مجله علمی پژوهشی «علوم و فناوری‌های پدافند نوین»
سال ششم، شماره ۴، زمستان ۱۳۹۴؛ ص ۲۳۴- ۲۲۱

افزایش مقاومت ویدئو فشرده‌شده با H.264 نسبت به خطای کانال، با استفاده از شبکه عصبی و کد هافمن

پوریا اعتضادی‌فر[۱]، حسن فرسی[۲*]، علی ناصری[۳]

۱- دانشجوی دکتری، ۲- دانشیار، دانشگاه بیرجند ۳- دانشیار، دانشگاه جامع امام حسین (ع)

(دریافت: ۹۲/۱۱/۰۲، پذیرش: ۹۴/۰۵/۳۱)

چکیده

در حالاتی که کد کردن منبع و کانال به صورت جدا و بدون بازخورد نسبت به یکدیگر انجام می‌شوند، کدگذار منبع به دنبال افزایش بازدهی در حذف افزونگی داده‌های منبع اطلاعاتی می‌باشد، در حالی که کدگذار کانال قابلیت اطمینان داده‌های ارسالی در کانال را بالا می‌برد. می‌دانیم، ظرفیت کانال باعث محدود شدن داده‌های ارسالی می‌شود، بنابراین بسته به شرایط باید مصالحه‌ای بین کدگذار منبع و کانال صورت پذیرد. در این مقاله هدف افزایش کیفیت ویدئویی دریافتی با افزایش مقاومت آن نسبت به کانال مخابراتی در یک نرخ ارسال ثابت می‌باشد. به این معنا که بدون افزایش نرخ داده‌ی ارسالی مقاومت قاب‌های ارسالی در کانال را افزایش داده که منجر به بهبود و افزایش کیفیت منابع ویدئویی می‌شود. اساس کار بدین صورت است که با استفاده از شبکه عصبی هوشمند و کدهافمن در استاندارد H.264، اطلاعات ارسالی را به مقدار قابل توجهی فشرده می‌شود. سپس با توجه به مقدار فشرده‌سازی توسط روش پیشنهادی، اطلاعات فشرده شده دوباره با کدگذار کانال ثانویه که نرخ کدینگ آن وابسته مقدار فشرده‌سازی است، کد می‌شود. به این ترتیب روش پیشنهادی قادر است بدون افزایش حجم اطلاعات ارسالی برای هر قاب، نرخ کد کردن کانال و در نتیجه محافظت از اطلاعات را بالا برده و توانسته است قاب‌های ویدئویی را نسبت به خطاهای کانال مقاوم‌تر سازد. در نهایت نتایج به‌دست آمده را با چندین نرخ ارسال برای منبع و چندین SNR برای کانال با نتایج به‌دست آمده از روش‌های متداول مقایسه می‌شود.

کلید واژه‌ها: فشرده‌سازی ویدئویی، H.264، نرخ بیت متغیر، کدینگ کانال، تطبیق کدینگ کانال و منبع، شبکه عصبی مصنوعی.

Robustness of Compressed Video in H.264 Against Channel Using Neural Network with Huffman Coding

P. Etezadifar, H. Farsi*, A. Naseri
University of Birjand

(Received: 22/01/2014; Accepted: 22/08/2015)

Abstract

As source and channel coding are performed independently from each other without any feedback, source coding tries to remove redundancy of the information whereas channel coding tries to increase reliability of transmitted data. As known, channel capacity restricts volume of the transmitted data and so depending to conditions, it is required to have a tradeoff between source and channel coding. The aim of this paper is to improve the quality of the synthesized video by increasing the robustness against channel errors in fixed transmission rate. In other words, the robustness of the transmitted video frames increases without any increment in bit rate. This results in improvement in the quality of the synthesized video. In the proposed method, the transmitted information is considerably compressed using neural network with Huffman coding in H. 264. Then a secondary channel coding whose rate depends on the amount of the compression is applied on the compressed information. This causes that the proposed method is able to increase channel coding rate and therefore provides higher protection for the transmitted information and more robustness against channel errors. The obtained results by the proposed method are compared to the other methods for different source coding rates and SNRs.

Keywords: Video Compression, H. 264, Variable Bit Rate, Channel Coding, Adaptive Source and Channel Coding, Artificial Neural Network.

*Corresponding Author E-mail: Hfarsi@birjand.ac.ir

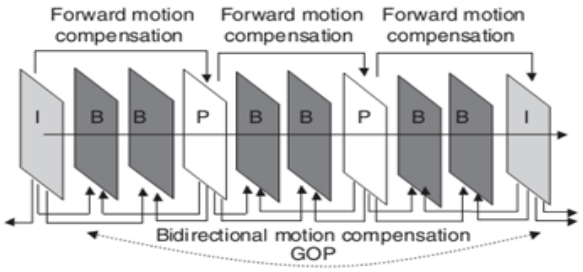
۱. مقدمه

در ارتباطات چندرسانه‌ای بی‌سیم، یکی از مهم‌ترین مسائل پیش‌رو، دستیابی به نرخ ارسال بالا تحت شرایط کانال‌های ارتباطی بی‌سیم است. یک فشرده‌سازی کارآمد می‌تواند داده ارسالی را کاهش دهد، اما فشرده‌سازی منبع را نسبت به خطا و نویز کانال حساس می‌کند. به منظور غلبه در مقابل تأثیر نویز، تداخل و محوشوندگی[1] نیاز به استفاده از کد کردن کانال و افزایش نرخ تصحیح آن‌ها می‌باشد. باید به این نکته اشاره نمود، کدگذارهای کانالی که برای تصحیح خطا به‌کار گرفته می‌شوند، اغلب منجر به افزایش پیچیدگی سامانه و بنابراین افزایش تأخیر شده و نیازمند پهنای باند می‌باشد [۱]. مطالعات و کارهای صورت پذیرفته، اغلب کد کردن منبع و کانال را به صورت مجزا و با استفاده از تئوری شنون بررسی می‌کنند. البته این نظریه برای طول کدهای نامحدود و برای ارسال نقطه به نقطه مطرح شده است. بنابراین کد کردن منبع برای افزایش میزان حساسیت و کاهش داده های بیان کننده اطلاعات به‌کار می‌رود، در حالی که کد کردن کانال با افزایش افزونگی به داده فشرده شده به کاهش نرخ خطای ایجاد شده بر روی داده فشرده شده می‌پردازد. بنابراین از دو عمل متضاد برای بهبود کیفیت داده‌های ارسالی استفاده می‌شود. آنچه گفته شد، دلیلی بر وجود مصالحه بین کد کردن کانال و منبع است [۱ و ۲]. یکی از کارآمدترین روش‌های استفاده شده، کدکردن کانال است که برای تشخیص و تصحیح خطا به‌کار گرفته می‌شود. در سال‌های اخیر، روش‌های متنوعی در رابطه ترکیب توأم کد کردن منبع و کانال[2] معرفی شده است. روش‌های متداولی برای کد کردن هم‌زمان منبع و کانال وجود دارد که عموماً بر اساس تخمین کانال صورت می‌گیرد [۶-۳]. روش به‌کار گرفته شده در این مقاله با روش‌های اشاره شده در دیگر گزارشات [۶-۴] متفاوت می‌باشد. اساس این روش اختصاص بیت‌های بیشتر به کدگذار کانال (و در واقع کاهش نرخ کد کانال) است. به این معنی که در یک نرخ ارسال ثابت بتوان نرخ کدگذار کانال را افزایش داد. با افزایش نرخ کدگذار کانال، فریم‌های ویدئویی ارسالی را می‌توان نسبت به کانال مقاوم‌تر ساخت. یکی از مشکلات روش‌های به‌کار گرفته شده در برخی گزارشات [۵-۲] این است که باید از کانال با خبر باشند و بسته به اعوجاج کانال نرخ کد منبع و کانال را تغییر دهند. اما روش پیشنهادی وابسته به کانال نبوده و قادر است بر روی هر کدگذار منبعی به صورت مستقل عمل کرده و کیفیت تصاویر دریافتی در گیرنده را افزایش دهد. در ادامه ابتدا به معرفی چند پارامتر استفاده شده در مقاله پرداخته می‌شود و در قسمت بعد به توضیح پیکره‌بندی مقاله اشاره‌ای خواهد شد. در این مقاله اساس سنجش قاب‌ها بر حسب بیشینه نسبت توان سیگنال به توان نویز PSNR[3] و BER[4] است که میزان خطای ایجاد شده توسط کانال بر روی اطلاعات را بیان می‌کند. این مقاله به صورت زیر ساماندهی شده است:

در بخش ۲، ابتدا روش کد کردن فریم‌های ویدئویی با استفاده از کدگذار H.264 بررسی می‌شود. در بخش ۳ به بررسی اجمالی کد کردن کانال و به صورت دقیق‌تر کد کردن کانال ریدسالامون پرداخته می‌شود. در بخش ۴ چگونگی محاسبه تابع چگالی احتمال با استفاده از یک رشته داده با طول متغیر بررسی خواهد شد و پس از آن در بخش ۵ به صورت اجمالی شبکه عصبی معرفی می‌شود. در بخش ۶ به مفهوم حافظه‌دار کردن رشته اطلاعات و سپس به مزیت حافظه‌دار کردن رشته‌ها پرداخته می‌شود. در بخش ۷ روش پیشنهاد شده در این مقاله معرفی شده و به صورت کامل به شرح و تفسیر آن پرداخته، در بخش پایانی به ارزیابی روش پیشنهادی و مقایسه داده‌های خروجی با روش معرفی شده در این حوزه پرداخته می‌شود و میزان بهره‌وری روش پیشنهادی نسبت به روش‌های دیگر به صورت تجربی مورد ارزیابی و بررسی قرار می‌گیرد.

۲. نحوه کدینگ فریم‌های ویدئویی

در این بخش ابتدا به صورت خلاصه به معرفی گروه تصاویر[5] پرداخته و در مرحله بعد نحوه کد و دیکد کردن فریم‌های ویدئویی با استفاده از H.264 را توضیح داده می‌شود. در استاندارد H.264، ۳ نوع تصویر معرفی می‌شوند که به اختصار با I، P و B نشان داده می‌شوند. از ترکیب این تصاویر با هم گروه تصاویر ساخته می‌شود. نمونه ای از ساختار گروه تصاویر در شکل (۱) نشان داده شده است.



شکل ۱. نمونه‌ای از ساختار GOP

به تصویری که با استفاده از اطلاعات موجود در خود آن تصویر کد می‌شود، تصویر I گفته می‌شود. بنابراین برای فشرده‌سازی این تصاویر از روش‌های فشرده‌سازی مثل JPEG استفاده می‌گردد [۷]. به تصویری که با استفاده از نزدیک‌ترین تصاویر P و یا I قبلی تخمین زده می‌شوند، تصویر P گفته می‌شود. در این نوع کدینگ از جبران‌ساز حرکت نیز استفاده می‌گردد. به تصویری که با استفاده از تصاویر P و یا I قبلی و بعدی که به عنوان مرجع هستند، تخمین زده می‌شود، تصویر B گفته می‌شود. در این نوع کدینگ نیز از جبران‌ساز حرکت استفاده می‌گردد. در این مقاله فقط از گروه تصاویر دو نوع قاب تصویر I و تصویرP استفاده شده است. علت انتخاب این تصاویر

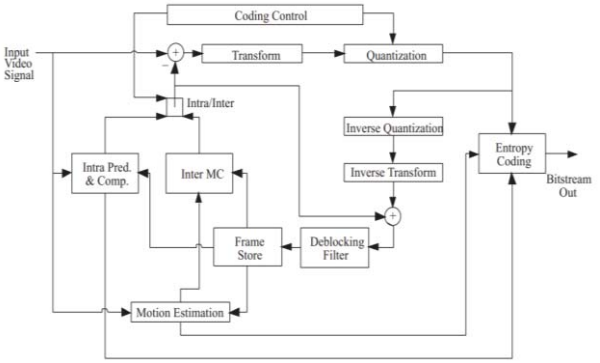
[1] Fading
[2] Joint Source and Channel Coding (JSCC)
[3] Peak Signal to Noise Ratio
[4] Bit Error Rate
[5] Group of Pictures (GOP)

آن است که در تصویر B به دلیل اینکه تخمین با استفاده از فریم‌های قبل و بعد صورت می‌پذیرد، سامانه باید تا زمانی که قاب بعد وارد و ذخیره شود، منتظر بماند تا بتواند با استفاده از آن تخمین را انجام دهد که منجر به تأخیر در سامانه می‌گردد. از این رو از تصویر B در این نوع کدینگ استفاده نشده است.

۱-۲. مروری بر روش کدگذار H.264

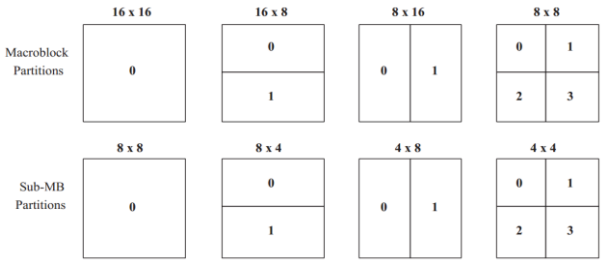
استاندارد H.264 از الگوریتم‌های پیشرفته‌ای برای تخمین حرکت، تخمین بین فریمی[1]، پیش‌بینی مکانی درون فریمی و تبدیلات استفاده می‌کند. این الگوریتم، بر خلاف الگوریتم MPEG-4 که دارای بلوک‌های تخمین حرکت ثابت با اندازه ۸×۸ است، می‌تواند محدوده متغیری بین ۱۶×۱۶ تا ۴×۴ را پشتیبانی کند. این قسمت را می‌توان به عنوان وجه تمایز و برتری الگوریتم H.264 نسبت به دیگر روش‌های موجود از قبیل MPEG-1,2,4 و ... دانست.

ساختار کدگذار H.264 در شکل (۲) نشان داده شده است. در این روش، تمام نمونه‌های روشنایی[2] و مشخصه‌های رنگی[3] از یک بلوک بزرگ در دو حوزه مکانی و زمانی تخمین زده می‌شوند و در مرحله بعد، اطلاعات باقیمانده پس از تخمین کد می‌شوند. استاندارد H.264 در انتخاب اندازه و شکل بلوک جبران‌ساز حرکت انعطاف بیشتری نسبت به دیگر روش‌های فشرده‌سازی ویدئو دارد. در این استاندارد اندازه کوچک‌ترین بلوک جبران‌سازی که برای تخمین مقادیر روشنایی به‌کار گرفته می‌شود، ۴×۴ است. در شکل (۳)، نحوه تخمین جبران‌ساز حرکت برای بلوک‌های بزرگ مختلف نشان داده شده است. در بالای شکل (۳) نحوه قسمت‌بندی بلوک‌های بزرگ نشان داده شده است و در قسمت پایین شکل (۳)، هر کدام از نواحی ۸×۸ به ۴ قسمت تقسیم شده است. بنابراین، برای نواحی که آنتروپی آن‌ها کم باشد بلوک‌های استفاده شده بزرگ می‌باشند ولی برای نواحی با آنتروپی بالا، بلوک‌های تخمین حرکت را به بلوک‌های کوچک‌تری تقسیم می‌کند. این کار باعث می‌شود خطای تخمین نسبت به روش‌های دیگر از جمله MPEG-2 کمتر شود [۸].



شکل ۲. ساختار پایه کدگذار H.264 با استفاده از [۹]

[1] Inter Prediction
[2] Luminance
[3] Chrominance



شکل ۳. قسمت‌بندی بلوک بزرگ و قسمتی از بلوک بزرگ و نحوه اسکن هر کدام از بلوک‌ها

در ادامه به توضیح برخی از عملیات استفاده شده در استاندارد H.264، نشان داده شده در شکل ۲.) پرداخته می‌شود.

پیش‌بینی کننده درون فریمی[4]: هر بلوک بزرگ بسته به نوع تخمین‌گر استفاده شده در آن می‌تواند به چندین صورت فشرده شود. یکی از روش‌های تخمین استفاده شده در استاندارد H.264 تخمین درون فریمی است. در این روش هر قاب به بلوک‌های کوچک‌تری تقسیم می‌شود و هر بلوک با استفاده از همسایگی اطراف آن تخمین زده شده و مقدار باقیمانده محاسبه می‌شود. در استاندارد H.264 از ۹ روش برای پیش‌بینی بین فریمی استفاده می‌شود (شکل (۴)). همان‌طور که در شکل (۴) نشان داده شده است، می‌توان این تخمین‌ها را به ۴ دسته کلی تخمین عمودی، افقی، قطری و DC تقسیم کرد. بنابراین در ادامه به صورت مختصر این ۴ دسته را توضیح داده می‌شود.

- تخمین گر عمودی[5]: در این تخمین‌گر ۴ مقدار A, B, C و D به صورت عمودی در ستون‌های ۴ تایی کپی شده و بلوک ساخته شده به عنوان تخمینی از بلوک اصلی با استفاده از ۴ همسایه بالا است.

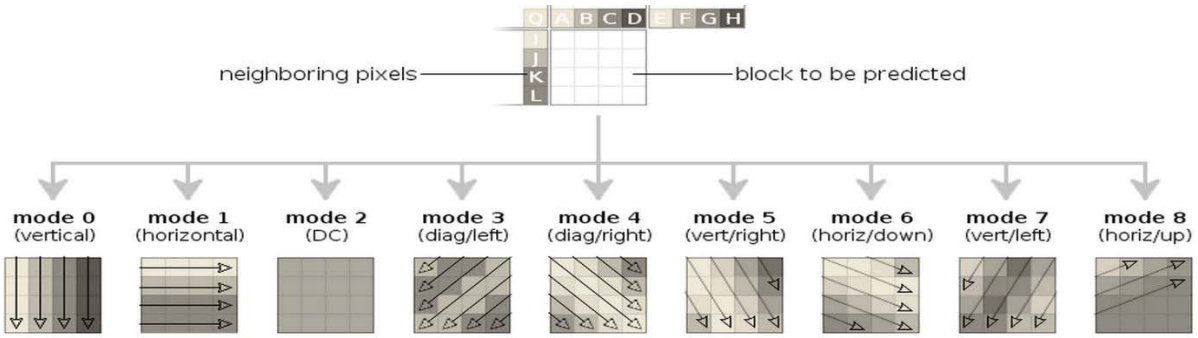
- تخمین‌گر افقی[6]: این روش شبیه به تخمین‌گر عمودی می‌باشد. با این تفاوت که به جای مقادیر همسایه بالایی بلوک مورد نظر، ۴ همسایه سمت چپ که برابرند با I, J, K و L به صورت افقی در ردیف‌های ۴ تایی کپی شده و بلوک تخمین را می‌سازند.

- تخمین‌گر قطری[7]: در این روش با استفاده میانگین قطری بین نمونه‌های مجاور بلوک مورد نظر بلوک تخمین‌گر ساخته می‌شود. به عنوان مثال برای تخمین‌گر قطری چپ میانگین A و I، B و J، C و K، D و L حساب شده و به صورت قطری چیده می‌شوند. با این کار بلوک تخمین‌گر ساخته می‌شود.

- تخمین‌گر DC: در این روش میانگین ۸ نمونه همسایگی بلوک مورد نظر حساب شده و مقدار میانگین به تمام درایه‌های بلوک تخمین‌گر نسبت داده می‌شود.

در ادامه به بررسی پیش‌بینی کننده بین‌فریمی پرداخته می‌شود.

[4] Intra-Frame Prediction
[5] Vertical Prediction
[6] Horizontal Prediction
[7] Diagonal Prediction



شکل ۴. روش‌های تخمین درون فریمی

پیش‌بینی کننده بین فریمی[1]: در این روش یک قاب از دو و یا چند قاب قبل و یا بعد از خود تخمین زده می‌شود. در ساده‌تری حالت تنها از یک قاب قبل از خود تخمین زده می‌شود. تفاوت این روش تخمین با روش‌های دیگر در آن است که برای حداقل کردن واریانس اطلاعات باقیمانده از تخمین بر پایه حرکت استفاده می‌شود. همچنین در استاندارد H.264 به منظور افزایش بهره‌وری و کاهش واریانس داده باقیمانده از بلوک‌هایی با طول متغیر استفاده می‌شود. این بلوک‌ها دارای اندازه‌های ۱۶×۱۶، ۸×۱۶، ۱۶×۸ و ۸×۸ می‌باشند. همچنین بلوک ۸×۸ نیز می‌تواند به ۳ زیر قسمت ۴×۸، ۸×۴ و ۴×۴ تقسیم شود. همان‌طور که در شکل (۳) مشاهده می‌شود، نحوه اسکن زیر بلوک‌ها نیز با مقادیر ۱ تا ۴ نشان داده شده است. در ادامه به بررسی چگونگی تخمین حرکت پرداخته می‌شود.

- تخمین‌گر حرکت: بلوک تخمین حرکت بر اساس معیار محاسبه کمترین میانگین مربعات خطای تطبیق بین دو فریم متوالی کار می‌کند که نحوه کار آن بر اساس پیاده‌سازی معادلات (۱ و ۲) است.

$$MSE\left(d_x,d_y\right)= \frac{1}{M_1N_1}\sum_{(m,n)\in W}\left(\begin{matrix} b[m,n,k]- \\ b[m-d_y,n-d_x,k-1]\end{matrix}\right)^2 \quad (۱)$$

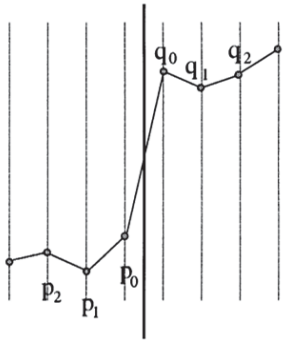
$$\begin{pmatrix} d_x \\ d_y \end{pmatrix}=\arg\min_{(d_x,d_y)} MSE\left(d_x,d_y\right) \quad (۲)$$

در رابطه (۱) آرگمان سوم متغیر b شامل k و $k-1$ پیکسل‌های دو قاب پشت سر هم را فراخوانی می‌کند و دو متغیر m و n مکان یک پیکسل را در فضای دو بعدی هر قاب نشان می‌دهد. متغیر W نشان دهنده پنجره‌ای است که باید در آن بازه جستجو صورت گیرد و بردار تخمین حرکت با جستجو در این بازه به‌دست می‌آید. مقدار W در این مقاله برابر با ۱۵ در نظر گرفته شده است. این بدان معناست که پنجره‌ای با اندازه ۱۶×۱۶ برای جستجو انتخاب می‌شود و طول پنجره تخمین، بسته به آنتروپی و اطلاعات ناحیه مورد نظر یکی از ۸ حالت نشان داده شده در شکل (۳) انتخاب می‌گردد. به عبارت دیگر، در بازه پنجره جستجو دو قاب متوالی با همدیگر مقایسه شده و کمترین مقدار به عنوان یکی از اعضا ماتریس تخمین حرکت ذخیره می‌شود. در مرحله بعد پس از تخمین کامل قاب مورد نظر و ساخته شدن ماتریس تخمین حرکت، مقدار تخمین زده شده از قاب قبلی کم می‌شود. در انتها مقدار به‌دست آمده به بلوک DCT اعمال می‌گردد. پس از عبور از بلوک DCT مقدار به‌دست آمده با ۶۴ سطح کوانتیزه می‌شود و در انتها بلوک‌هایDCT را جهت ارسال به کانال با استفاده از روش کدگذار با طول متغیر[2] کد می‌نماید [۵]. در این مقاله با استفاده از روشی نوین برای تخمین و ارسال تابع چگالی احتمال بر مبنای کد هافمن و استفاده از شبکه عصبی به منظور حافظه‌دار کردن رشته اطلاعات ارسالی برای افزایش میزان فشردگی بیت‌های اطلاعات پس از گذشتن از بلوک کدگذار با طول متغیر ارائه شده است و در مرحله بعد کدگذار کانال ثانویه‌ای با نرخی وابسته به تعداد بیت‌هایی که از تفاضل بین بیت‌های ساخته توسط بلوک کدگذار با طول متغیر و بیت‌های ساخته شده با استفاده از روش فشرده‌سازی معرفی شده در این مقاله به‌دست می‌آیند، دوباره رشته بیت‌های ارسالی را کد می‌کند. در این مقاله سعی شده است رشته بیت‌ها را حافظه‌دار نمود تا بتوان چندین مرتبه الگوریتم هافمن را با تابع چگالی‌های مختلف پیاده‌سازی کرد. در روش پیشنهادی به دلیل حافظه‌دار کردن این نوع کدگذار به راحتی می‌توان رشته بیت‌ها را در گیرنده بازیابی کرد.

محو کردن لبه‌های بلوک‌ها در فرآیند کدگذاری قاب‌ها:
مشکلی که برای رمزگذارهای MPEG-2,4 وجود دارد، به وجود آمدن خطا در لبه بلوک‌های بزرگی می‌باشد که با بقیه بلوک‌های بزرگ هماهنگ نیستند. این خطا نه تنها چشم بیننده را اذیت می‌کند، بلکه در فرآیند تخمین حرکت نیز ممانعت ایجاد می‌شود. استاندارد H.264 با اعمال فیلتر تجزیه بلوک‌ها در طی فرآیند کدگذاری، راه حلی برای حذف اثرات بصری لبه بلوک‌ها و پیامدهای به وجود آمده در فرآیند تخمین حرکت ارائه کرده است. این روش با عنوان In-Loop Deblocking شناخته می‌شود. با استفاده از مثال ساده‌ای که در شکل (۵) نشان داده شده است، می‌توان نحوه عملکرد این فیلتر را بیان کرد. در حالت کلی اساس این روش با تغییر در کوانتیزاسیون استفاده شده در مرز بین دو بلوک می‌باشد.

[1] Intra-Frame Prediction

[2] Variable Length Coding (VLC)



شکل ۵. قاعده کلی فیلتر In Loop Deblocking

به عنوان مثال فرض کنید حد آستانه استفاده شده در این مسئله با QP نشان داده شود که این مقدار آستانه با استفاده از مقدار ارزش هر پیکسل و درایه‌ی مربوط به آن در ماتریس چندی کننده است. در صورت برآورده شدن سه شرط زیر مقادیر موجود با ضرایب تعیین شده کوانتیزه می‌شوند:

$$|p_0 - q_0| < \alpha(QP) \quad (۳)$$

$$|p_1 - q_0| < \beta(QP) \quad (۴)$$

$$|q_1 - q_0| < \beta(QP) \quad (۵)$$

همچنین مقدار $\beta(QP)$ به طور قابل ملاحظه‌ای از $\alpha(QP)$ کوچک‌تر می‌باشد که می‌توان آن را به صورت ذیل نمایش داد.

$$|p_2 - p_0| < \beta(QP) \quad or \quad |q_2 q_0| < \beta(QP) \quad (۶)$$

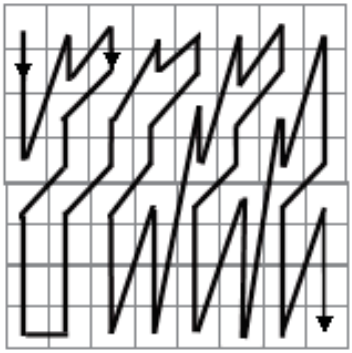
در صورتی که چندی کننده مورد نظر در لبه‌ها میزان تغییرات را بزرگ نماید، به آن ضریبی داده می‌شود تا این تغییرات به حداقل مقدار خود برسد [۱۰]. در ادامه به بررسی چندی کردن پرداخته می‌شود.

تبدیل و چندی کردن: در این مرحله، از تبدیل DCT پس از تخمین بلوک‌ها استفاده می‌شود. این تبدیل به صورتی است که ضرایب آن دارای اهمیت متفاوتی با یکدیگر می‌باشند. در مرحله بعد پس از پیاده‌سازی تبدیل روی داده باقیمانده، ماتریس ساخته شده کوانتیزه می‌شود. عملیات چندی کردن یکی دیگر از روش‌های متنوعی است که در استاندارد H.264 به‌کار می‌رود. چندی کننده استفاده شده در این استاندارد به صورت ماتریسی می‌باشد و برای هر درایه بلوک باقیمانده عددی مجزا در نظر گرفته شده است. این ماتریس در شکل (۶) نشان داده شده است. همان‌طوری که مشاهده می‌شود، با توجه به اینکه ضرایب ماتریس به‌دست آمده از تبدیل DCT دارای ارزش متفاوت می‌باشند، بسته به ارزش هر ضریب، مقدار درایه مربوط به چندی کردن آن متفاوت می‌باشد. این تفاوت در شکل (۶) نشان داده شده است [۱۱].

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

شکل ۶. ضرایب کوانتیزاسیون ماتریس DCT

مرتب‌سازی مجدد[1]: بعد از چندی‌کردن ضرایب DCT، با توجه به زیاد بودن مقادیرصفر، ضرایب به صورت زیگزاگ کنار همدیگر قرار داده می‌شوند. با این کار مقادیر مربوط به فرکانس ها گروه بندی می‌شوند [۱۲]. نحوه چیدمان زیگزاگ در شکل (۷) نشان داده شده است.



شکل ۷. نحوه انتخاب ضرایب DCT برای کنار هم چیدن در روش زیگزاگ

قسمت‌بندی کردن داده‌ها[2]: در این مدل داده‌های بسته‌بندی شده به دو ناحیه تقسیم می‌شوند. ایده اصلی این مدل، جداسازی دادها با ارزش بیشتر (ضرایب DC ماتریس DCT، اطلاعات روش کدگذاری و بردارهای حرکت) از داده‌ها با ارزش پایین‌تر (ضرایب AC ماتریس DCT و خطاهای باقیمانده) می‌باشد. برای ارسال تصویر I، اولین ناحیه، شامل اطلاعات روش کدگذار منبع و همچنین ضرایب DC است. اما ناحیه دوم که داده‌های با ارزش پایین‌تر را داراست، شامل ضرایب AC می‌باشد. برای ارسال تصویر P اولین ناحیه شامل اطلاعات روش کدگذار منبع و همچنین بردارهای حرکت است، در حالی که ناحیه دوم شامل اطلاعات DCT (بافت، ضرایب DC و AC) می‌باشد. برای هم‌زمان‌سازی در قسمت‌بندی کننده داده‌ها نیز از نشانه‌گذاری استفاده می‌شود که این نشانه‌گذاری یکتا است.

کدگذار با طول متغیر[3]: این قسمت با عنوان کدگذار آنتروپی[4] در شکل (۲) نشان داده شده است. این نوع کدگذار بر پایه کاهش اطلاعات ارسالی طراحی شده است. این کدگذار به دو صورت این عمل را انجام می‌دهد.

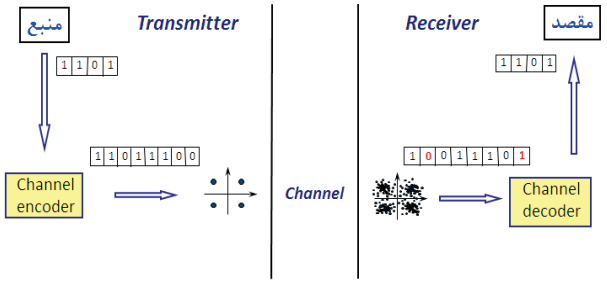
[1] Reorder
[2] Data Partitioning
[3] Variable Length Coding
[4] Entropy Encoder

- یکی از روش‌هایی که برای کاهش اطلاعات ارسالی در این نوع کدگذار استفاده می‌شود با توجه به زیاد بودن مقادیر یک در ماتریس DCT پس از چندی کردن است. دلیل این امر مقادیر زیاد ضرایب چندی کردن برای مقادیر کم‌ارزش تبدیل DCT می‌باشد. بنابراین تعداد بیت کمی به آن‌ها اختصاص می‌دهد. برای مثال تنها ۳ بیت برای ارسال عدد یک اختصاص می‌دهد که عبارآزمایش از $11s$ که این کد با استفاده از جدول استاندارد تعیین شده برای MPEG به‌دست آورده شده است [۵]. در مثال قبل مقدار s با توجه به علامت درایه مورد نظر تعیین می‌شود، بدین صورت که اگر مثبت باشد $s=0$ و اگر منفی باشد $s=1$ در نظر گفته می‌شود. به همین ترتیب تعداد بیت ارسالی برای اعداد بزرگ‌تر از ۱ با توجه به اینکه نسبت به عدد ۱ کمتر ارسال می‌شوند، بیشتر است. برای مثال برای ارسال عدد ۷، ۱۱ بیت استفاده می‌شود.

- یکی دیگر از روش‌هایی که برای کاهش اطلاعات ارسالی در این نوع کدگذار استفاده می‌شود با در نظر گرفتن این واقعیت است که تعداد عناصر صفر نیز در ماتریس DCT پس از چندی کردن زیاد است. این نوع کدگذار پس از چیده شده درایه‌های ماتریس DCT به صورت زیگزاگ جستجو می‌کند که آیا درایه بعدی صفر است یا خیر. در صورت صفر بودن به عدد بعد مراجعه می‌کند و در هر مرحله تعداد صفرها را می‌شمارد و این کار تا زمانی ادامه پیدا می‌کند که به درایه غیر صفر برسد و یا اینکه تمام درایه‌های ماتریس DCT تمام شوند. پس از مشخص کردن تعداد صفرها به ازای تعداد تکرار صفر پس از درایه غیر صفر یک کد از جدول استاندارد تعیین می‌گردد. برای مثال برای ارسال عدد یک که پس از آن ۵ صفر وجود دارد به جای اینکه برای هر درایه ۸ بیت اختصاص داده شود که منجر به ارسال ۴۸ بیت می‌شود، تنها ۷ بیت که عبارتست از $000111s$ ارسال می‌گردد که متغیر s در همین بخش توضیح داده شده است. این نوع کدگذار بر اساس دو پارامتر در جدول استاندارد به صورت میزان/ ادامه[1] نشان داده می‌شود. مقدار میزان برابر است با درایه غیر صفر ماتریس DCT و ادامه برابر است با تعداد صفر موجود پس از عدد انتخاب شده قبل از اینکه به عدد غیر صفر دیگری برسد، این مقدار می‌تواند صفر باشد. به این معنی که عدد بعدی نیز مخالف صفر می‌باشد [۱۳].

۳. کدگذاری کانال

همان‌طور که می‌دانید، اگر کانالی خالی از هر گونه خطا باشد و نیز گیرنده‌ای با توان تشخیص کاملاً درست بیت‌های دریافتی در دسترس باشد، آنچه ارسال می‌شود توسط گیرنده بازشناسی خواهد شد. ولی وجود عوارض فیزیکی متعدد و متنوع در کانال‌های انتشار نظیر نویز، فیدینگ، تداخل، دریافت چند مسیره و ... منجر شده است که روش‌هایی برای محافظت اطلاعات در مقابل این اعوجاج‌ها پیشنهاد شود. به عنوان یک اصل، برای تمامی روش‌های کدگذاری با افزودن تعدادی بیت، با افزایش افزونگی[2] نرخ بیت خطا کاهش می‌یابد [۱۴ و ۱۵]. این عملیات را می‌توان در شکل (۸) مشاهده نمود.



شکل ۸. مثالی از چگونگی ایجاد تداخل و حذف آن توسط کدگذار کانال.

به طور کلی کد کردن کانال را می‌توان در دو دسته طبقه‌بندی کرد. دسته اول کدگذار با نگاه به عقب (ARQ[3]) و دسته دوم کدگذار مبتنی بر تصحیح مستقیم (FEC[4]) است. در این مقاله از کدگذار ریدسالامون[5] که از نوع EFC می‌باشد، برای کد کردن کانال استفاده شده است که این کدکلاسی از کدهای BCH دوره‌ای [۱۶ و ۱۷]، خطی و غیر باینری که توسط میدان $GF(q)$ ساخته می‌شوند، می‌باشد [۱۸]. دلیل استفاده از این نوع کدگذار غیر باینری بودن آن است. بدین صورت که اطلاعات هر بلوک را می‌توان بدون اینکه بلوک مورد نظر را به باینری تبدیل کرده و مستقیماً کد نمایید و بدین صورت سرعت کدگذاری قاب‌های ویدئویی بالاتر می‌رود. در این روش با فرض اینکه پیغام کد شده $c(X)$ ارسال شده و پیام نویزی شده $r(X)$ در گیرنده دریافت شود، داریم:

$$r(X)=r_0+r_1X+\cdots+r_{n-1}X^{n-1} \qquad (۷)$$

همچنین چند جمله‌ای خطا $e(X)$ و رابطه آن با پیام ارسالی و دریافتی در رابطه (۸) آورده شده است.

$$e(X)=e_0+e_{1(X)}+\cdots+e_{n-1}X^{n-1}$$
$$r(X)=c(X)+e(X) \qquad (۸)$$

بنابراین با توجه به رابطه (۸) می‌توان نشان داد که $r(\alpha^i)=c(\alpha^i)+e(\alpha^i)=e(\alpha^i)$.

در این نوع کدگذاری برای پیدا کردن خطا علاوه بر محل خطا باید مقدار خطا نیز محاسبه شود. با توجه به مقدارمتغیر t و همچنین رابطه $\beta_i=\alpha^{ji}$ می‌توان با استفاده از رابطه (۵) محل و مقدار خطا را به‌دست آورد. این نوع کدگذاری قادر است هر الگوی خطا با طول t یا کمتر را تصحیح نماید.

$$S_1=r(\alpha)=e(\alpha)=e_{j1}\beta_1+e_{j2}\beta_2+\ldots+e_{jt}\beta_t$$
$$S_1=r(\alpha^2)=e(\alpha^2)=e_{j1}\beta_1^2+e_{j2}\beta_2^2+\ldots+e_{jt}\beta_t^2 \qquad (۹)$$
$$\vdots$$
$$S_{2t}=r(\alpha^{2t})=e(\alpha^{2t})=e_{j1}\beta_1^{2t}+e_{j2}\beta_2^{2t}+\ldots+e_{jt}\beta_t^{2t}$$

[1] Run/Level

[2] Redundancy
[3] Automatic Repeat Request
[4] Forward Error Correction
[5] Reed-Solomon

در ساده‌ترین حالت، در این نوع کدگذاری با فرض $t=1$ مقدار و محل خطا با استفاده از رابطه (۱۰) محاسبه می‌شود.

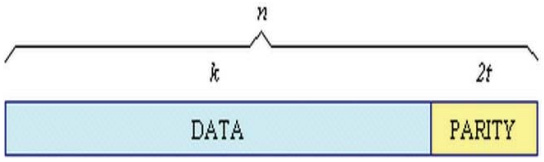
$$\alpha^{j1}=\frac{S_2}{S_1}\Rightarrow Error\ Place \qquad e_{j1}=\frac{S_1^2}{S_2}\Rightarrow ErrorValue \tag{۱۰}$$

رابطه نرخ کد کننده کانال در رابطه (۱۱) نشان داده شده است و واحد آن بیت بر ثانیه می‌باشد.

$$R_c=\frac{k}{n} \tag{۱۱}$$

در رابطه (۱۱)، k تعداد بیت‌های مربوط به اطلاعات قاب می‌باشد که قرار است کد شوند. n طول رشته ساخته شده برای ارسال است که شامل پیام اصلی و اطلاعات اضافه شده توسط رمزگذار کانال می‌باشد.

رمزگذار با مؤلفه‌های فوق قادر است به میزان $\frac{n-k}{2}$ خطا را تشخیص و تصحیح کند، بنابراین هر چه میزان $n-k$ (تعداد بیت‌های توازن) بیشتر باشد تعداد بیشتری از بیت‌های پیام را می‌توان تصحیح کرد. در این مقاله از $\mathrm{RS}(255,223)$ به عنوان کدگذار کانال استفاده ده است. نمونه‌ای از کلمه‌کد ساخته شده در شکل (۹) نشان داده شده است. این کدگذار روی میدان $GF(2^8)$ می‌باشد به طوری که $k=223$، $s=8$، $n=2^s-1=2^8-1=255$ که هر سمبل تنها از ۸ بیت برای کد کردن اطلاعات استفاده می‌کند. همچنین باید به این نکته اشاره کرد که $2t=32\Rightarrow t=16$ است. به این معنا که این نوع کدگذار قادر به تصحیح ۱۶ بایت (۱۶ سمبل) از ۲۵۵ سمبل می‌باشد.



شکل ۹. نمونه‌ای از کلمه کد ریدسالامون[1]

۴. نحوه به‌دست آوردن تابع چگالی احتمال برای یک رشته بیت با طول دلخواه

برای به‌دست آوردن تابع چگالی احتمال نیاز به تعداد زیادی بیت است. بدین منظور در ابتدا طول مورد نظری از بیت‌ها را از رشته اطلاعات جدا می‌کنید یا اینکه منتظر می‌مانید تا بافر به اندازه طول مورد نظر پر شود. پس از آن هر ۸ بیت از رشته مورد نظر را داخل یک بافر دیگر می‌ریزید. با این کار بافر به جای اعداد ۱ و ۰ اعدادی بین ۰ تا ۲۵۵ را در مبنای ده شامل می‌شود. در مرحله بعد تعداد تکرار این ۲۵۶ عدد در رشته مورد نظر محاسبه می‌گردد. بنابراین می‌توان هیستوگرام را بر حسب تعداد تکرار این ۲۵۶ عدد ساخته شده در رشته مورد نظر به‌دست آورد.

پس از این مرحله باید تابع احتمال رخداد نمونه‌ها را که از هیستوگرام (شکل (۱۰)) ساخته شده، به‌دست آورد. برای به‌دست آوردن تابع احتمال رخداد نمونه‌ها از روی هیستوگرام باید شروط (۱۲، ۱۳ و ۱۴) ارضا شوند.

$$f_X(X)\geq 0 \tag{۱۲}$$

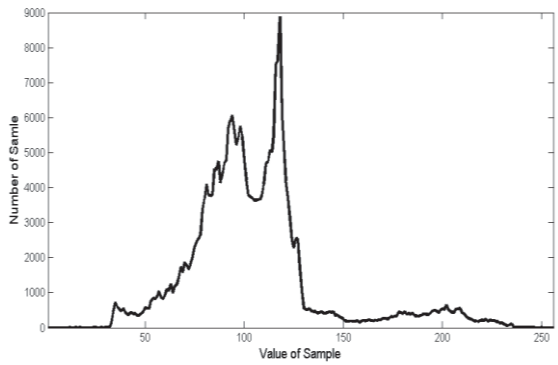
$$f_X(X)\leq 1 \tag{۱۳}$$

$$\sum_{-\infty}^{+\infty} f_X(X)=1 \tag{۱۴}$$

در شروط (۱۲، ۱۳ و ۱۴)، $f_x(X)$ مقدار تابع هیستوگرام ساخته شده است که نشان دهنده تعداد تکرار اعداد بین ۰ تا ۲۵۵ می‌باشد. با بررسی هیستوگرام می‌توان نشان داد که شرط (۸) ارضا می‌شود، زیرا تعداد تکرار یک عدد همواره عدد مثبتی خواهد بود و در صورت نبودن یک عدد مقدار آن در نمودار هیستوگرام (رابطه (۱۱)) برابر با صفر خواهد شد. اما تابع هیستوگرام به تنهایی شروط ۲ و ۳ را ارضا نمی‌کند. برای ارضا کردن شروط ۲ و ۳ باید تعداد تکرار برای هر نقطه را بر مجموع کل تکرارهای موجود تقسیم کرد (رابطه (۱۱)). در این صورت شروط ۲ و ۳ نیز ارضا می‌شوند.

$$f_X(X_i)=\frac{f_X(X_i)}{\sum_{k=0}^{255} f_X(X_k)} \tag{۱۵}$$

تا این مرحله می‌توان تابع احتمال رخداد نمونه‌ها در یک رشته را به‌دست آورد. اما با توجه به اینکه تابع احتمال رخداد نمونه‌ها برای هر رشته شکلی متفاوت از رشته دیگر دارد، برای کد کردن این رشته‌ها با الگوریتم هافمن باید تابع احتمال رخداد نیز همراه با رشته اطلاعات مربوط به آن تابع احتمال رخداد ارسال شود. یکی از روش‌هایی که می‌توان برای ارسال تابع احتمال رخداد استفاده کرد، این است که بر روی آن نموداری منطبق کرده و اطلاعات آن نمودار را به همراه رشته اطلاعات ارسال نمود. اکنون این سؤال پیش می‌آید که با توجه به تغییرات سریع و غیر خطی تابع احتمال رخداد نمونه‌ها چه نموداری می‌توان بر روی آن منطبق کرد به وجهی که کمترین خطا را با تابع چگالی احتمال مورد نظر داشته باشد. برای این کار می‌توان از شبکه عصبی مصنوعی کمک گرفت که در بخش ۵ به صورت خلاصه به توضیح آن پرداخته می‌شود.



شکل ۱۰. هیستوگرام تعداد تکرار نمونه ها در یک رشته اطلاعات.

[1] Reed-Solomon Code Word

۵. شبکه عصبی مصنوعی

شبکه عصبی مصنوعی از یک سری لایه‌ها و گره‌ها تشکیل شده است. هر شبکه الزاماً دارای لایه ورودی و خروجی می‌باشد [۱۹ و ۲۰]. در هر شبکه می‌توان به دلخواه تعدادی لایه فرعی اضافه کرد. هر لایه شامل تعداد دلخواهی گره است که تحت عنوان نرون‌ها شناخته می‌شوند. ارتباط بین هر لایه توسط لینک‌هایی که به نرون‌ها متصل می‌شوند، برقرار می‌گردد. به این لینک‌ها در اصطلاح وزن‌های شبکه گفته می‌شود. برای مطالعه بیشتر می‌توان به مراجع [۲۰ و ۲۱] مراجعه کرد.

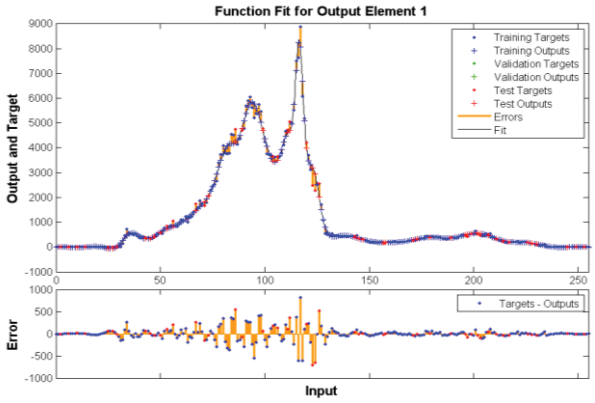
۵-۱. نحوه استفاده از شبکه عصبی برای توصیف تابع چگالی احتمال

هدف از این قسمت، منطبق نمودن تابعی بر منحنی احتمال رخداد نمونه‌ها (شکل (۱۰)) با استفاده از شبکه عصبی می‌باشد [۲۱]. برای این کار ابتدا یک شبکه عصبی با یک لایه پنهان که آموزش ندیده است، ایجاد می‌شود. سپس تمام وزن‌ها و بایاس‌های شبکه به عنوان متغیرهای بهینه‌سازی معرفی می‌شوند. ورودی شبکه عصبی مقادیر محور افقی تابع چگالی احتمال (اعداد صحیح بین ۰ تا ۲۵۵) و خروجی شبکه عصبی مقادیر محور عمودی تابع احتمال رخداد نمونه‌ها معرفی می‌شوند. تابع هدف در شبکه عصبی به‌کار گرفته شده، بر اساس میانگین مربعات خطا بین احتمال رخداد نمونه‌های تابع چگالی احتمال و مقادیر حساب شده با توجه به وزن‌ها و بایاس‌های شبکه عصبی معرفی می‌شوند. بنابراین در هر مرحله از بهینه‌سازی، مقادیر وزن‌ها و بایاس‌های شبکه به گونه‌ای به‌روز می‌شوند که خطای شبکه کاهش یابد. در این مقاله برای منطبق کردن تابعی بر نمودار چگالی احتمال از شبکه عصبی پرسپترون با یک لایه پنهان که دارای ۶۰ نرون می‌باشد، استفاده شده است. تابع انتقال نرون‌های لایه پنهان tansig است و تابع انتقال لایه خروجی purelin می‌باشد. همچنین لایه ورودی و خروجی تک نرونی هستند و تمامی نرون‌های هر سه لایه دارای بایاس می‌باشند. 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 [۲۲ و ۲۳]، پس از امتحان بر روی چندین تابع چگالی احتمال انتخاب شده‌اند. در این روش قصد ما در آموزش دادن شبکه عصبی بر خلاف آموزش‌های معمول می‌باشد. به این دلیل که در آموزش‌های معمول درصدی از اطلاعات به عنوان داده‌های آزمون کنار گذاشته می‌شوند و در انتهای هر بار آموزش و به‌روز شدن وزن‌ها و بایاس‌ها، داده‌های آزمون به عنوان ورودی به شبکه عصبی وارد می‌شوند و خروجی‌های به‌دست آمده با مقادیر صحیح مقایسه می‌شوند. اگر خطا از حد مشخص شده کمتر باشد، وزن‌ها و بایاس‌های به‌روز شده به عنوان وزن‌ها و بایاس‌های شبکه ثبت می‌گردند. ولی اگر خطا بیشتر از حد مشخص شده باشد، آموزش دوباره تکرار می‌شود. این کار از آموزش بیش از حد جلوگیری می‌کند.

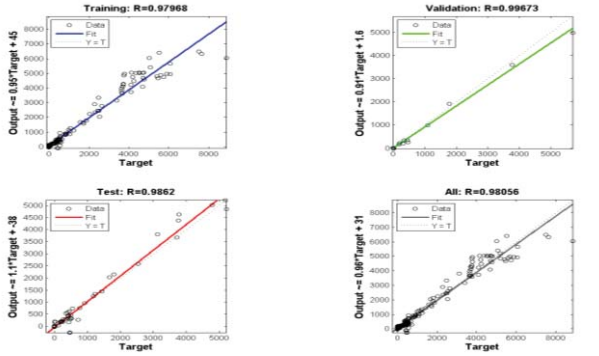
اما در این مقاله بر خلاف روش‌های معمول سعی بر این است که شبکه کاملاً بر مقادیر احتمال رخداد نمونه‌ها منطبق گردد و آموزش بیش از حد رخ دهد. برای اینکه بتوان به این هدف رسید تمامی مقادیر احتمال رخداد نمونه‌ها به عنوان نقاط آموزشی به شبکه عصبی وارد می‌شوند و در نتیجه پس از چندین بار به‌روز شدن مقادیر وزن‌ها و بایاس‌ها، مقادیر تابع به‌دست آمده با استفاده از شبکه عصبی بسیار به منحنی چگالی احتمال نزدیک می‌شود.

۵-۲. پیاده‌سازی شبکه عصبی برای تابع چگالی احتمال

به دلیل متغیر با زمان بودن منحنی‌های چگالی احتمال و همچنین به دلیل افت و خیزهای بالا در این منحنی‌ها نمی‌توان توابعی را با استفاده از روش‌های کلاسیک برای منطبق کردن بر روی این منحنی‌ها به‌دست آورد. بنابراین در این مقاله از شبکه عصبی برای منطبق ساختن تابعی بر روی منحنی چگالی احتمال استفاده شده است. اگر بتوان تابعی بر منحنی چگالی احتمال منطبق سازید، قادر به ذخیره‌سازی مقادیر تابع محاسبه شده خواهید بود که در این مقاله از این روش برای حافظه‌دار کردن توابع چگالی احتمال استفاده شده است. برای مثال اگر شبکه عصبی ساخته شده با ۶۰ نرون در لایه پنهان که دارای ۱۲۰ وزن و ۶۱ بایاس باشد[۱] را برای رشته اطلاعاتی که دارای منحنی احتمال رخداد نمونه‌ها شبیه به شکل (۱۰) است آموزش دهیم. خروجی در بهترین حالت با فرض (خطای میانگین مربعی کمتر از 10^{-10}) در شکل (۱۱) و شکل (۱۲) نشان داده شده است.



شکل ۱۱. رسم هم‌زمان مقادیر تابع هدف همراه با مقادیر خروجی به دست آمده با شبکه عصبی و نمودار خطا بین تابع هدف با خروجی شبکه عصبی



شکل ۱۲. رسم میزان پراکندگی مقادیر هدف از مقادیر خروجی برای ۴ مقدار آموزش، تصدیق، آزمایش و تمامی مقادیر

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

۶. مفهوم حافظه‌دار کردن رشته اطلاعات

با توجه به مطالب گفته شده در بخش‌های ۵-۱ و ۵-۲ می‌توان گفت که تابع چگالی احتمال را با استفاده از شبکه عصبی مصنوعی و در مواردی که بیشتر مقادیر تابع چگالی احتمال در چند نمونه محدود شده است با استفاده از این چند نمونه (در بخش ۷-۱ توضیح داده می‌شود) می‌توان توصیف کرد. بنابراین از این به بعد می‌توان تابع احتمال رخداد نمونه‌ها را با وزن‌ها و بایاس‌های به‌دست آمده از آموزش شبکه عصبی و یا تعداد محدودی عدد توصیف نمود. همچنین رفتار تابع احتمال رخداد نمونه‌ها را با استفاده از مجموعه وزن‌ها و بایاس‌های شبکه عصبی و یا تعداد محدودی عدد ذخیره‌سازی کرد. برای ذخیره هرکدام از وزن‌ها و بایاس‌ها و یا اعداد توضیح داده شده ۲ بایت بیشتر نیاز نیست. در ادامه مزیت حافظه‌دار کردن رشته اطلاعات را خواهید دید.

با استفاده از حافظه‌دار کردن رشته اطلاعات می‌توان الگوریتم هافمن را چندین بار بر روی رشته اطلاعات پیاده‌سازی کرد. بدین صورت که قبل از هر بار ارسال اطلاعات، تابع چگالی احتمال تقریب زده می‌شود و سپس الگوریتم هافمن با استفاده از مقادیر تابع احتمال رخداد نمونه‌ها محاسبه شده برای هر مرحله اجرا می‌گردد. اطلاعات مربوط به تابع احتمال رخداد نمونه‌ها به ابتدای رشته بیت فشرده شده اضافه می‌شوند. با تکرار این روش می‌توان حجم رشته اطلاعات ارسالی را بسیار کم کرد. البته در ادامه نشان داده خواهد شد که این عملیات به تعداد محدود قابل انجام می‌باشد.

۷. الگوریتم پیشنهادی

۷-۱. کدگذار الگوریتم پیشنهادی

فلوچارت کدگذار الگوریتم پیشنهادی درشکل (۱۳) نشان داده شده است. اساس این روش بدین صورت است که پس از فشرده‌سازی اولیه توسط الگوریتم فشرده‌ساز H.264 رشته بیت‌های ساخته شده مجدد با استفاده از احتمال رخداد نمونه‌های ساخته شده فشرده می‌شوند. این روش بر ۲ پایه استفاده از شبکه عصبی و بدون استفاده از آن صورت می‌گیرد. روش استفاده شده برای فشرده کردن الگوریتم هافمن است که با روش پیشنهادی ترکیب شده و منجر به میزان فشرده‌سازی بیشتری شده است. در مرحله بعد کد کننده کانال ثانویه‌ای با نرخی وابسته به تعداد بیت‌هایی که با استفاده از الگوریتم پیشنهاد شده بهینه گردیده‌اند دوباره رشته بیت‌های ارسالی را کد می‌کند که این کار باعث مقاوم‌تر شدن اطلاعات نسبت به خطا و اعوجاج کانال می‌شود. در ادامه به بررسی جزیی تمامی مراحل کد کننده پیشنهادی پرداخته می‌شود.

- مرحله ۱: رشته بیت اطلاعات را در بسته‌های ۸ بیتی جدا نموده و سپس از مبنای دودویی به مبنای دهدهی تبدیل کنید (بخش ۴).
- مرحله ۲: منحنی هیستوگرام احتمال رخداد نمونه‌ها را بر اساس تعداد تکرار اعداد به‌دست آمده از مرحله ۱ (محدوده تغییرات این اعداد بین ۰ تا ۲۵۵ است) به‌دست آورید (بخش ۴).
- مرحله ۳: به‌دست آوردن منحنی احتمال رخداد نمونه‌ها با استفاده از منحنی هیستوگرام، با ارضا کردن شروط ۸، ۹ و ۱۰ (بخش ۴).
- مرحله ۴: اگر تعداد تکرار الگوریتم با استفاده از شبکه عصبی از ۴ مرتبه بیشتر باشد، به مرحله ۱۶ پرش کن در غیر این صورت استفاده از شبکه عصبی برای منطبق کردن تابعی بر روی منحنی چگالی احتمال (بخش ۵). علت این مرحله به آن دلیل است که استفاده زیاد از شبکه عصبی منجر می‌شود به وزن‌ها و بایاس‌هایی که باید ذخیره گردد، افزایش پیدا می‌کند. بنابراین، میزان فشرده‌سازی با افزایش تعداد تکرار شبکه عصبی از جایی به بعد کاهش پیدا می‌کند. تعداد دفعات تکرار استفاده از شبکه عصبی با متغیر f در بلوک دیاگرام روش پیشنهادی نشان داده شده است.
- مرحله ۵: اجرای الگوریتم هافمن با استفاده از تابع احتمال رخداد نمونه‌های به‌دست آمده از مرحله ۴.
- مرحله ۶: تبدیل وزن‌ها و بایاس‌ها از مبنای دهدهی به مبنای دودویی و اضافه کردن آن‌ها به انتهای رشته اطلاعات فشرده شده جهت ارسال. این قسمت به منظور حفظ اطلاعات ساختاری شبکه عصبی در هر مرحله فشرده‌سازی صورت می‌پذیرد که با عنوان حافظه‌دار کردن اطلاعات در بخش ۰ توضیح داده شده است.

مرحله بعد با توجه به توضیح زیر در رابطه با ویژگی کد هافمن ادامه داده می‌شود:

با توجه به اینکه کد هافمن در جهتی رشته اطلاعات را کد می‌کند که تعداد صفرهای ساخته شده بیشینه باشد [۲۴ و ۲۵]، اگر دوباره عملیات مرحله ۱ روی رشته بیت ساخته شده از خروجی کد هافمن صورت پذیرد، همان‌طور که در شکل (۱۴) نشان داده شده است، بیشتر اعداد در محدوده‌های پایین تکرار شده‌اند و اعداد بزرگ‌تر تنها دارای تکرارهای محدودی هستند. این موضوع به صورت تجربی به‌دست آمده است، بنابراین شرطی برای برقراری این موضوع چک می‌شود و در صورت برآورده شدن به مرحله بعد می‌رود.

- مرحله ۷: رشته بیت ساخته شده در مرحله ۶ و یا ۱۴ را دوباره در بسته‌های ۸ بیتی جدا نموده و سپس از مبنای دودویی به مبنای دهدهی تبدیل کنید (بخش ۴).
- مرحله ۸: منحنی هیستوگرام را بر اساس تعداد تکرار اعداد به‌دست آمده از مرحله ۷ (رنج این اعداد بین ۰ تا ۲۵۵است) به‌دست آورید (بخش ۴).
- مرحله ۹: به‌دست آوردن منحنی احتمال رخداد نمونه‌ها با استفاده از منحنی هیستوگرام، با ارضا کردن شروط ۸، ۹ و ۱۰ (بخش ۴).
- مرحله ۱۰: در صورت برقرار بودن شرط ۱۲ به مرحله بعد رفته و در غیر این صورت به مرحله ۴ پرش کنید.

$$\sum_{k=0}^{j} f_X(X_k) > 0.9 \;\; ; \;\; j = 1, 2, 3, 4 \qquad (۱)$$

• مرحله ۱۱: مقادیر احتمال‌های 0 تا j را ذخیره‌سازی می‌شود.

• مرحله ۱۲: در این مرحله تابع چگالی احتمال جدیدی ساخته می‌شود. با توجه به این مطلب که بیشتر مقادیر تابع چگالی احتمال را که در چند نمونه اولیه قرار داشت، ذخیره شده است. بنابراین به دلیل بسیار ناچیز بودن دیگر مقادیر تابع چگالی احتمال فرض می‌شود تمام این نمونه‌ها دارای مقادیر تابع چگالی احتمال برابر هستند. بنابراین مقدار به‌دست آمده با استفاده از رابطه (۱۳) به هریک از آن‌ها نسبت داده می‌شود.

$$f_X(X_i) = \left. \frac{1 - \sum_{k=0}^{j} f_X(X_k)}{256 - j} \right|_{i=\begin{cases} j+1 \\ j+2 \\ \vdots \\ 255 \end{cases}} \qquad (۲)$$

• مرحله ۱۳: اجرای الگوریتم هافمن با استفاده از تابع چگالی احتمال ساخته شده در مرحله ۱۲.

مرحله ۱۴: مقدار j عدد اولیه تابع چگالی احتمال ذخیره شده در مرحله ۱۱ و همچنین مقدار j نیز از مبنای دهدهی به مبنای دودویی تبدیل شده و به انتهای رشته اطلاعات فشرده شده جهت ارسال اضافه می‌شوند.

• مرحله ۱۵: اگر تعداد تکرار الگوریتم ساختن تابع چگالی احتمال بدون استفاده از شبکه عصبی از ۴ مرتبه بیشتر باشد به مرحله ۱۶ پرش کن در غیر این صورت دوباره به مرحله ۷ باز گردید.

• مرحله ۱۶: مقدار f و d (به ازای هر مقدار ۳ بیت اختصاص داده می‌گردد. به دلیل اینکه بیشترین مقدار روش پیشنهادی ۵ بار تکرار است که با ۳ بیت می‌توان نمایش داد) و همچنین ترتیب اجرای آن‌ها به انتهای رشته بیت ساخته شده اضافه می‌شود (به این معنا که آیا تابع چگالی احتمال در هر مرحله با استفاده از شبکه عصبی تخمین زده شده یا خیر). به این منظور به هر روش یک کد ۲ بیتی داده می‌شود و پس از مقدار f و d نحوه ترتیب آن‌ها نیز آورده می‌شود که این کد به صورت 10 و 01 است.

• مرحله ۱۷: در این مرحله کد گذار کانال ثانویه با استفاده از نرخ کد کردن محاسبه شده در رابطه (۱۴) رشته بیت ساخته شده را کد می‌کند و نرخ کد کردن نیز به انتهای رشته اضافه می‌شود.

$$R_c^{'} = \frac{N_b^{'}}{N_b} \qquad (۳)$$

در رابطه (۱۴) منظور از $R_c^{'}$ نرخ کدینگ کانال ثانویه است و منظور از N_b تعداد بیت ساخته شده توسط بلوک VLC می‌باشد (تعداد بیتی که در نهایت ساخته می‌شود باید به همین تعداد باشد) و منظور از $N_b^{'}$ تعداد بیت ساخته شده پس از فشرده‌سازی با استفاده از کد هافمن می‌باشد.

مرحله ۱۸: رشته بیت ساخته شده ارسال می‌شود.

۷-۲. رمزگشای الگوریتم پیشنهادی

فلوچارت رمزگشای الگوریتم پیشنهادی در شکل (۱۵) نشان داده شده است. مراحل رمزگشایی اطلاعات به شرح زیر می‌باشد:

• مرحله ۱: ابتدا نرخ کدگذاری کانال ثانویه از انتهای رشته دریافتی محاسبه می‌شود و سپس کدبردار کانال با استفاده از این نرخ کد کردن، اطلاعات را رمزگشایی می‌نماید.

• مرحله ۲: مقادیر f و d از انتهای رشته بیت رمزگشایی شده به همراه ترتیب اجرای آن‌ها محاسبه می‌شوند.

مرحله ۳: در صورتی که تابع چگالی احتمال با استفاده از شبکه عصبی تخمین زده شده باشد (شرط ۱۲ صورت نپذیرفته است) به اندازه Q بیت به‌دست آمده با استفاده از رابطه (۱۵) از انتهای رشته اطلاعات جدا کنید.

$$\mathrm{Q} = (2 + 3 \times \mathrm{N}_n) \times \frac{2 \times \mathrm{byte}}{\mathrm{neuron}} \times \frac{8 \times \mathrm{bit}}{\mathrm{byte}} \qquad (۴)$$

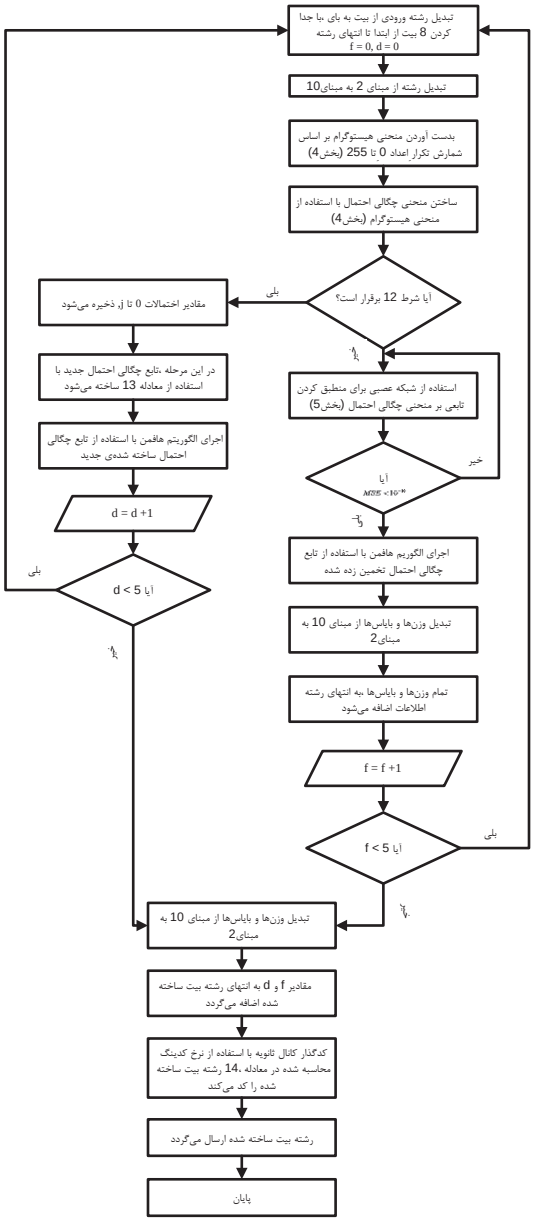
در رابطه (۱۵)، منظور از N_n تعداد نرون لایه پنهان و به دلیل اینکه لایه خروجی تک نرونی است، تعداد وزن‌های لایه پنهان برابر است با $2*\mathrm{N}_n$. همچنین هر لایه یک بایاس نیز دارد، بنابراین به تعداد N_n بایاس در لایه پنهان دارید، بنابراین برای ارسال وزن‌ها و بایاس لایه پنهان مقدار $3*\mathrm{N}_n$ عدد باید ارسال شود. برای لایه خروجی به دلیل اینکه تک نرونی است فقط ۲ عدد که یکی برای وزن خروجی و دیگری برای بایاس است ارسال می‌شود. همچنین برای ارسال هر وزن و بایاس ۲ بایت[1] در نظر گرفته شده است.

• مرحله ۴: با استفاده از وزن‌ها و بایاس‌های تولید شده از مرحله ۳ و با استفاده از شبکه عصبی تابع چگالی احتمال دوباره بازسازی می‌شود.

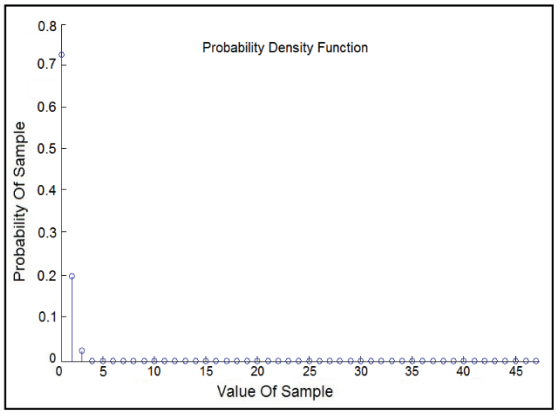
• مرحله ۵: با توجه به تابع چگالی احتمال به‌دست آمده، رشته اطلاعات را با استفاده از الگوریتم هافمن رمزگشایی می‌شود.

• مرحله ۶: دوباره بررسی می‌شود که آیا در کدگذار تابع چگالی احتمال با استفاده از شبکه عصبی تخمین زده شده یا خیر. اگر با استفاده از شبکه عصبی تخمین زده شده بود که به مرحله ۳ پرش می‌کنید. با استفاده از دو کد 01 و 10 مشخص می‌گردد که در بخش ۷-۱ مرحله ۱۶ توضیح داده شده است، (برای تمام قاب‌های ارسالی تنها برای مرتبه اول شبکه عصبی مورد استفاده قرار گرفته شده است) در غیر این صورت به مرحله بعد بروید.

[1] Byte



شکل ۱۳. فلوچارت کدگذار الگوریتم پیشنهادی



شکل ۱۴. منحنی تابع احتمال رخداد نمونه‌ها جدید به‌دست آمده از منحنی شکل (۱۰)، پس از پیاده‌سازی کد هافمن بر روی رشته اطلاعات در بازه $0 \le Value\ of\ \ sample \le 47$.

• مرحله ۷: در این مرحله به تعداد P بیت که از رابطه (۱۶) محاسبه می‌شود، از انتهای رشته بیت ساخته شده از مرحله قبل جدا کنید. در رابطه (۱۶) منظور از j تعداد اعدادی که در رابطه (۱۲) صدق می‌کند را نشان می‌دهد.

$$P = j \times \frac{2 \times byte}{neuron} \times \frac{8 \times bit}{byte} \qquad (۵)$$

در رابطه (۱۶) منظور از j ، تعداد مقادیر احتمالی که به انتهای رشته اطلاعات اضافه شده است در مرحله ۱۰ کدگذار روش پیشنهادی محاسبه شده بودند و در مرحله ۱۴ کدگذار روش پیشنهادی به انتهای رشته بیت اطلاعات اضافه گردیده‌اند.

• مرحله ۸: در این مرحله مقادیر دیگر تابع چگالی احتمال با استفاده از رابطه (۱۳) محاسبه می‌شوند و بدین ترتیب تابع چگالی احتمال ساخته می‌شود.

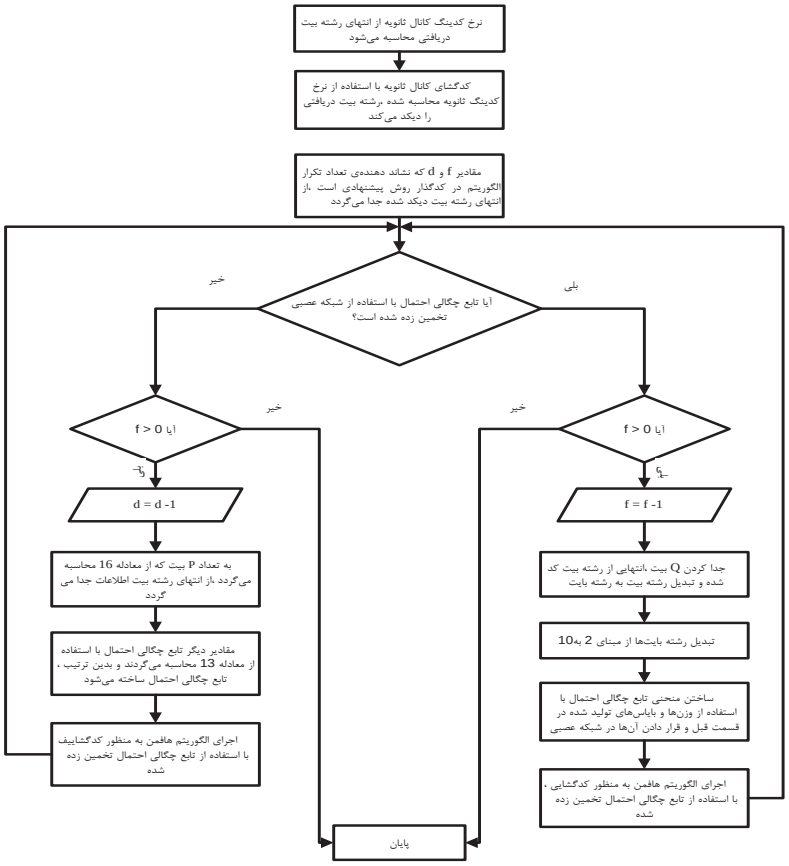
• مرحله ۹: با توجه به تابع چگالی احتمال به‌دست آمده رشته اطلاعات را با استفاده از الگوریتم هافمن رمزگشایی کنید.

• مرحله ۱۰: این الگوریتم به تعداد $f+d$ مرتبه تکرار می‌شود. اگر تعداد تکرار کمتر از $f+d$ باشد به مرحله ۶ پرش کنید در غیر این صورت الگوریتم رمزگشایی پایان می‌یابد.

۸. ارزیابی روش پیشنهادی

ابتدا به معرفی چندین پارامتر استفاده شده در شبیه‌سازی پرداخته می‌شود و سپس نتایج به‌دست آمده را با نتایج چند مقاله مقایسه می‌نماییم. کانال استفاده شده در این مقاله کانال باینری متقارن[1] می‌باشد که خطای به وجود آمده در این کانال شامل نویز سفید گوسی[2]، فیدینگ با تابع چگالی احتمال رایلی[3] است. در اولین مرحله الگوریتم پیشنهادی با دو الگوریتم IHVSM[4] PDWZ[5] برای سه ویدئو فورمن[6]، عمارت[7] و گارد ساحلی[8] [۲۶] مقایسه می‌شود که به ترتیب در مراجع [۲۷ و ۲۸] ذکر شده‌اند. در این روش‌ها خروجی بر روی ۳ قطعه ویدئو گفته شده محاسبه گردیده است. ویژگی قطعات استفاده شده بدین صورت است که فورمن دارای دقت فریمی ۲۸۸×۳۵۲ می‌باشد و عمق هر پیکسل آن ۸ بیت است. همچنین تعداد قاب ارسالی در واحد زمان برابر با ۳۰ قاب می‌باشد. همچنین تعداد قاب استفاده شده در این آزمایش ۳۰۰ قاب بوده است. دو قطعه ویدئویی استفاده شده عمارت و گارد ساحلی نیز دارای ویژگی‌هایی شبیه به فورمن می‌باشند. در شکل‌های (۲۱-۱۶)، روش پیشنهادی با دو روش IHVSM و PDWZ که همگی بر روی

[1] Binary Symmetric Channel (Bsc)
[2] Awgn
[3] Rayleigh Fading Pdf (Power Density Function)
[4] Iterative Horizontal-Vertical Scanline Model
[5] Pixel And Transform Domain Wyner-Ziv
[6] Foreman
[7] Hall
[8] Coastguard

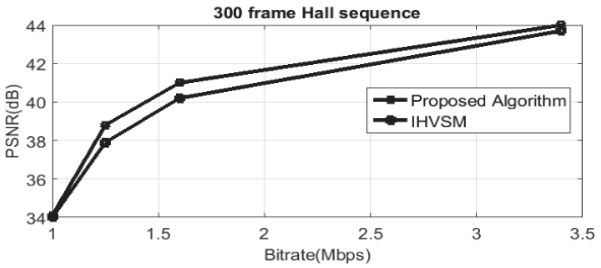


شکل ۱۵. فلوچارت رمزگشایی الگوریتم پیشنهادی

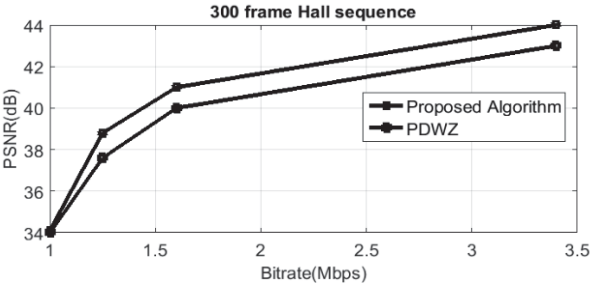
قطعه‌های ویدئویی فشرده شده با استاندارد H.264 مقایسه شده است و نتایج نشان دهنده بهبود کیفیت فایل‌های ویدئویی در روش پیشنهادی نسبت به روش‌های پیشنهاد در این دو مقاله می‌باشد. در مرحله بعد، الگوریتم پیشنهادی با الگوریتم MSSM[1] [۲۹] مقایسه می‌شود. در این روش از ۳ مجموعه ویدئویی روزنامه‌نگار[2] ترک لپ‌تاپ[3] و ویدئویی بیرونی[4] [۳۰] برای بررسی عملکرد الگوریتم مورد نظر استفاده شده است. این فایل‌های ویدئویی دارای ویژگی‌هایی می‌باشند که در ادامه به آن‌ها اشاره خواهد شد. ویدئو روزنامه‌نگار دارای دقت فریمی ۷۲۰×۹۶۰ است و تعداد قاب ارسالی در واحد زمان برابر با ۲۵ قاب می‌باشد. همچنین تعداد قاب استفاده شده در این آزمایش ۱۶۰۰ قاب است. ویدئو ترک لپ‌تاپ دارای رزولوشن فریمی ۷۶۸×۱۰۲۴ و تعداد قاب ارسالی در واحد زمان برابر با ۱۰۰ قاب در ۶ ثانیه و یا متوسط ۱۶/۶ قاب در واحد زمان است. همچنین تعداد قاب استفاده شده در این آزمایش ۱۰۰ قاب است. ویدئو بیرونی نیز ویژگی‌هایی شبیه به ترک کردن لپ‌تاپ دارد. در شکل‌های (۲۴-۲۲)، روش پیشنهادی با روش MSSM بر روی ۳ قطعه ویدئوی فشرده شده با استاندارد H.264 مقایسه شده است و نتایج نشان دهنده بهبود کیفیت فایل‌های ویدئویی در روش پیشنهادی نسبت به روش پیشنهاد شده در این مقاله می‌باشد. در انتها الگوریتم پیشنهادی با الگوریتم WSVC[5] برای رشته ویدئویی فورمن مقایسه می‌شود. نتیجه مقایسه در شکل (۲۵) نشان داده شده است. همان‌طور که در شکل (۲۵) مشاهده می‌شود برای بیشتر قاب‌ها روش پیشنهادی کیفیت بهتری نسبت به روش WSVC داشته است و تنها در برخی از قاب‌ها میزان کیفیت نسبت به الگوریتم WSVC بهبود زیادی پیدا نکرده است که می‌توان آن را به دلیل افزایش آنتروپی در آن قاب که به عنوان مرز شات می‌باشد، توجیه نمود.

این شبیه‌سازی را برای سه نرخ ارسال منبع پیاده‌سازی شده و با توجه به اینکه تعداد قاب ارسالی ثابت است (نرخ کدکننده منبع ثابت است)، هرچه نرخ ارسال بالاتر باشد، در نتیجه نرخ رمزگذار کانال بالاتر رفته و قاب‌های ویدئویی در مقابل نویز مقاوم‌تر می‌شوند. نرخ رمزگذار کانال R_c و همچنین نرخ کدگذار کانال ثانویه $R_c^{'}$ برای دو فایل ویدئویی با نرخ‌های ارسال مختلف نشان داده شده است. در انتها پس از انجام مقایسه بین روش پیشنهادی و چندین روش جدید به این نتیجه رسید که روش پیشنهادی توانسته است بدون افزایش نرخ بیت ارسالی کیفیت قاب‌های دریافتی را افزایش دهد که این امر در نمودارهای نشان داده شده، قابل مشاهده است و می‌توان نمودارهای انتهایی را دلیل برتری روش پیشنهادی با روش‌های دیگر دانست.

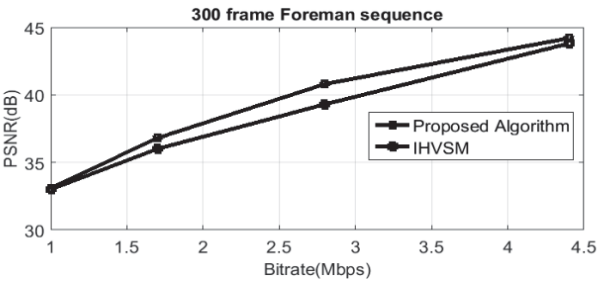
[1] Mesh-Structured Source Model
[2] Newspaper
[3] Leaving-Laptop
[4] Outdoor
[5] Wireless Scalable Video Coding



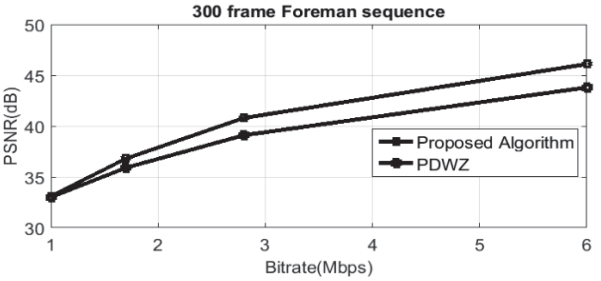
شکل ۱۶. مقایسه PSNR برای تمامی قاب‌های بین عمارت روش پیشنهادی و الگوریتم IHVSM برای نرخ بیت‌های متفاوت



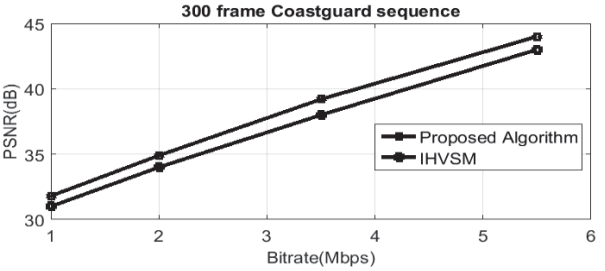
شکل ۱۷. مقایسه PSNR برای تمامی قاب‌های عمارت بین روش پیشنهادی و الگوریتم PDWZ برای نرخ بیت‌های متفاوت



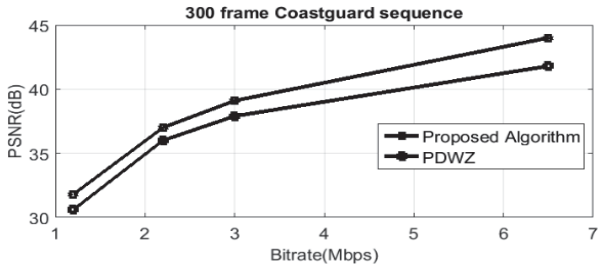
شکل ۱۸. مقایسه PSNR برای تمامی قاب‌های فورمن بین روش پیشنهادی و الگوریتم IHVSM برای نرخ بیت‌های متفاوت



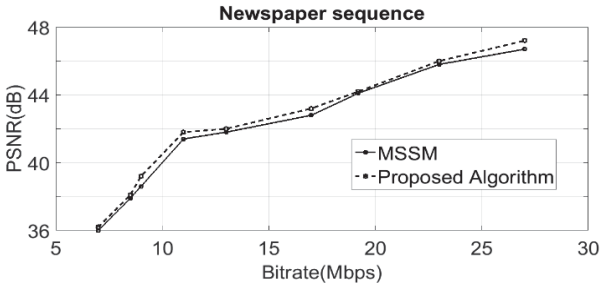
شکل ۱۹. مقایسه PSNR برای تمامی قاب‌های فورمن بین روش پیشنهادی و الگوریتم PDWZ برای نرخ بیت‌های متفاوت



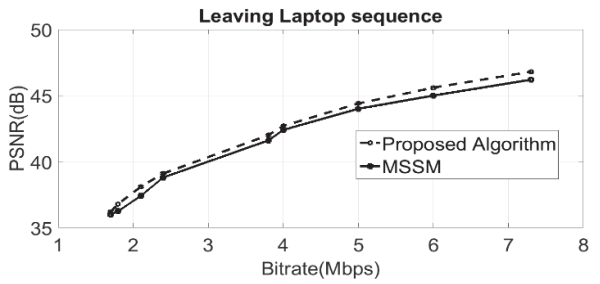
شکل ۲۰. مقایسه PSNR برای تمامی قاب‌های گارد ساحلی بین روش پیشنهادی و الگوریتم IHVSM برای نرخ بیت‌های متفاوت



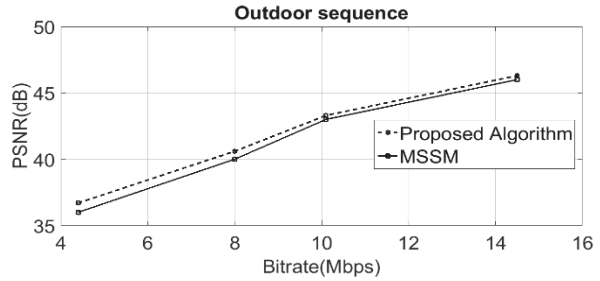
شکل ۲۱. مقایسه PSNR برای تمامی قاب‌های گارد ساحلی بین روش پیشنهادی و الگوریتم PDWZ برای نرخ بیت‌های متفاوت



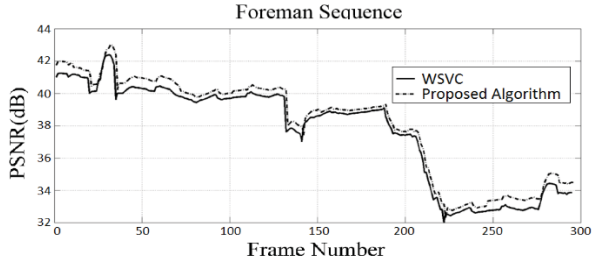
شکل ۲۲. مقایسه PSNR برای تمامی قاب‌های روزنامه نگار بین روش پیشنهادی و الگوریتم MSSM برای نرخ بیت‌های متفاوت



شکل ۲۳. مقایسه PSNR برای تمامی قاب‌های ترک ویدئو بین روش پیشنهادی و الگوریتم MSSM برای نرخ بیت‌های متفاوت



شکل ۲۴. مقایسه PSNR برای تمامی قاب‌های بیرونی بین روش پیشنهادی و الگوریتم MSSM برای نرخ بیت‌های متفاوت



شکل ۲۵. مقایسه PSNR برای تمامی قاب‌های فورمن بین روش پیشنهادی و الگوریتم WSVC برای ۳۰۰ قاب ویدئو فورمن

۹. نتیجه‌گیری

در روش پیشنهادی PSNR قاب‌های ویدیویی در گیرنده، دارای مقادیر بیشتری نسبت به روش‌های ارایه شده اخیر می‌باشد. این بدان معنا است که قاب‌های دریافتی با روش پیشنهادی دارای کیفیت بالاتری نسبت به آن روش‌ها می‌باشند. در روش پیشنهادی به دلیل فشرده‌سازی بیت‌های به‌دست آمده از بلوک VLC و جایگزین کردن بیت‌های ساخته شده توسط کدگذار کانال ثانویه (که از تفاضل بین بیت‌های ساخته توسط بلوک VLC و بیت‌های ساخته شده پس از فشرده‌سازی به‌دست می‌آیند) توانستیم کیفیت قاب‌های ویدیویی دریافتی را نسبت به روش‌های جدید افزایش دهیم. همانطور که در شکل‌های ۱۶ تا ۲۵ نشان داده شده است، روش پیشنهادی توانسته است حدود ۱ تا ۴ dB کیفیت ویدیوی دریافتی را افزایش دهد. این افزایش کیفیت برای نرخ ارسال‌های بالاتر بیشتر از نرخ ارسال‌های پایین است. روش پیشنهادی با افزایش دادن نرخ کدکردن کانال و در نتیجه محافظت بیشتر از بیت‌های ارسالی، ضمن ثابت نگه داشتن نرخ کلی ارسال، توانسته است کیفیت قاب‌های ویدیویی دریافتی را بهبود بخشد. این امر منجر به افزایش مقاومت فایل‌های ویدیویی نسبت به نویز و اعوجاج کانل گردیده است که می‌توان نتایج آن را در با بررسی شکل‌های ۱۶ تا ۲۵ بررسی نمود.

۱۰. مراجع

[1] Robert, E.; Van, D.; David, J. “Transport of Wireless Video Using Separate, Concatenated and Joint Source Channel Coding”; In Proc. of the IEEE 1999, 87, 1734–1750.

[2] Flierl, M.; Girod, B. “Video Coding with Superimposed Motion-Compensated Signals”; Kluwer Academic, 2004.

[3] Bystrom, M; Modestino, J. W. “Combined Source-Channel Coding Schemes for Video Transmission over an Additive White Gaussian Noise Channel”; IEEE J. Sel. Areas Communication 2000, 18, 880–890.

[4] Cheung, G.; Zakhor, A. “Bit Allocation for Joint Source/Channel Coding of Scalable Video”; IEEE Trans. Image Process 2000, 9, 340–356.

[5] Kondi, L. P.; Ishtiaq, F.; Katsaggelos, A. K. “Joint Source-Channel Coding For Motion-Compensated Dct-Based Snr Scalable Video”; IEEE Trans. Image Process 2002, 11, 1043–1052.

[6] Zhai, F.; Eisenberg, Y.; Pappas, T.; Berry, R.; Katsaggelos, A. “Rate-Distortion Optimized Hybrid Error Control for Real-Time Packetized Video Transmission”; IEEE Trans. Image Process 2006, 15, 40–53.

[7] Salomon, D. “Data Compression”; Third Ed., Springer, 2004.

[8] Wenger, S. “H.264/AVC Over IP”; IEEE Trans. Circuits Syst. Video Tech. 2003, 13, 645–656.

[9] Wiegand, T.; Sullivan, G. J.; Bjonte, G.; Luthra, A. “Overview of the H.264/AVC Video Coding Standard”; IEEE Trans. Circuits and Syst. for Video Tech. 2003, 13, 560–576.

[10] List, P.; Joch, A.; Lainema, J.; Bjøntegaard, G.; Karczewicz, M. “Adaptive Deblocking Filter”; IEEE Trans. Circuits Syst. Video Tech. 2003, 13, 614–619.

[11] Shi, Y. Q.; Sun, H. “Image and Video Compression for Multimedia Engineering, Fundamentals, Algorithms, and Standards”; CRC-Press, 1999.

[12] Richardson, I. E. G. “H.264 and MPEG-4 Video Compression, Video Coding for Next-Generation Multimedia”; Wiley, New York, 2003.

[13] MPEG Video Group Standard “MPEG-4 Video Verification Model Version 11.0”; in ISO/IEC JTC1/SC29/WG11 MPEG98/N2172, Tokyo, Japan, 1998.

[14] Hagenauer, J.; Lutz, E. “Forward Error Correction Coding for Fading Compensation in Mobile Satellite Channels”; IEEE J. on Communication. 2007, 5, 215–225.

[15] Wang, C.; Sklar, D.; Johnson, D. “Forward Error-Correction Coding”; The Aerospace Corporation Magazine of Advances in Aerospace Technology, 2002.

[16] Shannon, C. E. “Communications in the Presence of Noise”; Proc. IEEE Theory of Communication. 1998, 86, 447–458.

[17] Carlson, B. “Communication Systems. An Introduction to Signals and Noise in Electrical Communication”; 3rd Ed., McGraw-Hill, New York, 1986.

[18] Shannon C. E. “A Mathematical Theory of Communication”; Bell. Syst. Tech. J. 1948, 27, 379–423.

[19] Vemuri, V. “Artificial Neural Networks, Theoretical Concepts, Computer Society Press Technology Series, Computer Society of Institute of Electrical and Electronics Engineers”; Washington, DC, 1988.

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

[21] Cobourn, W. G.; Dolcine, L.; French, M.; Hubbard, M. C. “A Comparison of Nonlinear Regression and Neural Network Models for Ground-Level Ozone Forecasting”; J. Air & Waste Manage 1999-2009

[22] Stone, M. “Cross-Validatory Choice and Assessment of Statistical Predictors”; J. Royal Statistical Soc. 1974, 36, 111-147.

[23] Fukunaga, K.; Hummels, D. M. “Leave-one-out procedure for nonparametric error estimate”; IEEE Transactions on Pattern Analysis and Machine Intelligence 1989, 11, 421-430.

[24] Proakis, J. G. “Digital Communications”; Mcgraw Hill, Hardcover, 1995.

[25] Huffman, D. A. “A Method for the Construction of Minimum Redundancy Codes”; in Proc. IRE. 1952, 1098-l 101.

[26] “Database Video Clip”; http://see.xidian.edu.cn/vipsl/database-Video.html.

[27] Yongkai, H.; Wang, T.; Maunder, R.; Hanzo, L. “Two-Dimensional Iterative Source-Channel Decoding for Distributed Video Coding”; IEEE Communication 2014, 18, 1, 90-94.

[28] Brites, C.; Pereira, F. “Correlation Noise Modeling for Efficient Pixel and Transform Domain Wyner-Ziv Video Coding”; IEEE Trans. Circuits and Systems for Video Tech. 2008, 18, 1177-1190.

[29] Huo, Y.; Wang, T.; Maunder, G.; Hanzo, J. “Motion-Aware Mesh-Structured Trellis for Correlation Modelling Aided Distributed Multi-View Video Coding”; IEEE Trans. Image Processing 2014, 23, 319–331.

[30] “Outdoor Video Clip Database”; http://sp.cs.tut.fi/mobile3dtv /video-plus-depth/