Um Método para Melhorar a QoE em Sistemas de Transmissão IPTV

Carlos Eduardo Maffini Santos e Carlos Marcelo Pedroso

***Resumo*— A televisão transmitida através do protocolo IP (IPTV) está entre as mais promissoras tecnologias para entrega multimídia, permitindo um alto nível de interatividade com o usuário e integração com a internet. A transmissão de fluxos multimídia em tempo real requer garantia de recursos, como limitada perda de pacotes, largura de banda e baixo atraso e jitter para assegurar um bom nível de QoE (*Quality of Experience*). Este artigo propõe o uso de uma estratégia de descartes de pacotes prioritária, juntamente com um reconhecedor de carga útil, implementado com redes neurais artificiais, para evitar o descarte de pacotes transportando informações relevantes para a reconstrução da imagem. Mostra-se que o método proposto apresenta um melhor desempenho se comparado as abordagens existentes atualmente.**

I. Introdução

O IPTV (*Internet Protocol television*) é um serviço multimídia de entrega de TV/vídeo/áudio/dados sobre uma rede baseada no protocolo IP (*Internet Protocol*). O tráfego gerado pelos sistemas IPTV apresenta um comportamento em rajada [1], devido aos algoritmos utilizados para a codificação dos vídeos (codecs) possuírem características de taxa variável (VBR - *Variable Bit Rate*). Adicionalmente, a literatura reporta um comportamento auto similar do tráfego resultante a nível de pacotes.

Tal comportamento auto similar pode ocasionar congestionamentos nas filas dos roteadores, levando possíveis perdas de pacotes, mesmo com níveis de utilização relativamente baixos, impactando negativamente na QoE (*Quality of Experience*). Mesmo a mínima perda de pacotes em um fluxo de vídeo pode resultar em uma degradação da qualidade [2], e 1% ou menos de pacotes perdidos poderiam afetar severamente a qualidade da imagem [3].

Os algoritmos de codificação de vídeo MPEG-2 e MPEG-4 são um dos mais utilizados atualmente. O MPEG é uma família de padrão internacional aberto que fornece ferramentas para o uso em aplicações multimídia [4], que incluem codecs de áudio, vídeo e gráficos. O MPEG-4 possui a vantagem de exigir menores taxas de transmissão quando comparado com seus antecessores, MPEG-1 e MPEG-2. O algoritmo MPEG representa a sequência de quadros que compõe a imagem por um grupo de figuras (GOP - *Group of Pictures*), consistindo em uma sequência específica de quadros para representar cada cena do vídeo. O GOP sempre é iniciado por um quadro I (*Intra Coded Frame*), que é decodificado sem necessidade de informações contidas em outros quadros, seguido pelos quadros P (*Predictive-Frame*) e quadros B (*Bidiretional-Frame*). Os quadros P, para serem decodificados, dependem das informações dos quadros I ou P anteriores mais próximos e os quadros B usam informações dos quadros P e I mais próximos, tanto os passados quando os futuros, como referência para a decodificação da imagem. A sequência dos quadros depende dos ajustes realizados durante o processo de codificação. A notação mais comum utiliza o par (x, y), onde x indica o número de quadros do GOP e y o número de quadros B entre os quadros P.

Departamento de Engenharia Elétrica, Universidade Federal do Paraná, Curitiba-PR. E-mail: carlos.maffini@pucpr.br, pedroso@eletrica.ufpr.br

O impacto da perda de pacotes na QoE foi estudado por Greengrass et al. em [3]. Os autores mostram que o descarte de pacotes que transportam quadros I podem resultar em distorções na imagem que são propagadas por todos os quadros ao longo do mesmo GOP. A degradação pode durar um longo período de tempo (tipicamente 0,5 a 1 segundo), sendo que a qualidade do vídeo será recuperada apenas quando o decodificador receber um novo quadro I intacto. Dependendo de qual pacote é perdido, as distorções podem resultar em altos níveis de degradação, por exemplo, a perda de único pacote IP no início de um quadro I contendo parte do cabeçalho do quadro pode ter o mesmo efeito como o de perder um quadro I por inteiro. Também é mostrado que quanto maior o número de quadros do GOP, pior o efeito da perda de um pacote do quadro I.

Para melhorar o nível da qualidade de experiência percebida pelo usuário, Hong et al. propuseram em [5] um método, a ser implementado pelo servidor que realiza a transmissão do fluxo de vídeo, chamado SAPS (*Significance Aware Packet Scheduling*), que ajusta os intervalos de tempo entre os pacotes baseado na significância da informação que ele transporta. Por padrão, os quadros são gerados a um intervalo fixo de tempo. A proposta dos autores é aumentar o intervalo de tempo entre pacotes com nível maior de significância, alterando assim o comportamento em rajada resultante no tráfego. Isso possibilita que os roteadores liberem algum espaço em seus *Buffers* antes da chegada desses quadros. O nível de significância é obtido a partir da relação sinal ruído de pico (PSNR-*Peak signal-to-noise ratio*), que é calculado com a interpretação do impacto da perda de cada bit do quadro no PSNR, considerando a estrutura de dependência do GOP. Como resultado, a qualidade de experiência percebida pelo usuário é aumentada. Hong et al. comparam seus resultados com dois algoritmos conhecidos, o *Size Based Packet Scheduling* (SBPS) e utilizando uma abordagem de melhor esforço (*Best-Effort*, BE), ambos com descarte de fim de fila (*Drop Tail*), sendo que o SBPS e o BE apresentam desempenhos semelhantes, enquanto que o SAPS leva a uma melhor significância na QoE se comparado com este dois métodos.

Neste artigo propomos um método para melhorar a QoE, a ser implementado nos roteadores que compõe uma rede de

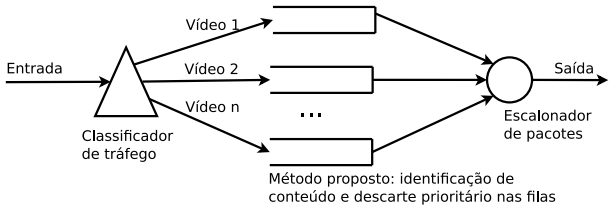

Fig. 1. Cenário de uso do método proposto: cada fluxo de vídeo deve ser classificado em uma fila independente nos roteadores da rede

transmissão para sistemas IPTV. O método proposto realiza o reconhecimento da carga útil dos pacotes em fluxos individuais de vídeo e, em caso de congestionamento, realiza um descarte seletivo de pacotes. Como premissa básica de operação, os roteadores devem estar preparados para realizar a separação do tráfego por fluxo de vídeo, que deve ser classificado em uma fila específica, conforme ilustrado pela Figura 1. Para realizar o reconhecimento do tipo de quadro que um pacote transporta foram consideradas duas alternativas: (a) reconhecimento de pacotes transportando quadros I, evitando o seu descarte (b) reconhecimento de pacotes transportando quadros B, priorizando o seu descarte. A abordagem (a) foi escolhida porque caso o reconhecedor não seja preciso, alguns pacotes B e P seriam preservados, enquanto que na abordagem (b) um erro de reconhecimento implicaria em um possível descarte de um pacote I, o que deve ser evitado. Redes neurais foram utilizadas no reconhecimento devido a sua baixa complexidade computacional, permitindo sua implementação em roteadores. O principal benefício introduzido é a melhoria da QoE percebida com menor complexidade computacional se comparado com o método proposto por [5].

Além desta seção introdutória, este artigo está estruturado da seguinte maneira. A Seção II descreve as técnicas usadas para o reconhecimento da carga útil dos pacotes, as topologias das redes neurais utilizadas, a origem do conjunto de dados em estudo e os resultados de reconhecimento. A Seção III apresenta o método proposto para descarte de pacotes e os resultados obtidos. Finalmente a conclusão e trabalhos futuros na Seção IV.

II. Reconhecimento da Carga Útil dos Pacotes

Para realizar o reconhecimento da carga útil dos pacotes foram utilizadas redes neurais artificiais. Esse escolha justifica-se por elas serem ferramentas capazes de resolver problemas complexos de previsão e reconhecimento de séries temporais, além da possibilidade de serem implementadas em sistemas de tempo real devido à sua baixa complexidade computacional. De acordo com [6], as redes neurais fornecem um conjunto de algoritmos não lineares para extração de características e classificação, podendo ser mapeados e implementados com eficiência em *hardware*.

Os parâmetros de entrada utilizados nas redes neurais foram os intervalos de tempo entre pacotes sucessivos, δ_k, e o tamanho de cada pacote, ρ_k, $1 \leq k \leq N$, onde N representa o tamanho da janela. A saída da rede neural é um único parâmetro y, $0 \leq y \leq 1$, onde a saída 1 representa a existência de um pacote I no conjunto de entradas e a saída 0 representa

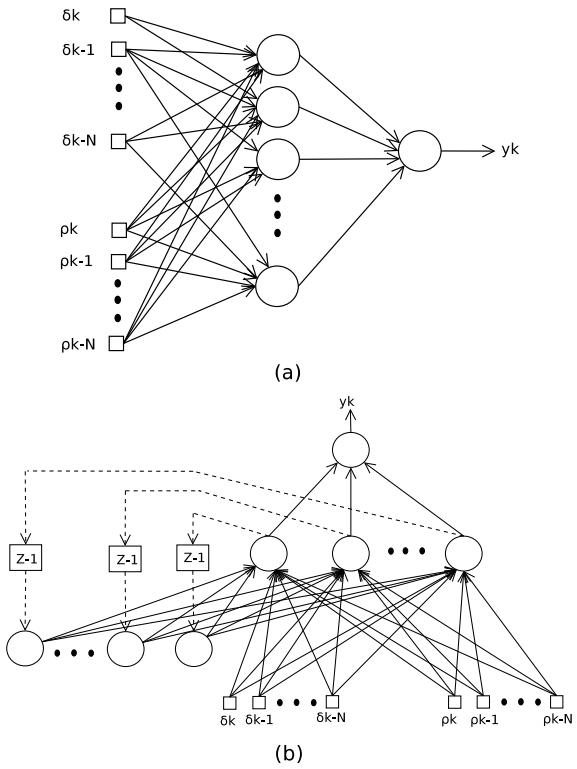

Fig. 2. (a) Rede FFTD e (b) Rede ER usando o método de aproveitamento de atraso.

a ausência. Portanto, a entrada da rede será composta por δ_{k-1}, δ_{k-2}, ..., δ_{k-N}, ρ_{k-1}, ρ_{k-2}, ..., ρ_{k-N}. Desta forma, o número de entradas da rede será de $2N$. Para o treinamento das redes neurais, os dados foram divididos em dois conjuntos: o primeiro, composto por 70% do total, usado para o treinamento e o conjunto restante, utilizado no processo de validação.

A. Topologias de Redes Neurais em Estudo

Foram utilizadas duas topologias de redes neurais: (a) rede *Feed-Forward* com aproveitamento de atraso (*Feed-Forward with Tapped Delay*, FFTD) e a (b) rede de Elman Recorrente (ER), ambas utilizando o método de aproveitamento de atraso, principalmente devido a simplicidade da rede FFTD e aos bons resultados reportados na literatura pela rede ER no reconhecimento de séries temporais [7].

A Figura 2 (a) e (b) apresenta as respectivas estruturas das redes neurais FFTD e ER. Em ambas arquiteturas existem N entradas, uma camada escondida e uma camada de saída com um neurônio. A saída reporta se os pacotes da janela N transportam informações de quadros I ou não. Adicionalmente, a rede ER possui uma camada de contexto, onde o número de neurônios utilizados foi o mesmo da camada escondida. O número de neurônios da camada escondida foi estabelecido pelo uso da média aritmética entre o número de neurônios de entradas e saídas.

O tamanho da janela N é fundamental no sucesso do reconhecimento. Se N for menor que o número de pacotes de um quadro I, a rede neural poderia não reconhecer a presença de um quadro I devido a falta de dados de entrada. Se N for maior que o tamanho do GOP, a janela necessariamente

irá conter um quadro I, tornando sem sentido a abordagem planejada, pois a saída da rede neural seria sempre 1. Desta forma, foram realizados testes utilizando-se sempre N maior que o número mínimo de pacotes de um quadro I e menor que o tamanho do GOP. Busca-se o menor tamanho de janela N possível, o que torna o reconhecedor mais preciso.

B. Origem dos Dados

Os vídeos usados para os testes estão publicamente disponíveis em [8], todos com resolução de 352x288 linhas, que são frequentemente utilizados por outros autores no estudo de sistemas de imagem, como por [7], [9], [4] e [3]. Todos os vídeos foram codificados com o codec MPEG-4 com a ferramenta *ffmepg* [10], com configuração de GOP (12,2), resultando na sequência de quadros dada por: IBBPBBPBBPBB.

Outra ferramenta utilizada foi o *mp4trace* [9], que realiza transmissão de vídeos MPEG-4, escolhida pela sua capacidade de identificar o tipo da carga útil dos pacotes que estão sendo enviados pela rede (I, P ou B), permitindo a montagem dos conjuntos de dados para o treinamento e validação das redes neurais. Para capturar δ_k e ρ_k, os vídeos foram transmitidos através de uma rede Ethernet não congestionada e os dados foram capturados com ferramentas de monitoramento de tráfego *Tcpdump* e *Wireshark*.

TABELA I
SUMÁRIO DE ESTATÍSTICAS BÁSICAS DOS VÍDEOS UTILIZADOS

Vídeo	Quadros	Tam. médio dos quadros (bytes)	Nº de pacotes	Tam. médio dos pacotes (bytes)	Duração (s)
Highway	2001	13016	18810	1416	66
Bridge Far	2101	12247	18637	1403	70
Coast Guard	300	20514	4360	1448	10
Paris	1065	11413	8845	1408	35
Soccer	300	15575	3345	1431	10

A Tabela I sumariza as principais características dos vídeos utilizados, apresentando a quantidade total e o tamanho médio dos quadros, quantidade total e o tamanho médio dos pacotes e o tempo de duração em segundos. A escolha dos vídeos foi feita devido as suas características, variando entre imagens estáticas e dinâmicas, resultando em vários níveis de tráfego em rajada.

C. Treinamento e Validação do Reconhecedor

Os testes experimentais foram feitos com o simulador de redes neurais javaNNS (Java Neural Network Simulator), desenvolvido pelo *Wilhelm-Schickard-Institute for Computer Science* (WSI) [11]. A escolha deste simulador deve-se à sua confiabilidade e ao grande número de algoritmos de treinamento e de topologias suportadas, além da capacidade de gerar código em linguagem C, facilitando a implementação futura do simulador de filas.

As redes neurais foram treinadas com o algoritmo de retropropagação padrão (*BackPropagation*). O algoritmo de retropropagação é o mais famoso entre os algoritmos de aprendizado, podendo ser especialmente utilizado em casos de conjuntos de treinamento com muitos exemplos [12], como ocorre no problema em questão.

Os parâmetros do algoritmo de treinamento, *dmax* (diferença máxima entre o valor de aprendizado e o valor encontrado pela saída do neurônio) e η (taxa de aprendizado) foram ajustados, de maneira empírica, respectivamente em 0,01 e 0,1. Tipicamente, o *dmax* deve ser ajustado em valores de 0 a 0,2, de acordo com o erro desejado. O parâmetro η indica o tamanho do passo de ajuste dos pesos sinápticos entre as conexões dos neurônios para cada ciclo de treinamento. Quanto menor a taxa de aprendizado, menor serão os ajustes dos pesos sinápticos, porém um tempo de treinamento consideravelmente longo é demandado. O ajuste de 0,1 para η foi realizado devido ao tempo de treinamento não ser importante para a aplicação em consideração, por ser um processo *off-line*. A quantidade de ciclos de treinamento foi configurado em 50.000, em razão à observação de uma sensível redução no erro após 5.000 ciclos de treinamento. Todos os neurônios foram configurados com a função de ativação sigmoidal, que possui características muito interessantes, dentre elas, o fato de permitir capturar características não lineares do processo [13].

D. Resultados do Reconhecimento

Foram realizados testes com tamanhos de janelas de 15, 25, 35, 45 e 55. O tamanho da janela foi escolhido devido a estrutura do GOP dos vídeos, que possuem tamanho médio de 125 pacotes, tipicamente com 15 pacotes transportando um quadro I e em torno de 10 pacotes para os quadros P e B.

As Tabelas II e III mostram a porcentagem de pacotes transportando quadros I reconhecidos pelas redes neurais testadas. Os resultados indicam que, para os vídeos em análise, foi possível reconhecer a carga útil dos pacotes com um alto grau de acerto. Ambas topologias (FFTD e ER) atingiram bons resultados. A porcentagem de reconhecimento melhora com o aumento da janela, o que era esperado. Também pode ser notado o fraco desempenho da janela $N = 15$, porque este é o número necessário de pacotes para transportar um quadro I, não tendo a rede neural um número suficiente de parâmetros para identificar a transição entre os quadros. Os resultados mostram um melhor desempenho com $N \geq 25$; com $N = 25$ a porcentagem de erro de reconhecimento foi em média de 2.1%, bastante aceitável para a aplicação planejada.

TABELA II
PORCENTAGEM DE PACOTES TRANSPORTANDO INFORMAÇÕES DE QUADROS I RECONHECIDOS PELA REDE FFTD

N	*Highway*	*Bridge Far*	*Coast-Guard*	*Paris*	Video *Soccer*	Média
15	16.7%	31.5%	21.7%	95.2%	50%	43.2%
25	98.9%	100%	90.9%	100%	100%	97.9%
35	100%	100%	100%	100%	100%	100%
45	100%	100%	100%	100%	100%	100%
55	100%	100%	96.4%	100%	100%	99.3%

TABELA III
PORCENTAGEM DE PACOTES TRANSPORTANDO INFORMAÇÕES DE QUADROS I RECONHECIDOS PELA REDE ER

N	*Highway*	*Bridge Far*	*Coast-Guard*	*Paris*	Video *Soccer*	Média
15	86%	80.6%	65%	98%	90%	83.9%
25	96%	99.5%	85%	99%	90.6%	94%
35	98%	99%	85.7%	100%	96%	95.7%
45	99%	99%	93%	100%	100%	98.2%
55	98.5%	100%	93%	100%	100%	98.3%

III. DESCARTE DE PACOTES

O algoritmo de descarte de pacotes padrão utilizado atualmente é o descarte de cauda (*Drop Tail*): quando a fila está com sua capacidade máxima esgotada, os novos pacotes que chegam são descartados. Outras opções populares de algoritmos de gerência de filas são o RED (*Randon Early Detection*) e o WRED (*Weighted RED*), os quais descartam os pacotes antes mesmo que a fila atinja sua capacidade máxima, como aviso aos mecanismos de controle de congestionamento, atuando sobre as fontes de tráfego, para que essas reduzam suas taxas de transmissão. No entanto, nenhum destes algoritmos foi projetado para ser sensível à carga útil dos pacotes, de forma que os testes serão realizados apenas com o *Drop Tail*. Os resultados serão comparados também com o SAPS [5], que trata de problema semelhante.

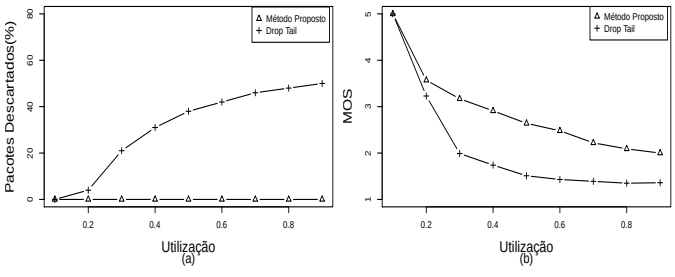

Fig. 3. (a) Porcentagem de pacotes I descartados e (b) avaliação de MOS para o vídeo Highway para vários níveis de utilização

A. Marcador de Pacotes e Descarte Prioritário Proposto

Propomos o uso de um mecanismo de descarte prioritário de pacotes para realizar descartes de acordo com a identificação feita pela rede neural, supondo que o tráfego agregado dos vídeos está sendo classificado em filas distintas. O método possui três etapas distintas: (1) o tempo entre chegada de pacotes sucessivos e tamanho dos N últimos pacotes recebidos são armazenados e utilizados como entrada da rede neural, para reconhecimento do tipo da carga útil dos pacotes; (2) a identificação é realizada, de acordo com a saída da rede neural, y_k, com $0 \leq y_k \leq 1$. Se $y_k > Lim_1$ é presumida a presença de pacotes contendo informações de quadros I, e neste caso os pacotes serão marcados como *verdes*. Caso $y_k < Lim_2$, é assumido que os pacotes na janela não carregam informações de quadros I, e eles serão marcados como *vermelhos*. Caso a saída esteja entre o intervalo $Lim_2 \leq y_k \leq Lim_1$, a presença de informações do quadro I não pode ser confirmada nem negada e os pacotes serão marcados como *amarelos*. Caso a capacidade da fila chegue ao seu limite, (3), o método proposto descarta primeiro os pacotes *vermelhos*, a seguir os *amarelos* e por último os *verdes*. Nos testes, Lim_1 e Lim_2 foram configurados com $0,2$ e $0,8$ - estes ajustes foram realizados empiricamente.

B. Resultados do Descarte

A eficiência do método proposto foi medida através de um simulador de filas, desenvolvido em linguagem C. O simulador permite a avaliação de desempenho de uma fila alimentada por um tráfego real. O simulador foi cuidadosamente validado comparando-se os resultados obtidos com modelos analíticos conhecidos. Para todos os testes, a configuração da taxa do enlace e o tamanho máximo da fila foram ajustados para obter-se uma situação de não congestionamento, mas próximo dos limites de congestionamento. Neste caso, a característica em rajada dos vídeos pode ocasionar perda de pacotes por um tempo limitado.

A Figura 3 (a) e (b) apresenta, respectivamente, a porcentagem de pacotes transportando quadros I descartados e a avaliação do MOS (*Mean Opinion Score*) para vários níveis de utilização da fila, para um tamanho fixo de fila, para o vídeo Highway, se comparado com o *Drop Tail*. A fim de avaliar o QoE, o MOS foi estimado usando o conjunto de ferramentas do Evalvid [9]. O MOS é uma das métricas mais usadas na estimativa de QoE e é expressa por um número, 1 sendo a pior e 5 a melhor qualidade percebida. O Evalvid realiza a comparação da imagem do vídeo original com o vídeo reconstituído, efetuando o cálculo do PSNR para estimar o MOS. Observa-se que o método proposto apresenta um melhor nível de QoE para os diversos níveis de utilização, com um numero de pacotes I perdidos muito abaixo se comparado com o *Drop Tail*. A degradação do MOS apresentada com o método proposto é causada pela degradação do atraso, do *jitter* e da perda dos pacotes P e B. A Figura 4 (a) e (b) apresenta a porcentagem de pacotes transportando quadros I descartados em função da variação do tamanho máximo da fila, com um nível de utilização do enlace de 0,9, para os vídeos Highway e Bridge Far. Para cada figura, duas linhas são apresentadas, comparando o desempenho do método proposto com o *Drop Tail*. Em todos os casos, o método proposto apresenta um melhor desempenho. A Figura 4 (c) e (d) apresenta a avaliação do MOS para o mesmos vídeos. Nota-se um melhor QoE para o método proposto, por exemplo, na situação em que o *Drop Tail* atinge um MOS de $2,8$, o método proposto atinge $4,5$ para um tamanho de fila de 6000 bytes para o vídeo Bridge Far.

A Figura 5 (a) e (b) mostra uma estimativa do PSNR, calculado por Hong et al. [5], para dois vídeos usando os métodos SAPS, SBPS e BE, em função da variação do tamanho da fila. Para o caso do BE, os pacotes não sofrem processamento em relação a seus intervalos de tempo, e a resposta é resultado do descarte utilizando *Drop Tail* na fila. A mesma figura, (c) e (d) mostra o PSNR para dois vídeos utilizando o método proposto, também realizando a variação do tamanho da fila. O PSNR é proporcional ao MOS obtido, e foi utilizado aqui para permitir a comparação com o SAPS. Observa-se que o método proposto obtém ganhos semelhantes aos ganhos apresentados pelo SAPS, quando se toma como referência de ambos o desempenho do *Drop Tail* (o caso do SAPS é a curva com legenda *Best Effort*). A vantagem da utilização do método proposto em comparação com o SAPS é a maior complexidade computacional deste último, que exige que o valor da significância de cada pacote seja calculado estimando-se o prejuízo no PSNR resultante da perda de cada bit, em coordenadas (x, y), do quadro e o impacto resultante nos quadros dependentes. Como resultado, a complexidade computacional do SAPS aumenta na proporção de $O(n^2)$,

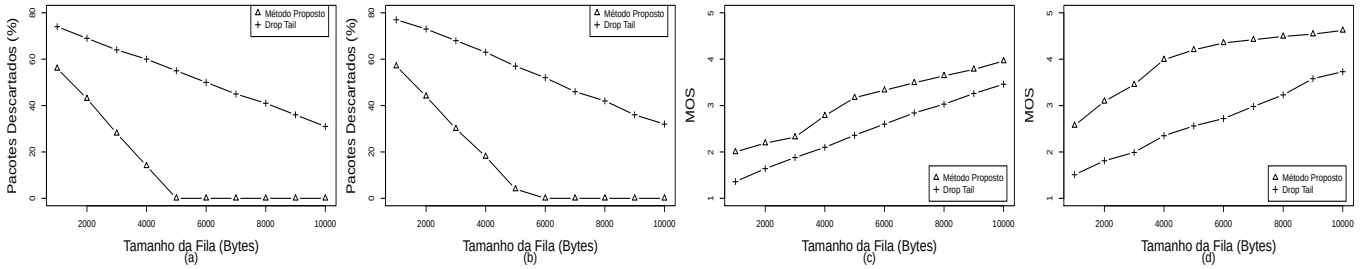

Fig. 4. Porcentagem de pacotes I descartados para os vídeos: (a) Highway e (b) Bridge Far e medida de MOS para os vídeos (c) Highway e (d) Bridge Far

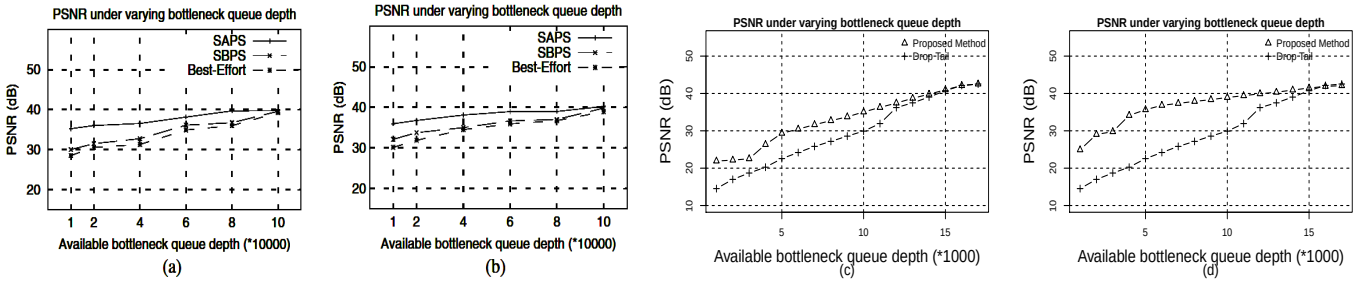

Fig. 5. Resultados apresentados pelo SAPS, (a) e (b) retirado de [5] e pelo método proposto, (c) e (d)

considerando que n é a resolução em pixels do filme (largura n e altura n) - ou seja, a complexidade aumenta em função do quadrado da resolução da imagem. Para o cálculo da significância é considerado o efeito acumulativo da perda de um pixel em todos os quadros do GOP. No método que propomos, a complexidade computacional depende praticamente do reconhecedor, que possui complexidade computacional, no pior caso, da ordem $O(n)$, sendo n o tamanho da janela utilizado (e não a dimensão da imagem propriamente dita).

IV. Conclusões e Trabalhos Futuros

Em sistemas IPTV o impacto na QoE devido ao descarte de pacotes pode ser severo, mesmo com porcentagens relativamente baixas de perdas. Os pacotes mais relevantes são aqueles que transportam informações de quadros I, devido à esses serem utilizados como referência na decodificação dos demais quadros do GOP. Mesmo com a capacidade da rede bem planejada, perdas de pacotes podem ocorrer devido a característica em rajada do tráfego de vídeo. O método padrão de gerência de filas é o *Drop Tail*, que não leva em consideração a importância de cada pacote no momento do descarte. Outras abordagens foram propostas, como o SAPS, que modifica as características do tráfego na origem, porém com complexidade computacional maior do que o método proposto e exigindo que seja realizada a decodificação do vídeo para estimar a significância de cada pacote.

Em nossa proposta, foi mostrado que é possível realizar o reconhecimento da carga útil dos pacotes utilizando-se redes neurais artificiais, o que permite a implementação de um método que evita o descarte de pacotes transportando quadros I, na camada de rede. Isso resulta em uma melhoria na qualidade de experiência (QoE) percebida pelo usuário em situações de congestionamentos. A abordagem proposta não exige a decodificação do vídeo, que permanece inalterado.

A sequência do trabalho, sendo executada neste momento, é estender os testes à filmes longos, com melhor qualidade. A verificação quantitativa dos quadros I preservados será realizada e também a qualitativa em função do MOS, para diversos cenários de congestionamento na rede.

Referências

[1] M. Dai, Y. Zhang, and D. Loguinov, "A unified traffic model for MPEG-4 and H.264 video traces," *IEEE Transactions on Multimedia*, vol. 11, no. 5, pp. 1010 –1023, aug. 2009.

[2] T. Szymanski and D. Gilbert, "Internet multicasting of IPTV with essentially-zero delay jitter," *IEEE Transactions on Broadcasting*, vol. 55, no. 1, pp. 20 –30, march 2009.

[3] J. Greengrass, J. Evans, and A. C. Begen, "Not all packets are equal, part 2: The impact of network packet loss on video quality," *IEEE Internet Computing*, vol. 13, pp. 74–82, March 2009.

[4] G. Van der Auwera, P. David, and M. Reisslein, "Traffic and quality characterization of single-layer video streams encoded with the H.264/MPEG-4 advanced video coding standard and scalable video coding extension," *IEEE Transactions on Broadcasting*, vol. 54, no. 3, pp. 698 –718, sept. 2008.

[5] S. Hong and Y. Won, "Incorporating packet semantics in scheduling of real-time multimedia streaming," *Multimedia Tools Appl.*, vol. 46, pp. 463–492, January 2010.

[6] T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *Communications Surveys Tutorials, IEEE*, vol. 10, no. 4, pp. 56 –76, quarter 2008.

[7] A. Abdennour, "Evaluation of neural network architectures for MPEG-4 video traffic prediction," *IEEE Transactions on Broadcasting*, vol. 52, no. 2, pp. 184 – 192, june 2006.

[8] A. S. University, "Video trace library." [Online]. Available: http://trace.eas.asu.edu/

[9] J. Klaue, B. Rathke, and A. Wolisz, "Evalvid-a framework for video transmission and quality evaluation," in *Proc. of 13th Int. Conf. on Modelling Tec. and Tools for Computer Perf. Eval.*, 2003, pp. 255–272.

[10] "FFMPEG documentation." [Online]. Available: http://ffmpeg.sourceforge.net/ffmpeg-doc.html

[11] I. Fischer, F. Hennecke, C. Bannes, and A. Zell, *Java Neural Network Simulator - User Manual - Version 1.1*, Wilhelm-Schickard-Institute for Computer Science - University of Tubingen, 2001.

[12] A. Zell, G. Mamier, and M. Vogt, "SNNS: Stuttgart neural network simulator - manual extensions of version 4.0," 2011. [Online]. Available: http://www.ra.cs.uni-tuebingen.de/SNNS/

[13] J. C. Principe, N. R. Euliano, and W. C. Lefebvre, *Neural and Adaptive Systems: Fundamentals Through Simulations*, Wiley, Ed. Principe, J. C., Dec 1999.