۲)$$

$$\sum_{-\infty}^{\infty} f_x(x) = 1 \qquad (۳)$$

در سه شرط (۱) و (۲) و (۳)، $f_x(x)$ مقدار تابع هیستوگرام ساخته شده است، که نشان‌دهندهٔ تعداد تکرار اعداد بین ۰ تا ۲۵۵ است. با بررسی هیستوگرام، می‌توان نشان داد که شرط (۱) برآورده می‌شود، زیرا تعداد تکرار یک عدد همواره عددی بزرگ‌تر و یا مساوی با صفر خواهد بود (شکل۳). اما تابع هیستوگرام به‌تنهایی شرط‌های (۲) و (۳) را برآورده نمی‌کند. برای برآورده کردن شرط‌های (۲)و (۳) باید تعداد تکرار برای هر عدد را بر مجموع کل تکرارهای به‌دست آمده برای آن عدد مشخص تقسیم کرد (معادله ۴). در این‌صورت شرط‌های

(۲) و (۳) نیز برآورده می‌شوند. با این کار می‌توان منحنی تابع چگالی احتمال (شکل ۴) را با استفاده هیستوگرام (شکل ۳) به‌دست آورد.

$$f_x(x_i) = \frac{f_x(x_i)}{\sum_{j=0}^{255} f_x(x_j)} \qquad (۴)$$

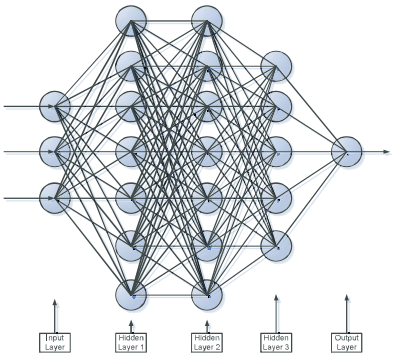


(شکل ۴): منحنی تابع چگالی احتمال

تا این مرحله تابع چگالی احتمال مربوط به تعداد تکرار اعداد ساخته شده در یک رشته به‌دست آورده می‌شود. اما با توجه به اینکه تابع چگالی احتمال به‌دست آمده برای هر رشته، شکلی متفاوت از رشته‌های دیگر دارد، برای کدکردن این رشته‌ها با الگوریتم هافمن باید تابع چگالی احتمال نیز همراه با رشته‌اطّلاعات مربوطه ارسال شود. یکی از روش‌هایی که می‌توان برای ارسال تابع چگالی احتمال استفاده کرد، این است که بر روی آن نموداری افراز کرده و اطّلاعات آن نمودار را به همراه رشته‌اطّلاعات ارسال کرد. اکنون این سؤال پیش می‌آید که با توجه به تغییرات سریع و غیر خطی تابع چگالی احتمال (شکل ۴) چه نموداری می‌توان بر روی آن افراز کرد به وجهی که کمترین خطا را با تابع چگالی احتمال مورد نظر داشته باشد. برای این کار می‌توان از شبکهٔ عصبی مصنوعی کمک گرفت که در بخش چهار به‌صورت خلاصه به توضیح آن می‌پردازیم.

۴- شبکهٔ عصبی مصنوعی

شبکهٔ عصبی مصنوعی از یک‌سری لایه‌ها و گره‌ها تشکیل می‌شود. هر شبکه به‌الزام دارای لایهٔ ورودی و خروجی است (Vemuri, 1988, Gardner, et al., 1998). در هر شبکه می‌توان به‌دلخواه تعدادی لایهٔ فرعی اضافه کرد. هر لایه شامل تعداد دلخواهی گره است که تحت عنوان نرون‌ها شناخته می‌شوند. ارتباط بین هر لایه توسط اتصال‌هایی که به نرون‌ها متصل می‌شوند، برقرار می‌شود. به این اتصال‌ها اصطلاحاً وزن‌های شبکه گفته می‌شود. در شکل ۵ نمونه‌ای از ساختاری یک شبکهٔ عصبی با سه ورودی و سه لایهٔ فرعی نشان داده شده است. لازم به ذکراست که علاوه‌بر وزن‌ها، هر نرون متأثر از کمیت مستقل دیگری با نام بایاس نیز است. نحوهٔ تأثیرپذیری مقادیر وزن‌ها و بایاس‌ها در نرون‌های هر لایه، به نرون‌های لایهٔ بعدی توسط تابعی تحت عنوان تابع انتقال، مشخص می‌شود. تابع انتقال در هر لایه می‌تواند خطّی و یا غیر خطّی باشد که توسط کاربر تعیین می‌شود. علاوه‌بر این، وزن‌ها و بایاس‌های مربوط به هر نرون، عامل مهم و حائز اهمیت در شبکه هستند که مقادیر آنها در نتایج خروجی شبکهٔ عصبی تأثیر گذار است.



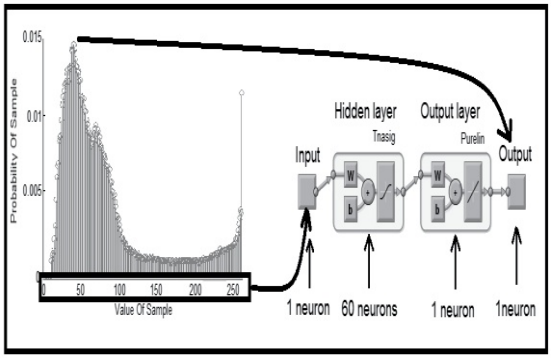
(شکل ۵): نمایش ساختار یک شبکه ی عصبی چند لایه

لذا در روش‌های متعددی که جهت بهبود دقت شبکه‌های عصبی به‌کار رفته است، سعی بر انتخاب مقادیر مناسب این وزن‌ها و بایاس‌ها، به‌گونه‌ای است که کم‌ترین خطای شبکه را دربرداشته باشد. به این عمل به‌اصطلاح آموزش شبکه گفته می‌شود.

۴-۱- نحوهٔ استفاده از شبکهٔ عصبی برای توصیف تابع چگالی احتمال

در این قسمت هدف افراز کردن تابعی بر منحنی چگالی احتمال (شکل ۴) با استفاده از شبکهٔ عصبی است. (Cobourn, et al., 2009). برای این کارابتدا یک شبکهٔ عصبی با یک لایهٔ پنهان آموزش‌ندیده، ایجاد می‌شود. سپس تمام وزن‌ها و بایاس‌های شبکه به‌عنوان متغیرهای بهینه‌سازی معرفی می‌شوند. ورودی شبکه عصبی مقادیر محور افقی تابع چگالی احتمال (اعداد صحیح بین ۰ تا ۲۵۵) و خروجی شبکهٔ عصبی مقادیر محور عمودی تابع چگالی احتمال (احتمال رخداد نمونه‌ها) درنظرگرفته می‌شوند. تابع

هدف در شبکهٔ عصبی به‌کار گرفته شده، بر اساس میانگین مربعات خطا بین احتمال رخداد نمونه‌های تابع چگالی احتمال و مقادیر محاسبه شده با توجه به وزن‌ها و بایاس‌های شبکهٔ عصبی معرفی می‌شود. بنابراین در هر مرحله از بهینه‌سازی، مقادیر وزن‌ها و بایاس‌های شبکه، به‌گونه‌ای به روز می‌شوند که خطای شبکه کاهش یابد. در این مقاله برای افرازکردن تابعی بر نمودار چگالی احتمال از شبکهٔ عصبی پرسپترون با یک لایهٔ پنهان که شصت نرون دارد، استفاده شده است. تابع انتقال نرون‌های لایهٔ پنهان tansig و تابع انتقال لایهٔ خروجی purelin است. همچنین لایهٔ ورودی و خروجی، تک‌نرونی هستند و تمامی نرون‌های هر سه لایه دارای بایاس هستند (شکل ۶).



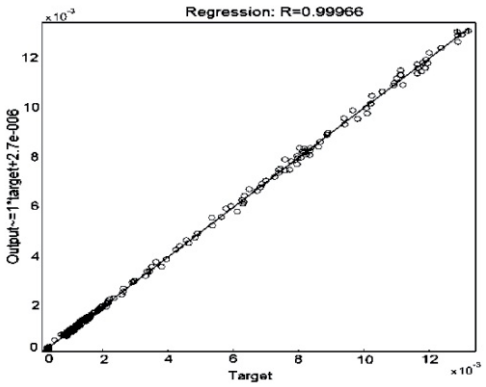
(شکل۶): ساختارشبکه استفاده شده در این مقاله با فرض ۶۰ نرون در لایه پنهان شبکه

همان‌طور که در شکل (۶) نشان داده شده است ورودی شبکهٔ عصبی مقادیر محور افقی و تابع هدف شبکه عصبی مقادیر عمودی متناظر با مقادیر افقی است. وزن‌های شبکه تا زمانی که خطا از 10^{-10} کمتر شود، به‌روز می‌شوند. باید به این نکته اشاره کرد که نوع تابع انتقال با روش Leave-One-Out (Fukunaga, et al., 1989, Stone, 1974) پس از امتحان بر روی چندین تابع چگالی احتمال انتخاب شده است. در این روش هدف ما در آموزش‌دادن شبکهٔ عصبی برخلاف آموزش‌های معمول است. به این معنی که در آموزش‌های متداول، درصدی از اطّلاعات به‌عنوان داده‌های آزمون کنار گذاشته می‌شوند و در انتهای هر بار آموزش و به‌روزشدن وزن‌ها و بایاس‌ها، داده‌های آزمون به‌عنوان ورودی به شبکهٔ عصبی وارد می‌شوند و خروجی‌های به‌دست آمده با مقادیر صحیح مقایسه می‌شوند. اگر خطا از حد معینی کم‌تر باشد، وزن‌ها و بایاس‌های به‌روزشده، به‌عنوان وزن‌ها و بایاس‌های شبکه، تثبیت می‌شوند. ولی اگر خطا بیشتر از حدّ مشخص شده باشد، آموزش دوباره تکرار می‌شود. این کار از Over Learning یا به‌اصطلاح آموزش بیش از حد جلوگیری می‌کند. اما در این مقاله برخلاف روش‌های معمول، سعی بر این است که شبکه به‌طور کامل بر مقادیر احتمال رخداد نمونه‌ها منطبق شود و به‌اصطلاح Overlearning رخ دهد. برای اینکه بتوان به این هدف رسید، تمامی مقادیر احتمال رخداد نمونه‌ها به‌عنوان نقاط آموزشی به شبکه عصبی وارد می‌شوند و درنتیجه پس از چندین‌بار به‌روزشدن مقادیر وزن‌ها و بایاس‌ها، مقادیر تابع به‌دست آمده با استفاده از شبکهٔ عصبی تا حدّ زیادی به منحنی چگالی احتمال نزدیک می‌شود.

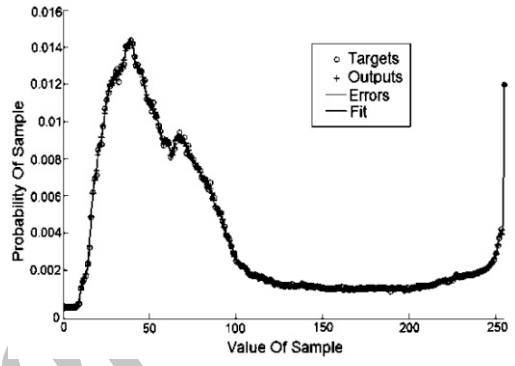
۴-۲- پیاده سازی شبکه عصبی برای تابع چگالی احتمال

به‌دلیل متغیر با زمان بودن منحنی‌های چگالی احتمال و همچنین به‌دلیل تغییرات زیاد در این منحنی‌ها نمی‌توان توابعی را با استفاده از روش‌های کلاسیک برای افرازکردن بر روی این منحنی‌ها به‌دست آورد. بنابراین در این مقاله از شبکهٔ عصبی برای افرازکردن تابعی بر روی منحنی چگالی احتمال استفاده شده است. این امر باعث می‌شود که مقادیر تابع محاسبه شده ذخیره شود و از آن برای حافظه‌دار کردن توابع چگالی احتمال استفاده شود. برای مثال اگر شبکهٔ عصبی ساخته شده با شصت نرون در لایهٔ پنهان را که دارای ۱۲۰ وزن و ۶۱ بایاس باشد[1] برای رشته‌اطّلاعاتی که دارای منحنی چگالی احتمالی شبیه به شکل(۴) است آموزش دهیم، خروجی در بهترین حالت با فرض (خطای میانگین مربعی کمتر از 10^{-10}) در شکل (۷) و شکل (۸) نشان داده شده است. در شکل (۷) تابع هدف و مقادیر خروجی به‌دست آمده با استفاده از شبکه عصبی با یکدیگر رسم شده‌اند. همان‌طور که در این شکل دیده می‌شود مقادیر خروجی با خطای بسیارکمی بر روی مقادیر تابع هدف منطبق شده‌اند.

[1] تعداد ۶۰ بایاس برای لایه پنهان و یک بایاس برای لایه خروجی



(شکل ۷): رسم همزمان مقادیر تابع هدف همراه با مقادیر خروجی به دست آمده با شبکه عصبی



(شکل ۸): رسم میزان پراکندگی مقادیر هدف از مقادیر خروجی

شکل (۸) مقدار رگرسیون[1] را نشان می‌دهد که دلیلی بر صحت میزان خطای ناچیز داده‌های خروجی است. البته باید به این نکته توجه کرد که در مواقعی که نمودار با شیب تند نزول می‌کند و مقادیر به سمت صفر میل می‌کنند، ممکن است که تابع تقریب زده مقادیری کمتر از صفر را برای این نقاط تقریب بزنند؛ که شرط (۱) را نقض می‌کند .

در این مواقع مقادیر منفی با کم‌ترین مقدار مثبت تابع تخمین زده شده جایگزین می‌شوند؛ زیرا نقاطی که با مقادیر منفی تخمین زده شده‌اند، دارای مقادیر ناچیزی دارند و لذا می‌توان آنها را با کم‌ترین مقدار مثبت تابع تخمین زده شده جایگزین کرد.

۵- مفهوم حافظه‌دارکردن رشته‌اطّلاعات

با توجه به مطالب گفته‌شده در بخش‌های ۴-۱ و ۴-۲ می‌توان گفت که تابع چگالی احتمال را با استفاده از شبکهٔ عصبی مصنوعی می‌توان توصیف کرد. این بدان معناست که می‌توان تابع چگالی احتمال را با وزن‌ها و بایاس‌های به‌دست آمده از آموزش شبکهٔ عصبی توصیف و همچنین رفتار تابع چگالی احتمال را با استفاده از مجموعهٔ وزن‌ها و بایاس‌های شبکهٔ عصبی می‌توان ذخیره‌سازی کرد. برای ذخیرهٔ هرکدام از وزن‌ها و بایاس‌ها دو بایت نیاز است. در ادامه مزیت حافظه‌دارکردن رشته‌اطّلاعات بیان می‌شود.

۵-۱- مزیت حافظه‌دارکردن رشته‌اطّلاعات

با استفاده از حافظه‌دارکردن رشته‌اطّلاعات می‌توان الگوریتم هافمن را چندین‌بار بر روی رشته‌اطّلاعات پیاده‌سازی کرد. بدین صورت که قبل از هر بار ارسال اطّلاعات، تابع چگالی احتمال با استفاده از شبکهٔ عصبی تقریب زده می‌شود و سپس الگوریتم هافمن با استفاده از مقادیر خروجی شبکهٔ عصبی که برای تابع چگالی احتمال به‌دست آمده است، اجرا می‌شود. وزن‌ها و بایاس‌های به‌روزشده به‌وسیلهٔ شبکهٔ عصبی، در انتها به رشته‌اطّلاعات اضافه می‌شوند. با تکرار این روش می‌توان حجم رشته‌اطّلاعات ارسالی را بسیار کم‌کرد؛ اما تعداد دفعات تکرار این الگوریتم برای فشرده سازی رشته‌اطّلاعات رابطه مستقیم با طول رشته‌اطّلاعات دارد. این بدان معناست که اگر بخواهیم چندبار این الگوریتم را اجرا کنیم، با توجه به اینکه در هر بار اجراکردن این الگوریتم، باید تابع چگالی احتمال توسط شبکهٔ عصبی تخمین زده شده و تعداد بیتی که بیان‌گر پارامترهای شبکهٔ عصبی به‌کار گرفته شده است با استفاده از معادلهٔ (۶) (که در بخش ۶-۲ آورده‌شده است) محاسبه‌شده و به انتهای رشته‌اطّلاعات اضافه می‌شود، که این امر موجب افزایش طول رشته‌اطّلاعات می‌شود. در اینصورت اگر طول رشته‌اطّلاعات اولیه کوتاه باشد، نمی‌توان این الگوریتم را چندین‌بار برای آن‌رشته به‌کار برد؛ زیرا تعداد بیت استفاده شده برای معرفی تابع چگالی احتمال به مراتب بیشتر از طول رشته‌اطّلاعات است. بنابراین تعداد تکرار این الگوریتم محدود است و رابطهٔ مستقیم با طول رشته دارد یعنی برای رشته‌هایی با طول زیاد تعداد تکرار می‌تواند بیشتر از تعداد تکرار برای رشته‌هایی با طول کوتاه‌تر باشد. در ادامه الگوریتم فشرده‌سازی گام به گام توضیح داده می‌شود.

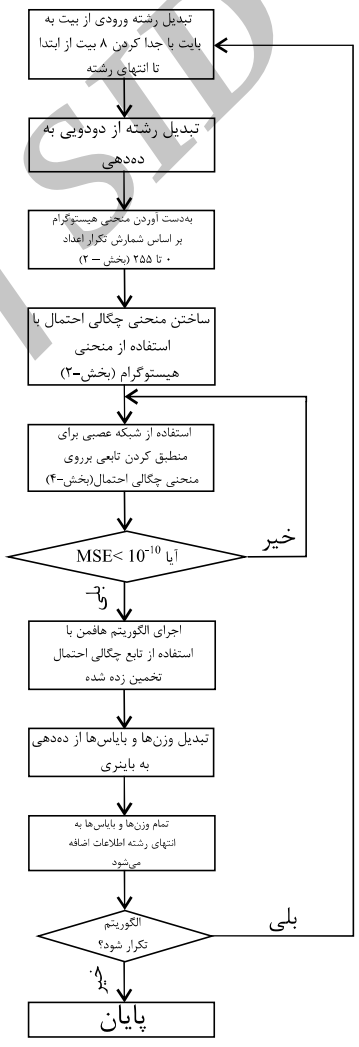
[1] میزان پراکندگی مقادیر هدف از مقادیر خروجی

۶-توضیح الگوریتم فشرده‌سازی

۶-۱ نحوهٔ کدکردن

شکل (۹) نمودار جریانی مربوط به الگوریتم فشرده‌سازی را نشان می‌دهد. مراحل انجام این الگوریتم به‌شرح ذیل است:

- مرحلهٔ ۱: ابتدا رشته‌اطّلاعات از مبنای دودویی به مبنای ده‌دهی تبدیل می‌شود. (بخش ۳)
- مرحلهٔ ۲: منحنی هیستوگرام بر اساس تعداد تکرار اعداد ۰ تا ۲۵۵ در رشته‌اطّلاعات ساخته شده در مرحلهٔ ۱ به‌دست آورده می‌شود. (بخش ۳)
- مرحلهٔ ۳: به‌دست آوردن منحنی چگالی احتمال با استفاده از منحنی هیستوگرام، با برآورده کردن شرط‌های (۱) و (۲) و (۳). (بخش ۳)
- مرحلهٔ ۴: استفاده از شبکهٔ عصبی برای افرازکردن تابعی بر روی منحنی چگالی احتمال. (بخش ۴)
- مرحلهٔ ۵: اجرای الگوریتم هافمن با استفاده از تابع چگالی احتمال تخمین زده شده در مرحلهٔ ۴.
- مرحلهٔ ۶: تبدیل وزن‌ها و بایاس‌ها از مبنای ده‌دهی به مبنای دودویی و اضافه‌کردن آنها به انتهای رشته‌اطّلاعات فشرده‌شده جهت ارسال.
- مرحلهٔ ۷: در صورت نیاز به تکرار الگوریتم جهت فشرده‌سازی بیشتر، مراحل ۱ تا ۶ دوباره تکرار شده و در انتها تعداد دفعات تکرار الگوریتم به همراه رشته‌اطّلاعات ارسال می‌شود.



(شکل ۹): رسم نمودار جریانی الگوریتم کدکردن

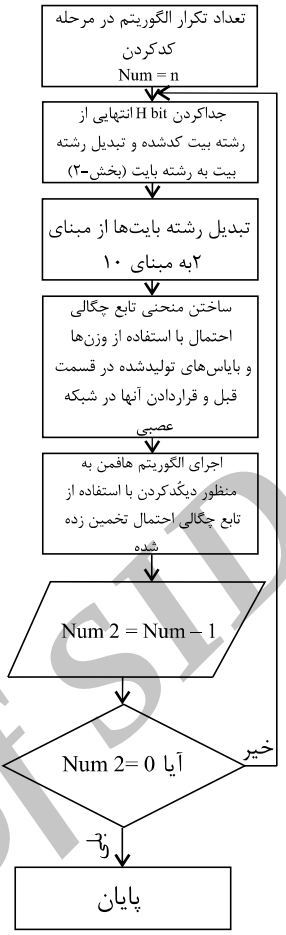
۶-۲ نحوهٔ کدگشایی کردن و بازیابی اطّلاعات

شکل (۱۰) **نمودار جریانی** الگوریتم بازیابی اطّلاعات را نشان می‌دهد. مراحل بازیابی اطّلاعات به شرح ذیل است:

- مرحلهٔ ۱: جداسازی H بیت به‌دست آمده از معادلهٔ (۶) از انتهای رشته‌اطّلاعات.

$$H = (\text{تعداد نرون در لایه پنهانی}+1) * \frac{2*\text{Byte}}{\text{نرون}} * \frac{8\ \text{bit}}{\text{Byte}} \quad (۶)$$

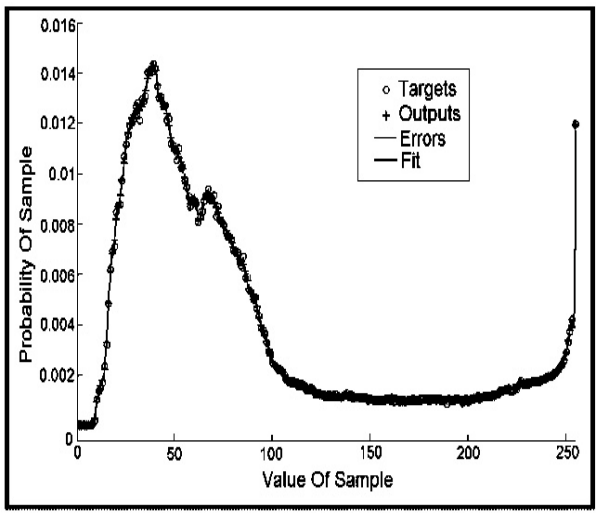
- مرحلهٔ ۲: H بیت جدا شده را از مبنای دودویی به مبنای ده‌دهی تبدیل کرده و درنتیجه وزن‌ها و بایاس‌های آخرین تابع چگالی احتمال را به‌دست آوریم.
- مرحله ۳: با استفاده از وزن‌ها و بایاس‌های تولیدشده از مرحلهٔ ۲ و با استفاده از شبکهٔ عصبی تابع چگالی احتمال دوباره تولید می‌شود.
- مرحلهٔ ۴: با توجه به تابع چگالی احتمال به‌دست آمده رشته‌اطّلاعات با استفاده از الگوریتم هافمن کدگشایی می‌شود.
- مرحلهٔ ۵: این الگوریتم به تعداد دفعاتی که در فرستنده اجرا شده در گیرنده نیز باید اجرا شود. برای اجرای دوباره الگوریتم کدگشایی، باید به مرحلهٔ ۱ بازگردیم. زمانی که تعداد دفعات اجرای الگوریتم کدگشایی برابر با تعداد دفعات اجرای الگوریتم در زمان کدگشایی شود، الگوریتم کدگشایی خاتمه می‌یابد.



(شکل ۱۰): رسم نمودار جریانی الگوریتم کدگشایی

۷- استفاده از الگوریتم ارائه‌شده برای فشرده‌سازی اطّلاعات

به‌عنوان مثال، شکل‌های (۲) و (۳) و (۴) با انجام مراحل ۱، ۲، ۳ و ۴ از بخش ۶-۱ به‌دست آمده‌اند. در مرحلهٔ بعد الگوریتم هافمن با توجه به تابع چگالی احتمال به‌دست آمده از شبکهٔ عصبی اجرا می‌شود و رشته‌اطّلاعات را فشرده‌سازی می‌کند. در این مثال، الگوریتم روش پیشنهادی چهار مرتبه اجرا شده است و نتایج آن در جدول (۲) و شکل های ۱۴ و۱۳، ۱۲، ۱۱ آورده شده است. همان‌طور که در منحنی‌های۱۴ و۱۳،۱۲ دیده می‌شود مقادیر منفی در تخمین تابع نیز وجود دارند که همانطور که در بخش ۴-۲ گفته شد این مقادیر با کوچک‌ترین مقدار مثبت تابع چگالی احتمال جایگزین شده و با تابع چگالی احتمال ساخته شده، الگوریتم هافمن اجرا می‌شود.



(شکل ۱۱): تابع چگالی احتمال تخمین زده شده در اولین اجراالگوریتم

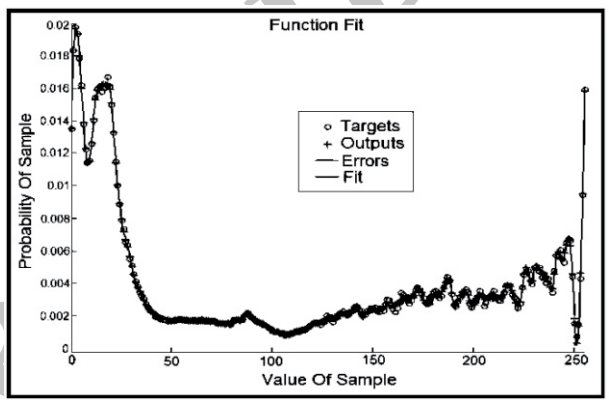
۸- ارزیابی روش پیشنهادی

قبل از ارزیابی الگوریتم پیشنهادی به‌صورت اجمالی در مورد ISCAS benchmark circuits توضیحی ارائه داده و سپس به ارزیابی روش پیشنهادی می‌پردازیم. ISCAS benchmark circuits شامل مداراتی هستند که با استفاده از درگاه منطقی شروع به تولید رشته‌بیت‌هایی با طول مشخص می کنند. هر یک از این مدارات ساختاری متفاوت داشته و رشته‌بیت‌های تولیدی آنها نیز دارای طول متفاوت از دیگری است. این مدارات به‌صورت نرم‌افزار برای FPGA ها نیز نوشته شده است. در این مقاله فایل چند سری از مداراتی که در مرجع (Chandra, et al., 2003) مورد استفاده قرار گرفته‌اند، اجرا شده و رشته‌بیت خروجی آنها توسط الگوریتم پیشنهادی فشرده‌سازی شده و با روش‌های FDR و Golomb از لحاظ درصد فشرده‌سازی مقایسه شده است. به‌منظور ارزیابی روش پیشنهادی دو آزمون با استفاده از ISCAS benchmark circuits ترتیب داده شده است. در آزمون اوّل از یک رشته‌بیت ساخته شده با استفاده از ATPG (Hamzaoglu, et al., 1998) و در آزمون دوم از رشته‌بیت ساخته‌شده برای ISCAS benchmark circuit از HITEC (Univ. of Illinois IGATE website) استفاده شده است. در آزمون اول درصد فشرده سازی بین روش پیشنهادی با روش‌های FDR (Chandra, et al., 2003) و Golomb مقایسه شده است و در آزمون دوم درصد فشرده‌سازی تنها بین روش پیشنهادی با روش FDR مقایسه شده است؛ زیرا روش

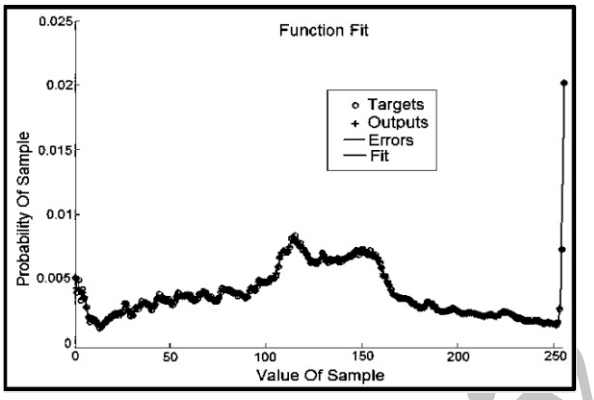
FDR دارای درصد فشرده‌سازی بیشتری نسبت به روش Golomb است و رشته‌اطّلاعات را بیشتر فشرده می‌کند. پس از شش مرتبه اجرای الگوریتم روش پیشنهادی، درصد فشرده‌سازی رشته‌اطّلاعات به‌ازای ۶ و۴ و۳ و۲ و۱ = M مرتبه اجرای الگوریتم در جدول (۳) گزارش شده است، که بهترین آنها با درصد فشرده‌سازی روش‌های FDR و Golomb مقایسه شده‌اند.

(جدول ۲): نمایش میزان فشردگی برای چهار حالت که در آنها دفعات تکرار الگوریتم متفاوت است

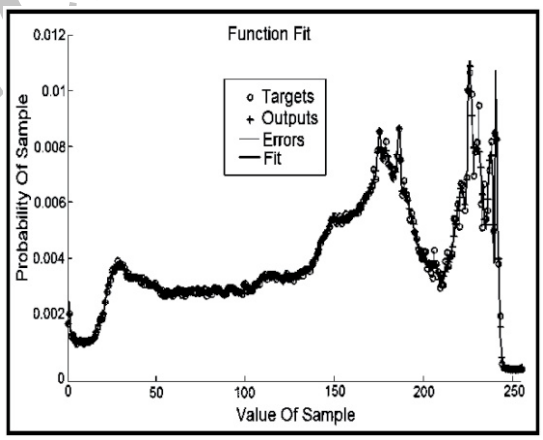
اندازهٔ رشته ورودی(bit) ۱۶۹۹۲۰۰	دفعات تکرار الگوریتم=M تعداد نرون لایه پنهان = ۶۰ وH= ۲۸۹۶ bit				
	M=۱	M=۲	M=۳	M=۴	M=۸
اندازهٔ رشته ساخته شده	۱۶۵۱۵۳۱	۲۴۷۴۵۲۳	۱۴۲۲۵۶	۷۰۵۹۵	۳۸۹۱
درصد فشرده سازی	٪۲/۸	٪۷۲/۰۷	٪۹۱/۶۲	٪۹۵/۸۴	٪۹۸/۵۳



(شکل۱۲): تابع چگالی احتمال تخمین زده شده در دومین اجرای الگوریتم



(شکل ۱۴): تابع چگالی احتمال تخمین زده شده در چهارمین اجرای الگوریتم



(شکل ۱۳): تابع چگالی احتمال تخمین زده شده در سومین اجرای الگوریتم

۸ -۱-آزمون اول

مقایسهٔ درصد فشرده‌سازی برای رشته‌بیت ساخته‌شده با استفاده از ATPG بین روش پیشنهادی با روش‌های FDR و Golomb:

همان‌طور که در جدول (۳) دیده می‌شود، بیش‌ترین اختلاف مقدار فشرده‌سازی بین روش پیشنهادی با روش FDR مربوط به رشتهٔ S38584 با مقدار ۳۳/۱٪ است و برای Golomb مربوط به رشته S38584 با مقدار ۳۸/۱۶٪ و کم‌ترین اختلاف مقدار فشرده‌سازی بین روش پیشنهادی با روش FDR مربوط به رشتهٔ S13207 با مقدار ۹/۵٪ است و برای Golomb مربوط به رشتهٔ S13207 با مقدار۱۲/۸۴٪ است.

(جدول ۳): مقایسه درصد فشرده‌سازی

اندازهٔ رشته کد شده (bit)	اندازهٔ رشته‌اطّلاعات (bit)	بهترین درصد فشرده سازی	درصد فشرده‌سازی با پیاده سازی الگوریتم معرفی شده در این مقاله با M بار تکرار الگوریتم تعداد نرون لایه پنهان = ۶۰ و H=۲۸۹۶ bit					درصد فشرده سازی FDR	درصد فشرده سازی Golomb	Circuit
			M=۶	M=۴	M=۳	M=۲	M=۱			
۴۱۷۴	۲۳۷۵۴	۸۲/۳۹	۸۲/۳۹	۸۷/۴۲	۸۰/۴۲	۷۲/۵۹	۵۶/۱۴	۶۱/۳۲	۵۳/۷۳	S5378
۴۵۵۹	۳۹۲۷۳	۸۸/۳۹	۸۸/۳۹	۸۶/۶۷	۸۷/۳۵	۸۱/۸۸	۶۱/۴۸	۶۰/۶۳	۵۹/۸۵	S9234
۴۶۷۵	۱۶۵۲۰۰	۹۷/۱۷	۹۷/۱۷	۹۶/۲۵	۸۳/۶۳	۸۹/۷۲	۶۸/۱۷	۸۷/۶۷	۸۴/۳۳	S13207
۴۲۰۳	۷۶۹۸۶	۹۴/۵۴	۹۴/۵۴	۹۳/۰۲	۸۹/۱۳	۶۷/۷۵	۳۵/۳۴	۷۱/۹۵	۶۶/۵۵	S15850
۶۲۲۷	۱۶۴۷۳۶	۹۶/۲۲	۹۴/۹۷	۹۶/۲۲	۹۴/۱۴	۷۲/۶۷	۵۳/۷۹	۶۵/۳۵	۵۸/۰۸	S38417
۴۴۴۰	۱۹۹۱۰۴	۹۷/۷۷	۹۷/۷۷	۹۵/۲۴	۸۵/۵۳	۸۹/۵۴	۶۸/۸۶	۶۴/۶۷	۵۹/۶۱	S38584

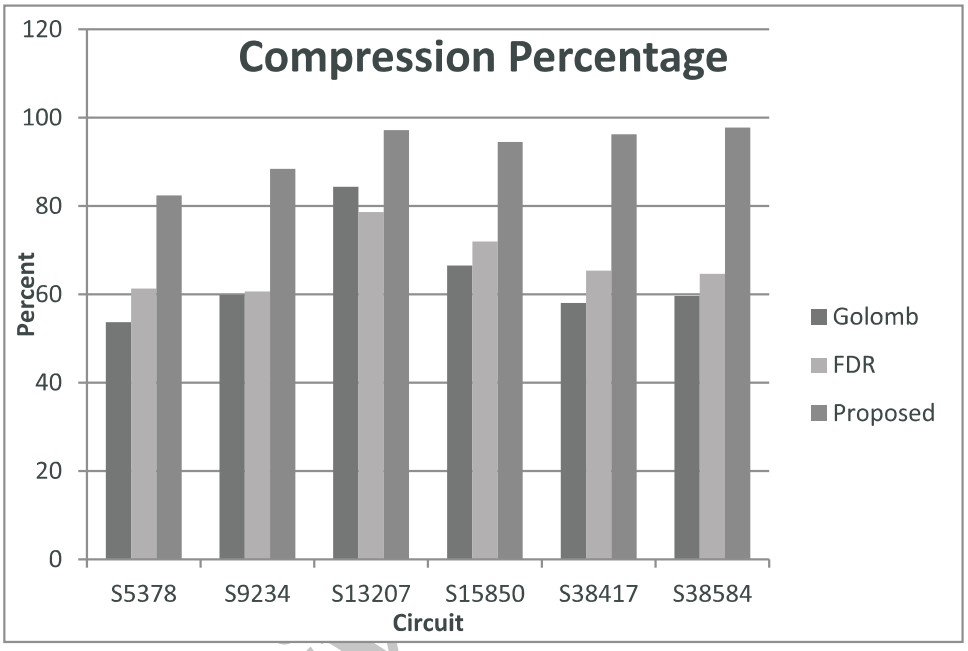
روش پیشنهادی برای رشته‌هایی با طول کم (مانند S953) قادر به فشرده‌سازی در حد بالایی نیست، زیرا افزایش تکرارهای الگوریتم پیشنهادی، به علت اضافه‌شدن اطّلاعات مربوط به تابع چگالی احتمال در انتهای رشتهٔ کدشده، باعث کم‌شدن درصد فشرده‌سازی می‌شود. اما برای رشته‌هایی با طول زیاد (مانند S35932)، درصد فشرده‌سازی با افزایش تکرار الگوریتم پیشنهادی افزایش می‌یابد. دلیل این امر آن است که داده‌های مربوط به شبکهٔ عصبی، برای رشته‌های با طول زیاد نسبت به طول رشته ارسالی ناچیز است. تعداد تکرارهای الگوریتم پیشنهادی باید محدود باشد، زیرا پس از چندین تکرار درصد فشرده‌سازی به‌دلیل افزایش داده‌های مربوط به ذخیره‌سازی شبکه عصبی، کاهش پیدا می‌کند. اما میزان تکرار برای رسیدن به بهترین درصد فشرده‌سازی در رشته‌هایی با طول زیاد نسبت به رشته‌هایی با طول پایین می‌تواند بیشتر باشد. برای روشن‌شدن مطلب و وضوح بیشتر شکل(۱۵)، نمودار میله‌ای مربوط به جدول (۳) را جهت مقایسهٔ بهتر نشان می‌دهد.

۸-۲- آزمون دوم

مقایسهٔ درصد فشرده‌سازی رشته‌بیت ساخته‌شده برای ISCAS benchmark circuits با استفاده از HITEC بین روش پیشنهادی با روش FDR :

در این آزمایش با توجه به اینکه طول رشته‌اطّلاعات S953 بسیار کم است، روش پیشنهادی قادر به فشرده‌سازی با درصد بالا را نخواهد داشت که علت آن در بخش ۸-۱ توضیح داده شد. در بهترین حالت درصد فشرده‌سازی برای

رشته S953 معادل با ۶۸/۴۷٪ است، که برای شبکهٔ عصبی با سه نرون در لایهٔ پنهان و چهار تکرار به‌دست آمده است. ولی برای رشته S13207 با تغییر تعداد نرون لایهٔ پنهان و تعداد تکرار الگوریتم نتایج جدول (۵) به‌دست می‌آید. همان‌طور که در جدول (۵) مشاهده می‌شود به‌ازای مقادیر M=14 و N=4 درصد فشرده‌سازی کاهش یافته است. دلیل این امر آن است که الگوریتم پیشنهادی تا میزان خاصی قادر به فشرده‌سازی رشته‌بیت‌های اعمال شده است، بنابراین اگر پس از رسیدن به بهترین درصد فشرده‌سازی، دوباره الگوریتم پیشنهادی اجرا شود و یا لایه‌های پنهان شبکهٔ عصبی به‌کار رفته در الگوریتم پیشنهادی افزایش یابد، به‌دلیل افزایش اطّلاعات مربوط به وزن‌های شبکه عصبی، درصد فشرده‌سازی رشته‌بیت اطّلاعات ارسالی کاهش می‌یابد.



(شکل ۱۵): نمایش میله ای مقایسه درصد فشرده سازی برای جدول (۳)

زیرا اجرای مجدد شبکهٔ عصبی و یا افزایش نرون‌ها فقط بیت‌های ارسالی به‌منظور بازسازی شبکهٔ عصبی را بیشتر می‌کند ولی میزان فشرده‌سازی رشته‌بیت ورودی افزایش نمی‌یابد. اما نتایج روش پیشنهادی برای سایر رشته‌ها به‌مراتب خیلی بهتر از روش FDR بوده است. در بهترین حالت درصد فشرده‌سازی به‌میزان ۱۶/۰۵٪ برای رشته S5378 بهبود یافته است. برای رشته‌های دیگر نیز درصد فشرده‌سازی در جدول(۴) نشان داده شده است. شکل(۱۶) نمودار میله‌ای مربوط به جدول (۴) را جهت مقایسه بهتر و وضوح بیش‌تر نشان می‌دهد.

۸-۳- مقایسهٔ درصد فشرده‌سازی با بهترین مقدار به‌دست آمده از scan vector= ۳۰

در (Chandra, et al., 2003) ، ۳۲ مرحلهٔ فشرده‌سازی روی رشته‌بیتی با طول ۳۶۲۹۲۲ بیت با روش‌های FDR و Golomb صورت پذیرفته که بهترین درصد فشرده‌سازی در scan vector =۳۰ به‌دست آمده است. مقادیردرصد فشرده‌سازی این آزمایش در جدول (۶) آورده شده است. با استفاده از الگوریتم پیشنهادی ۰/۵۴٪ درصد فشرده‌سازی را در مقایسه با روش FDR و۱/۴۲٪ درصد فشرده‌سازی را درمقایسه باروش Golomb افزایش داده‌ایم. لازم به ذکر است همان‌گونه که در جدول (۴) مشاهده می‌شود، درصد فشرده‌سازی به‌طور عمومی بالای ۹۹٪ است و لذا افزایش ۰/۵۴٪ فشرده‌سازی نیز قابل قبول است. درصد فشرده‌سازی با استفاده از معادله(۷)حساب می‌شود.

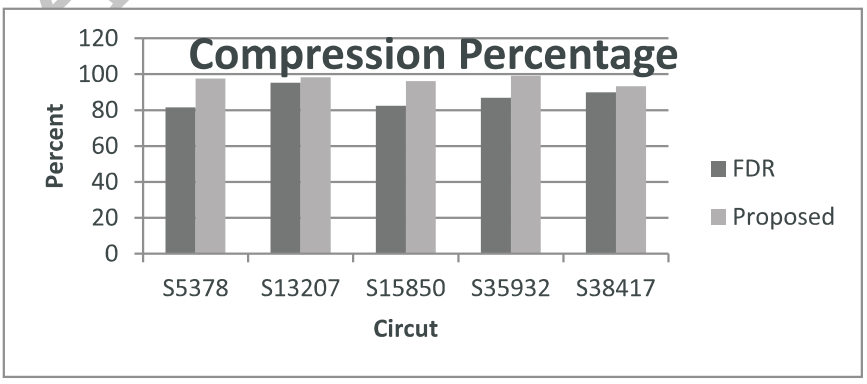
$$\text{درصد فشرده‌سازی} = \frac{\text{طول رشته کد شده} - \text{رشته طول ورودی}}{\text{رشته ورودی طول}} * 100 \qquad (۷)$$

۹- نتیجه‌گیری

در این مقاله روشی جدید برای کدکردن رشته‌اطّلاعات معرفی شد. این نوع کدگذاری بر روی رشته‌بیت‌های ساخته شده با استفاده از ISCAS benchmark circuits اعمال شد. نتایج به‌دست آمده از میزان فشردگی با دو کد FDR و Golomb مقایسه شد، که بیش‌ترین اختلاف مقدار فشرده‌سازی بین روش پیشنهادی با روش FDR معادل با ۳۳/۱٪ و برای روش Golomb معادل با ۳۸/۱۶٪ به دست آمد و کمترین اختلاف مقدار فشرده سازی بین روش پیشنهادی با روش FDR معادل با ۹/۵٪ و برای روش Golomb معادل با ۱۲/۸۴٪ به‌دست آمد. البته این نوع کدگذاری برای رشته‌هایی با طول پایین، مناسب نیست. به‌دلیل اینکه پس از هر بار اجرای الگوریتم پیشنهادی تعدادی بیت که برای فرستادن تابع چگالی احتمال مورد نیاز است به انتهای رشتهٔ کدشده اضافه می‌شوند. همچنین برای رشته‌اطّلاعات مورد نظر با طول ۳۶۲۹۲۲ بیت، بهترین درصد فشرده‌سازی به‌وسیلهٔ روش پیشنهادی که با کم‌کردن نرون‌های لایهٔ پنهان شبکهٔ عصبی به منظور کاهش تعداد بیت‌های اضافه شده به انتهای رشتهٔ کدشده و افزایش تعداد اجرای الگوریتم نسبت به بهترین درصد فشرده‌سازی با دو روش FDR و Golomb به‌دست آمده است.

(جدول ۴): مقایسه درصد فشرده سازی FDR با الگوریتم این مقاله با استفاده از HITEC

Circuit	بهترین درصد فشرده سازی FDR	درصد فشرده‌سازی با پیاده‌سازی الگوریتم معرفی‌شده در این مقاله با M بار تکرار الگوریتم تعداد نرون لایهٔ پنهان[1] =۶۰ و H=۲۸۹۶ bit					بهترین درصد فشرده سازی	اندازهٔ رشته اطّلاعات (bit)	اندازهٔ رشته کدشده (bit)
		M=۱	M=۲	M=۳	M=۴	M=۶			
S953	۷۵/۹۴	--	--	--	--	--	--	۱۱۶۸	--
S5378	۸۱/۴۵	۵۷/۱۹	۸۴/۶۷	۹۰/۷۷	۹۵/۵۳	۹۷/۵۰	۹۷/۵۰	۱۶۹۹۹۵	۴۲۴۹
S13207	۹۵/۱۷	۶۵/۴۲	۸۲/۹۴	۸۶/۵۲	۸۶/۱۲	۹۸/۲۳	۹۸/۲۳	۴۲۲۸۴	۴۵۵۴
S15850	۸۲/۴۱	۶۷/۹۱	۸۹/۰۴	۹۰/۷۷	۹۴/۱۲	۹۶/۲۰	۹۶/۲۰	۱۴۷۰۷۰	۵۵۸۸
S35932	۸۶/۸۴	۶۸/۴۱	۷۰/۵۰	۹۳/۷۷	۹۶/۱۶	۹۸/۷۶	۹۸/۷۶	۴۳۰۳۵۸	۵۳۳۶
S38417	۸۹/۸۶	۶۹/۷۷	۸۰/۴۳	۹۰/۱۱	۹۳/۳۸	۹۱/۰۴	۹۳/۳۸	۲۲۶۲۴	۱۴۹۷



(شکل ۱۶): نمودار میله ای مقایسه درصد فشرده‌سازی برای جدول (۴)

[1] Hidden

(جدول ۵): محاسبهٔ درصد فشرده‌سازی برای غلبه بر کد FDR با تغییر تعداد نرون‌های لایهٔ پنهان و تعداد تکرار الگوریتم

Circuit	بهترین درصد فشرده سازی FDR	درصد فشرده‌سازی با پیاده‌سازی الگوریتم معرفی‌شده در این مقاله با M بار تکرار الگوریتم و دارای N نرون در لایهٔ پنهان شبکه				بهترین درصد فشرده سازی	اندازهٔ رشته اطّلاعات (bit)	اندازهٔ رشته کد شده (bit)
		M = ۱۰ N=۳	M = ۱۲ N=۳	M = ۱۴ N=۳	M = ۱۴ N=۴			
S13207	۹۵/۱۷	۹۵/۱۸	۹۸/۲۷	۹۹/۱۱	۸۴/۹۷	۹۹/۱۱	۴۲۲۸۴	۳۷۶

(جدول ۶): مقایسه درصد فشرده‌سازی روش پیشنهادی با روش فشرده‌سازی در (Chakrabarty, et al., 2001)

اندازهٔ رشته کدشده (bit)	بهترین درصد فشرده‌سازی با پیاده‌سازی الگوریتم معرفی‌شده در این مقاله با M بار تکرار الگوریتم و دارای N نرون در لایهٔ پنهان شبکه M=۳ و N=۱۵	بهترین درصد فشرده‌سازی FDR	بهترین درصد فشرده‌سازی Golomb	اندازهٔ رشته‌اطّلاعات (bit)
۴۳۵	۹۹/۸۸	۹۹/۳۴	۹۸/۴۶	۳۶۲۹۲۲

۱۰ - منابع

Chakrabarty, K., Chandra., 2001. System-on-a-chip test data compression and decompression architectures based on golomb codes, IEEE Trans. Computer Aided Design, Vol. 20, pp. 355-368.

Chandra., Chakrabarty, K., 2003. Test data compress–ion and test resource partitioning for system on a chip using frequency-directed run-length (FDR) codes, IEEE Transactions on computer. Vol. 52.

Chandra., Chakrabarty, K., 2000. Test data compression for system-on-a-chip using golomb codes, Proc. IEEE VLSI Test Symp, pp. 113-120.

Cobourn, W.G., Dolcine, L., French, M., Hubbard, M.C., 2009. A Comparison of Nonlinear Regression and Neu–ral Network Models for Ground-Level Ozone Forecasting, J. Air & Waste Manage.

Fukunaga, K., Hummels, D.M., 1989. Leave-one-out procedure for nonparametric error estimate, IEEE transactions on pattern analysis and machine intelligence, Vol. 11, pp. 421-423.

Gardner, M.W., Dorling, S.R., 1998. Artificial Neural Networks (The Multilayer Perceptron), A Review of Applications in Atmospheric Sciences, Atmos, Environ.

Hamzaoglu, I., Patel, J.H., 1998. Test set compaction algorithms for combinational circuits, proc. int'l conf. computer-aided design, pp. 283-289.

Huffman, D.A., 1952. A method for the construction of minimum redundancy codes, Proc. IRE, Vol. 40, pp. 1098-l101.

Iyengar, V., Chakrabarty, K., Murray, B.T., 1999. Deterministic built-in self-testing of sequential circuits using pre computed test sets, J. Electronic Testing Theory and Applications (JETTA), Vol. 15, pp. 97-114.

Jas, A., Ghosh-Dastidar, J., Touba, N.A., 1999. Scan vector compression decompression using statistical coding, Proc. IEEE VLSI Test Symp, pp. 114-120.

Proakis, J.G., 1995. Digital Communications, Mcgraw Hill, Hardcover.

Shi, Y.Q., Sun, H., 2000. Image and Video Compression for Multimedia Engineering Fundamentals, Algorithms and Standards, CRC Press.

Stone, M., 1974. Cross-validatory choice and l assessment of statistical predictors, Journal of the Royal Statistical Societyz, Vol. 36, pp. 111-147.

Touba, N.A., Jas, A., 1998. Test vector decompression via cyclical scan chains and its application to testing core-based design, Proc. Int'1 Test Conf, pp. 458- 464.

Univ. of Illinois IGATE website :www.crh-c.uiuc.edu/IGATE.2000.

Vemuri, V., 1988. Artificial Neural Networks, Theoretical Concepts, Computer Society Press Technology Series, Computer Society of Institute of Electrical and Electronics Engineers, Washington, DC.

حسن فرسی در سال ۱۳۴۸ در شهرستان بیرجند به دنیا آمد. دورهٔ کارشناسی و کارشناسی ارشد خود را در رشته مهندسی برق – مخابرات در دانشگاه صنعتی شریف– تهران به‌ترتیب در سال‌های ۱۳۷۱ و ۱۳۷۴ به پایان رسانید. از سال ۱۳۷۱ به مدت یک‌سال در مرکز تحقیقات مخابرات ایران اشتغال داشت. در سال ۱۳۷۴ به عضویت هیئت علمی دانشگاه بیرجند درآمد و مدرک دکترای خود را از دانشگاه Surrey انگلستان در زمینهٔ پردازش سیگنال‌های دیجیتال اخذ کرد. در حال حاضر وی عضو هیئت علمی دانشگاه بیرجند است. زمینهٔ کاری ایشان پردازش سیگنال‌های دیجیتال، پردازش صحبت و تصویر هستند.

نشانی رایانامهٔ ایشان عبارت است از:

hfarsi@birjand.ac.ir

پوریا اعتضادی‌فر در سال ۱۳۶۸ در شهرستان نیشابور به دنیا آمد. دورهٔ کارشناسی خود را در دانشگاه بیرجند در رشته برق– مخابرات در سال ۱۳۹۰ به پایان رسانید. او در حال حاضر دانشجوی کارشناسی ارشد مهندسی برق – مخابرات در دانشگاه بیرجند است. زمینهٔ تحقیقاتی ایشان پردازش سیگنال‌های دیجیتال و تصویر است.

نشانی رایانامه‌های ایشان عبارت است از:

P.etezadifar@birjand.ac.ir
P.etezadifar@gmail.com