Predição de Viewports em Transmissão de Vídeo 360° com Aprendizado Profundo: Modelagem e Avaliação

Felipe Rosa[1], **Simone Ferlin**[2,3], **Joahannes B. D. da Costa**[1], **Bruno Kimura**[1]

[1]Universidade Federal de São Paulo, [2]Red Hat, [3]Karlstad University

{rosa.felipe, joahannes.costa, bruno.kimura}@unifesp.br, sferlin@redhat.com

Abstract. *360° immersive video streaming requires the prediction of future viewports from the angular motion of the Head-Mounted Display (HMD). Such predictions enable proactive content requests, mitigating playback rebuffering (stalls) caused by immersive user navigation and network instabilities. Despite advances in Deep Learning models, evaluation remains largely restricted to learning-domain metrics, namely regression errors, which are insufficient to reflect the actual impact and applicability of such predictors in real 360° streaming systems. In this context, we propose* `Viewport-P`*, a viewport prediction modelling framework aligned with the operations and metrics of immersive streaming systems. Based on this modelling, a complete prediction pipeline is implemented, integrating well-established Deep Learning models, including pure CNNs and hybrid CNN–GRU and CNN–LSTM architectures. The evaluation combines angular prediction error metrics with system-oriented metrics, in particular using* tile miss *events for spatial buffer efficiency. Experimental results show that spatial metrics capture more faithfully the effects of the prediction horizon and user dynamics, revealing consistent gains of hybrid CNN models in QoE-relevant scenarios.*

Resumo. *O streaming de vídeo imersivo 360° exige a predição de viewports futuras a partir do movimento angular do HMD (Head-Mounted Display), viabilizando requisições antecipadas e mitigando interrupções na reprodução (stalls) causadas pela navegação imersiva do usuário e por instabilidades de rede. Apesar do avanço de modelos de Aprendizado Profundo, a avaliação permanece majoritariamente restrita a métricas de domínio dos modelos de aprendizado, erros de regressão, o que é insuficiente para refletir o impacto e a aplicabilidade desses preditores em sistemas reais de streaming em 360°. Neste contexto, este trabalho propõe* `Viewport-P`*, uma modelagem de predição de viewports alinhada às operações e métricas de sistemas imersivos. Com base nessa modelagem, é implementado um pipeline completo de predição, integrando modelos de Aprendizado Profundo amplamente estabelecidos, como CNN puros e híbridos com GRU e LSTM. A avaliação combina métricas de erro angular com métricas orientadas ao sistema, em particular a eficiência espacial em buffer via eventos de tile miss. Os resultados mostram que métricas espaciais capturam de forma mais fiel o efeito do horizonte de predição e da dinâmica do usuário, evidenciando ganhos consistentes dos modelos CNN híbridos em cenários relevantes para QoE.*

1. Introdução

Nos últimos anos, a transmissão de vídeo pela Internet tem crescido de forma acelerada, atraindo atenção significativa da academia e da indústria. Estima-se que o tráfego de vídeo tenha superado $2/3$ do tráfego global de dados móveis em 2024[1], alcançando uma receita de US$ 199 bilhões em 2023[2]. Em particular, transmissões ao vivo e *webcasts* apresentam crescimento consistente, com 28,5% de usuários consumidores globais desse tipo de conteúdo semanalmente[3]. Esse avanço é impulsionado principalmente pelo HAS (*HTTP Adaptive Streaming*), paradigma dominante de distribuição de vídeo na Internet, no qual DASH[4] (*Dynamic Adaptive Streaming over HTTP*) e HLS[5] (*HTTP Live Streaming*) consolidaram-se como padrões de fato. Com a evolução para as Redes de Sexta Geração (6G), previstas para 2030 [Recommendation 2023], comunicações imersivas, como *streaming* de vídeo 360°, emergem como serviços fundamentais, exigindo taxas substancialmente superiores às do 5G, podendo atingir até 300 Mbps por usuário [Alidadi Shamsabadi et al. 2025].

Diferentemente do *streaming* sob demanda tradicional, transmissões em 360° impõem desafios adicionais sob condições dinâmicas de rede, uma vez que o sistema deve reagir rapidamente às interações do usuário, em especial às variações do Campo de Visão, i.e., FoV (*Field of View*). Para tanto, o vídeo 360° é particionado espacialmente em regiões retangulares (*tiles*), codificados em múltiplos níveis de qualidade. O FoV é então representado por uma janela de visualização, denominada *viewport*, composta por um conjunto de *tiles*. Dispositivos HMDs (*Head-Mounted Displays*) permitem navegação espacial por meio de *viewports* dinâmicos, diretamente associados ao FoV do usuário, gerando fluxos de controle reversos em tempo real e exigindo a entrega imediata do *viewport* correspondente. Embora DASH e HLS sejam amplamente adotados, seu modelo de requisições sequenciais e independentes de segmentos temporais mostra-se inadequado à navegação espacial interativa de vídeos 360° [Bentaleb et al. 2025]. A frequente troca de *viewports*, decorrente da movimentação do HMD, torna esses sistemas altamente suscetíveis a atrasos e interrupções, degradando a Qualidade de Experiência (QoE), sobretudo em cenários de rápida movimentação sob variações de latência e vazão da rede. O desafio central consiste em assegurar a disponibilidade antecipada dos *tiles* críticos no *player*, mesmo sob severas restrições de capacidade.

Diante disso, a predição do movimento do HMD é um componente crítico para a eficiência de sistemas de *streaming* de vídeo 360°, sobretudo para viabilizar a requisição antecipada de *tiles*. No entanto, o movimento HMD do usuário apresenta comportamento altamente não linear e estocástico, com degradação acentuada da acurácia à medida que o horizonte de predição aumenta. Embora a literatura proponha modelos sofisticados baseados em Aprendizado Profundo para esse problema, a avaliação concentra-se majoritariamente em métricas clássicas de erro de regressão (e.g., MAE e RMSE), que não capturam adequadamente o impacto sistêmico da predição sobre a eficiência de *buffer* e a QoE. Além disso, há pouca evidência empírica que relacione ganhos marginais em erro

[1] https://www.ericsson.com/en/reports-and-papers/mobility-report/reports/november-2024

[2] https://datareportal.com/reports/digital-2024-april-global-statshot

[3] https://datareportal.com/reports/digital-2024-april-global-statshot

[4] https://www.iso.org/standard/83314.html

[5] https://www.rfc-editor.org/info/rfc8216

angular com melhorias efetivas no desempenho de sistemas baseados em *tiles*, especialmente em horizontes de predição longos, que são os mais relevantes do ponto de vista operacional.

Para endereçar essa lacuna, este trabalho propõe `Viewport-P`, uma modelagem sistemática do *pipeline* de predição de *viewports*, implementada e avaliada em um ambiente controlado que combina métricas de regressão com métricas de desempenho para sistemas orientadas à comunicação multimídia. São investigadas arquiteturas CNN puras e modelos híbridos CNN–GRU e CNN–LSTM, como potenciais modelos preditores, considerando diferentes horizontes de predição e níveis de entropia do movimento HMD. Os resultados demonstram que, embora os ganhos em MAE e RMSE dos modelos híbridos sejam modestos e decrescentes em horizontes longos, esses modelos apresentam vantagens substanciais em métricas de eficiência espacial, como a taxa de *tile miss*, evidenciando que métricas sistêmicas são mais representativas do impacto prático da predição.

Diferente de estudos correlatos da literatura recente (§3), como principais contribuições deste trabalho, destacam-se:

- (i) A modelagem formal em `Viewport-P` para o *pipeline* completo de predição de *viewports*, i.e., da informação angular do movimento HMD capturado em domínio contínuo ao conjunto de *tiles* necessários em horizonte futuro para movimento HMD predito.
- (ii) A implementação do pipeline modelado que possibilitou a demonstração empírica de que métricas espaciais superam métricas de regressão na avaliação de QoE.
- (iii) A avaliação experimental, que levou a identificação de modelo preditor de maior potencial através da CNN híbrida com GRU, oferecendo desempenho comparável à LSTM, contudo, com menor complexidade computacional, o que é um facilitador para maior adequação e implantação em dispositivos HMD com recursos limitados.

O restante do texto está organizado em outras cinco seções. A próxima seção apresenta os fundamentos sobre temas abordados. A discussão dos trabalhos relacionados da literatura recente é apresentada na Seção 3. A Seção 4 apresenta detalhes da modelagem em `Viewport-P`. Na Seção 5 são apresentados aspectos de implementação da modelagem, bem como discutidos os resultados experimentais obtidos. Por fim, a Seção 6 apresenta as principais conclusões e direcionamentos para trabalhos futuros.

2. Fundamentos

A seguir, fundamentação teórica e principais conceitos são apresentados.

2.1. Múltiplas Dimensões de Imersão

Na dimensão de **qualidade**, obtém-se o conjunto Q, no qual o vídeo é codificado em múltiplas taxas de bits (*bitrates*), quadros por segundo e resoluções. Na dimensão de **tempo**, cada representação do vídeo em uma qualidade $q \in Q$ é particionada em um conjunto de segmentos temporais S, com duração típica entre 1 e 10 segundos. Na dimensão de **espaço**, cada segmento $s \in S$ é fragmentado em blocos retangulares independentes, denominados *tiles*, formando um conjunto T cuja cardinalidade varia de dezenas a centenas de elementos. Como resultado desse processamento, o vídeo passa a ser representado

por uma estrutura multidimensional, análoga a um mosaico em 360°, definida pelo conjunto de objetos $O = Q \times S \times T$, cuja cardinalidade é dada por $|O| = |Q| \cdot |S| \cdot |T|$. Em outras palavras, cada objeto web requisitado via HTTP pelo *player* HMD corresponde a um objeto $o \in O$, definido pela tupla $o = (q, s, t)$, que representa um arquivo específico de *tile* t pertencente a um segmento temporal s, codificado em uma qualidade q.

2.2. Visualização Imersiva

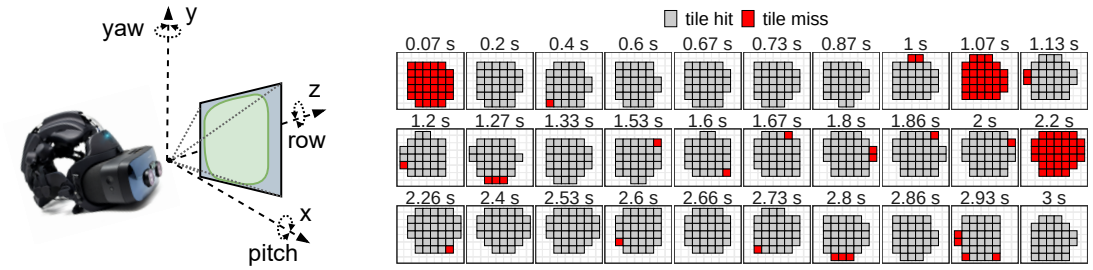
A orientação em 3D é descrita pelas rotações angulares em torno dos eixos, conforme ilustra a Figura 1(a): horizontal ψ (*yaw*), ao mover o olhar para esquerda/direita; vertical θ (*pitch*), com o mover do olhar para cima/baixo; e longitudinal ϕ (*roll*), com a inclinação lateral da cabeça. Assim, conteúdos imersivos distribuídos na Internet atualmente são predominantemente ofertados em três graus de liberdade (3DoF), i.e., considerando apenas orientação 3D através das rotações angulares (ψ, θ, ϕ). Dessa forma, movimentos em 3DoF possibilitam a visualização integral da esfera de 360°, contudo, a perspectiva espacial é fixa, definida pela posição das câmera omnidirecionais no momento da captura da mídia.

2.3. Navegação HMD em 360°

Um movimento em 3DoF resulta na visualização da região delimitada pelo FoV, tipicamente entre 100–110°de toda a esfera. A Figura 1 ilustra a interação de um usuário HMD, na qual o FoV descrito em informação angular é mapeado em informação discreta através da respectiva Janela de Visualização (*viewport*), que define respectivamente o subconjunto de *tiles*, T. Logo, a navegação HMD em 3DoF pode ser representada por um conjunto V, contendo a sequência de *viewports* resultantes de cada movimento rotacional angular sobre eixos (ψ, θ, ϕ). Para cada *viewport* V_j associada ao segmento temporal s_i, os *tiles* correspondentes $T_{j,i}$ devem estar disponíveis no *buffer* do *player* HMD, recuperados em níveis de qualidade $Q_{j,i}$ determinados por um algoritmo de adaptação de taxa de bits, ABR (*Adaptive Bitrate*). Portanto, a transição para um novo *viewport* V_{j+1}, durante o mesmo segmento s_i requer o download e o pré-carregamento antecipado dos *tiles* $T_{j+1,i}$ associados à projeção do novo FoV, a fim de evitar interrupções na reprodução, o que torna essenciais mecanismos eficientes de predição de *viewports*.

2.4. Streaming de Vídeo 360° Adaptado à Navegação HMD

Para uma navegação 3DoF livre durante o streaming, uma abordagem intuitiva, porém ingênua, consistiria em recuperar todos os *tiles* da esfera 360°, $\forall s \in S_t$, em cada segmento temporal $\forall t \in T$. Contudo, o campo FoV cobre tipicamente cerca de 15% da esfera, enquanto os usuários raramente exploram todo o espaço de 360°. Como resultado, aproximadamente $2/3$ dos *tiles* transmitidos podem nunca ser visualizados pelo usuário, então baixar todos os tiles ocasiona não somente sobrecarga nos sistemas finais, mas tráfego redundante na rede, sobrecarregando desnecessariamente recursos de rede, i.e., capacidade dos enlaces de gargalo compartilhados. Nesse contexto, uma abordagem mais eficiente consiste em requisitar apenas os *tiles* indicados por um mecanismo de predição e ausentes no *buffer*. Para cada segmento t_i, os *tiles* do *viewport* inicial do segmento, denotados por $S_{0,i}$, devem ser baixados e pré-carregados. Movimentos subsequentes do usuário com o HMD podem demandar *tiles* adicionais, caso não estejam previamente armazenados. A Figura 1(b) ilustra esse mecanismo com base em traços reais de navegação



(a) HMD, FoV, *viewport*. (b) Transições entre *viewports* de 1s a cada movimento do HMD.

Figura 1. Interação do usuário através do dispositivo HMD.

em HMD [Rosa et al. 2026] , evidenciando acertos (*hits*) e faltas (*misses*) de *tiles* no *buffer* ao longo de três segmentos de vídeo de 1 segundos. Na figura, observa-se que o *viewport* inicial (*tiles* em vermelho) de cada segmento de 1 segundo encontra-se ausente no *buffer* do player HMD e deve ser requisitado integralmente.

2.5. Aprendizado Profundo

Arquiteturas de Aprendizado Profundo, tipicamente implementadas por meio de DNNs (*Deep Neural Networks*), consistem em múltiplas camadas de neurônios (e.g., entrada, camadas ocultas e saída), nas quais cada camada transforma e propaga informações para as subsequentes, permitindo a extração hierárquica de características dos dados [Perumal et al. 2024]. Dentre essas arquiteturas, as Redes Neurais Convolucionais (CNNs) são empregadas em diversos domínios, destacando-se pela eficiência na extração de características e padrões espaço-temporais, especialmente sob restrições nos dados de entrada. Além disso, as CNNs possibilitam integração com outras arquiteturas de aprendizado, originando modelos híbridos com maior capacidade de representação. Nesse contexto, considerando a predição eficiente de *viewports*, problema central deste trabalho, arquiteturas baseadas em CNNs mostram-se particularmente adequadas para a implementação de preditores, sobretudo quando os conjuntos de dados são limitados a poucos atributos, e.g., apenas à representação do movimento do HMD do usuário (Figura 1(a)), expressa pelas rotações angulares em três eixos.

3. Trabalhos Relacionados

Em [Setayesh and Wong 2023], os autores propõem uma metodologia que combina modelo de detecção de saliência (regiões do vídeo mais vistas) baseado em CNN com modelo de predição de movimento baseado em GRU. Os dois componentes são integrados utilizando técnicas de fusão (AND, média e máximo), gerando um mapa de características que orienta a predição. A proposta utiliza um padrão de *tiling* 6×8 e superou algorítimos do estado da arte em 7.46%.

Em [Wan et al. 2024], os autores utilizam LSTM bidirecional combinada com módulo de detecção de objetos baseado em `YOLOv3`, incorporando a influência do conteúdo visualizado na movimentação do usuário, além da predição do *viewport*. Também é apresentada uma estratégia de seleção de qualidade de vídeo e priorização de *download* baseada na distância do *tile* em relação ao FoV atual. A combinação das estratégias supera outros modelos tradicionais, como ARIMA, ao prever *tiles* com 1 e 2 segundos de antecedência.

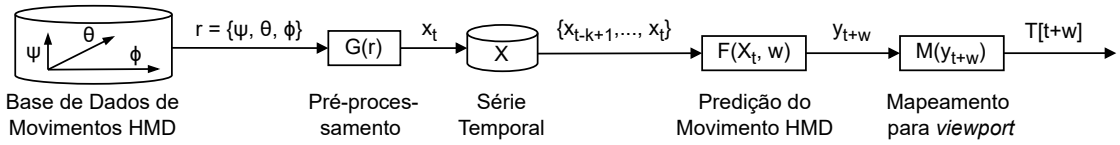


Figura 2. *Pipeline* de principais operações de um preditor de *viewport*.

Em [Mahmoud et al. 2024], são utilizados modelos baseados em DNN (CNN e LSTM) tanto para regressão, onde as séries de *pitch* e *yaw* são compostas por valores trigonométricos (seno e cosseno), quanto para classificação de *tiles*. Os resultados mostram que os modelos de regressão apresentam desempenhos semelhantes, especialmente para tempos de predição mais elevados. Na classificação, os modelos obtiveram melhor desempenho, ainda com dificuldade em horizontes de predição maiores.

Observa-se que, nesses trabalhos recentes da literatura, tanto CNN quanto LSTM aparecem como potenciais arquiteturas de modelos preditores, inclusive de forma híbrida. No entanto, esses trabalhos não apresentam uma modelagem formal e sistemática fim-a-fim das operações e componentes sistêmicas do *pipeline* de predição de *viewport*, em conjunto com a avaliação de desempenho de preditores CNN puros e híbridos via métricas de multi-domínio, cobrindo tanto o desempenho do modelo quanto o impacto da predição nas métricas de sistemas de *streaming*.

4. `Viewport-P`: Modelagem de Preditor de *Viewports* para Streaming 360

Esta seção descreve `Viewport-P`, uma modelagem dos principais componentes e de suas operações em um mecanismo de predição de *viewport* para sistemas de *streaming* de vídeo 360°, conforme o fluxo de dados do *pipeline* ilustrado na Figura 2.

4.1. Base de Dados de Movimento HMD

A modelagem proposta considera cenários atuais de *streaming* imersivo, definidos em três graus de liberdade, 3DoF (seção 2.3), i.e., $r = \{\psi, \theta, \phi\}$, rotações angulares do movimento HMD em torno dos eixos *yaw*, *pitch* e *roll*. A base de dados utilizada [Dharmasiri et al. 2021] é composta por traços reais de movimento 3DoF de múltiplos usuários, obtidos pela agregação de cinco conjuntos previamente disponibilizados na literatura. Essa base de múltiplos conjuntos foi escolhida por contornar a principal limitação de outros conjuntos disponíveis na literatura, i.e., falta de variedade de vídeos, usuários e baixa quantidade de entradas. Maiores detalhes são apresentados e discutidos na Seção 5.4.

4.2. Pré-processamento dos Dados

Em consonância com trabalhos correlatos [Setayesh and Wong 2023, Wan et al. 2024, Mahmoud et al. 2024], os traços de navegação HMD foram pré-processados para reduzir erros de predição por meio da função $\mathbf{G}(r)$ (Figura 2). Inicialmente, realiza-se seleção de atributos, restringindo o movimento a 2DoF, considerando apenas as componentes angulares (ψ, θ) (*yaw* e *pitch*), enquanto ϕ (*roll*) é descartada por não afetar o FoV. Para tratar descontinuidades periódicas, (ψ, θ) são convertidos para radianos e codificados por seno e cosseno. As velocidades angulares $(\dot{\psi}, \dot{\theta})$ são estimadas por diferenças finitas em séries temporais com $\Delta t = 100$ ms e, em seguida, normalizadas via *z-score*.

4.3. Conjunto de Dados em Série Temporal

A base de dados é então formalizada em um conjunto de dados contendo a série temporal $\mathbf{X} = \{\mathbf{x}_t\}_{t=1}^{N}$, em que cada amostra $\mathbf{x}_t \in \mathbb{R}^6$ é definida por

$$\mathbf{x}_t = \big[\sin(\psi_t), \cos(\psi_t), \sin(\theta_t), \cos(\theta_t), \dot{\psi}_t', \dot{\theta}_t'\big], \tag{1}$$

onde N denota o número total de amostras temporais, $\dot{\psi}_t \triangleq (\psi_t - \psi_{t-1})/\Delta t$ e $\dot{\theta}_t \triangleq (\theta_t - \theta_{t-1})/\Delta t$ representam as velocidades angulares nos eixos vertical e transversal, respectivamente.

4.4. Predição de Movimento HMD

O preditor de movimento do HMD é modelado como $\mathbf{F}(\mathbf{X}_t^k, w)$, que, a partir de um instante t, estima o estado futuro $\mathbf{y}_{t+w}$ em um horizonte de predição $w > 0$, condicionado a um histórico deslizante de k observações recentes $\{\mathbf{x}_{t-k+1}, \ldots, \mathbf{x}_t\}$. De acordo com a codificação angular adotada para a entrada do preditor, Equação (1), o estado futuro $\mathbf{y}_{t+w} \in \mathbb{R}^4$ é representado pelas componentes trigonométricas das orientações angulares futuras, em radianos, dadas por

$$\mathbf{y}_{t+w} = \big[\sin(\psi_{t+w}), \cos(\psi_{t+w}), \sin(\theta_{t+w}), \cos(\theta_{t+w})\big]. \tag{2}$$

4.5. Mapeamento do Movimento HMD predito em *Viewport*

O movimento HMD predito em $\mathbf{y}_{t+w}$, Equação (2), possui representação contínua, incompatível com sistemas reais de *streaming* imersivo baseados em requisições de *tiles* [Rosa et al. 2025]. Assim, propõe-se uma função de mapeamento $\mathbf{M}(\mathbf{y}_{t+w})$ que converte a predição angular em um subconjunto discreto $T[t+w]$ de *tiles* pertencentes ao *viewport*. Adota-se projeção equiretangular, considerando razão de aspecto 16:9, na qual a esfera é mapeada para um plano bidimensional. As coordenadas angulares (*yaw*, *pitch*) são transformadas em coordenadas normalizadas $(x, y) \in [0, 1]^2$ por:

$$x_{t+w} = \frac{\psi_{t+w} + \pi}{2\pi}, \qquad y_{t+w} = \frac{\theta_{t+w} + \frac{\pi}{2}}{\pi}. \tag{3}$$

A partir de (x_{t+w}, y_{t+w}), o plano é particionado em uma grade de *tiles*, definindo a janela de visualização. Embora a projeção introduza distorções geométricas, a cobertura do FoV é preservada, dispensando o mapeamento ponto a ponto da superfície esférica.

4.6. Restrições Temporais e Horizontes de Predição

Entretanto, em arquiteturas de streaming de vídeos 360 sob-demanda baseados em segmentos (§2.4), a definição do horizonte futuro w impõe restrições diretas sobre a disponibilidade de dados históricos. Para viabilizar a inferência antes do término de um segmento de duração s, o horizonte w limita o histórico máximo acumulável a

$$k_{\max} = \lfloor (s - w) \cdot f \rfloor, \tag{4}$$

sendo f a frequência de amostragem do HMD, tipicamente variando em 10-120 Hz [Ashida and Fujimoto 2022]. Estabelece-se, portanto, uma contra-partida crítica: horizontes w estendidos, necessários para compensar degradações de vazão e latência de rede, reduzem progressivamente a janela de histórico k disponível para o preditor.

4.7. Antecipação de Requisições de *Tiles*

Considerando as predições condicionadas à duração do segmento, identificam-se dois regimes operacionais para escalonamento de requisições de *tiles*. No primeiro, **intra-segmento** ($t < s - w$): onde a predição em $t + w$ permite antecipar requisições de *tiles* pertences ao segmento corrente. No segundo **inter-segmento** ($t \geq s - w$): onde o horizonte se projeta sobre o segmento subsequente ($t + w > s$), possibilitando a antecipação dos *tiles* do próximo segmento, o qual concentra o maior volume de dados a serem recuperados em t, devido ao *viewport* inicial de $s + 1$ (Figura 1). Em ambos os regimes, a antecipação de requisições eleva a taxa de acertos (*tile hit*) no *buffer* no player e, consequentemente, mitigando o impacto de interrupções (*stalls*) durante a reprodução sob navegação HMD no momento t.

4.8. Segmentação Espaço-Temporal do *Buffer*

Considerando os regimes temporais para escalonamento de requisições, assumindo um cenário ideal de *streaming* (i.e., sem restrições de vazão ou latência de rede), o estado do *buffer* do player HMD no instante t do segmento corrente s é definido por

$$\mathbf{B}_t(s) = \mathbf{P}_{\text{inter}}(s-1) \ \cup \ \mathbf{P}_{\text{intra}}(s) \ \cup \ \mathbf{T}_{\text{miss}}(s) \ \cup \ \mathbf{P}_{\text{inter}}(s+1), \tag{5}$$

onde $\mathbf{P}_{\text{inter}}(s-1)$ e $\mathbf{P}_{\text{inter}}(s+1)$ denotam, respectivamente, os conjuntos de *tiles* inter-segmento antecipados no segmento anterior e preditos para o segmento subsequente; $\mathbf{P}_{\text{intra}}(s)$ corresponde aos *tiles* intra-segmento preditos no segmento corrente; e $\mathbf{T}_{\text{miss}}(s)$ representa os *tiles* intra-segmento ausentes no *buffer* no instante de reprodução e recuperados sob demanda. Dado o conjunto de *tiles* requeridos pelo *viewport* no instante t no segmento s, $\mathbf{V}_t(s)$, os conjuntos de *tiles* disponíveis (*hit*) e ausentes (*miss*) em *buffer* são

$$\mathbf{T}_{\text{hit}}(s) = \mathbf{V}_t(s) \cap \mathbf{B}_t(s), \qquad \mathbf{T}_{\text{miss}}(s) = \mathbf{V}_t(s) \setminus \mathbf{B}_t(s), \tag{6}$$

cujas cardinalidades quantificam, respectivamente, em s os eventos de *tile hit* e *tile miss*.

5. Avaliação da Modelagem `Viewport-P`

Conforme modelagem proposta para um mecanismo de predição de *viewports*, detalhes da implementação dos componentes são discutidos a seguir. Em especial, maior foco é dado o componente central, o preditor $\mathbf{F}(\mathbf{X}|_t^k, w)$.

5.1. Modelos de Predição Implementados

Os modelos de predição considerados neste trabalho são baseados em três arquiteturas de Redes Neurais Convolucionais (CNN): a saber: CNN puras, híbridas GRU (*Gated Recurrent Unit*) e LSTM (*Long Short-Term Memory*). Esses modelos foram escolhidos por representarem diferentes níveis de capacidade de modelagem temporal e, enquanto são amplamente estabelecidos na literatura, são frequentemente utilizados em trabalhos recentes. Particularmente, a CNN captura padrões locais de curto prazo, enquanto as arquiteturas híbridas incorporam dependências temporais mais longas via camadas recorrentes. Enquanto o CNN–GRU busca melhor compromisso entre capacidade temporal e custo computacional, o CNN–LSTM oferece maior expressividade para dependências de longo prazo, ao custo de maior complexidade. Em todos os modelos, a entrada consiste

no histórico $\{\mathbf{x}_{t-k+1}, \ldots, \mathbf{x}_t\}$ processado por convoluções 1D com 64 filtros, kernel 3 e ativação ReLU, seguidas de *max pooling* (fator 2) e *dropout* (0.15). Na CNN, a saída é achatada e conectada diretamente a uma camada densa que produz $\mathbf{y}_{t+w} \in \mathbb{R}^4$. Nos modelos híbridos, as representações convolucionais alimentam camadas recorrentes empilhadas (128 e 64 unidades), com *batch normalization* ($\epsilon = 0.001$): GRU no CNN–GRU e LSTM no CNN–LSTM. Como *baseline*, pode-se adotar o modelo CNN, puramente convolucional, conforme práticas consolidadas na literatura [Mahmoud et al. 2024], recebendo como entrada apenas informações angulares elementares do movimento HMD, codificadas como $\mathbf{x}_t = [\sin(\psi_t), \cos(\psi_t), \sin(\theta_t), \cos(\theta_t)]$. Os modelos foram implementados em `Python 3.8.8` com a biblioteca `keras` 2.13.1.

5.2. Ambiente Experimental

A implementação da modelagem `Viewport-P` e os experimentos foram conduzidos em uma máquina virtual `Ubuntu-18.04.6 LTS`, executada via `VirtualBox` e gerenciada com `Vagrant`, configurada com 8 CPUs e 8 GB de RAM, sobre hardware Intel Xeon E3-1220 v6 3 GHz. Nesse ambiente, os modelos CNN foram treinados e avaliados, demonstrando viabilidade computacional em dispositivos de usuário, como HMDs *all-in-one* com recursos equivalentes, a exemplo do `HTC Vive`[6] e `Meta Quest`[7].

5.3. Metodologia e Configurações do Ambiente

Para analisar a robustez preditiva dos modelos, este trabalho avalia um conjunto de horizontes frequentemente utilizados tanto na literatura quanto em aplicações reais, de curto a longo prazo, em $W = \{0.5, 1.0, 1.5, 2.0, 2.5\}$ segundos. A amostragem de movimentos HMD considerada é de $f = 10$ Hz, conforme as amostras de intervalos uniformes em $\Delta t = 100$ ms (frequência de amostragem) na série temporal. Foram considerados vídeos em 360 com segmento de duração $s = 4$ segundos, típico em *streaming* na Internet [Kimura et al. 2025]. Considerando as restrições temporais (§4.6), conforme f, W e s definidos, o histórico k de observações passadas utilizado tanto para treinamento quanto para inferência dos modelos foi limitado a $k_{\max}$, Equação (4), sendo, portanto, $K = \{35, 30, 25, 20, 15\}$ amostras mais recentes para $w \in W$, respectivamente. O conjunto de dados (§4.1, §4.2) foi particionado em 70% para treino, 15% para validação e 15% para teste. O treinamento ocorreu em *batches* $= 512$ de tamanho, com 10 épocas de treinamento, função de perda (*loss*) com MSE (*Mean Square Error*), *EarlyStopping* em 10, decaimento de taxa de aprendizado em 10, *momentum* em 0.99. Foi utilizado otimizador ADAM parametrizado com $\alpha = 0.0001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-7}$.

5.4. Caracterização dos Conjuntos de Dados

Foram considerados cinco conjuntos de dados da literatura: D1 [Corbillon et al. 2017], D2 [Lo et al. 2017], D3 [Wu et al. 2017], D4 [Guan et al. 2019] e D5 [Nasrabadi et al. 2019]. Esses conjuntos foram agregados em único (Todos), utilizado no treinamento dos modelos. Cada usuário gerou 600 amostras de movimento HMD, correspondentes a um minuto de vídeo com taxa de amostragem $\Delta_t = 100$ ms. Para caracterização, foi aplicada a entropia sobre os movimentos HMD. Especificamente,

[6] `https://developer.vive.com/resources/hardware-guides/vive-specs-user-guide/`

[7] `https://www.meta.com/br/quest/quest-3/`

Tabela 1. Conjuntos de Dados.

Conjunto	n_s	n_u	n_v	Distribuição (%) dos quartis da entropia			
D3	72	38	9	38.9	26.4	26.4	8.33
D1	61	41	7	36.1	19.7	27.9	16.4
D4	144	45	18	24.3	28.5	22.9	24.3
D2	80	43	10	17.5	21.2	31.2	30.0
D5	140	29	28	17.9	25.0	21.4	35.7
Todos	497	250	144	24.9	24.9	24.9	25.2

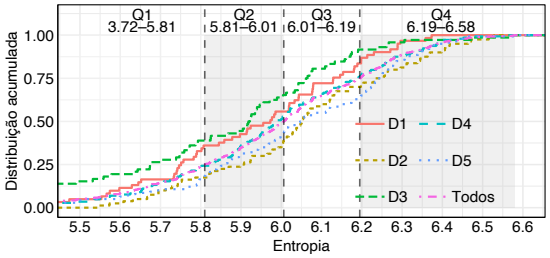


Figura 3. FDA da entropia.

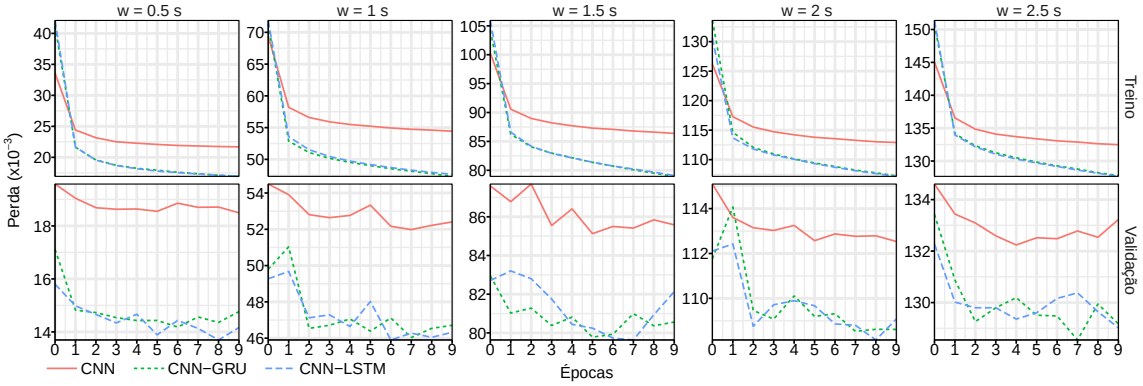


Figura 4. Perda no treinamento e validação dos modelos em cada horizonte w.

o espaço angular foi discretizado em regiões, e a entropia conjunta de Shannon foi estimada como $H(\Psi, \Phi) = -\sum_{\psi \in \Psi} \sum_{\phi \in \Phi} P(\psi, \phi) \log_2 P(\psi, \phi)$, onde Ψ e Φ representam os conjuntos de valores de movimento angular horizontal e vertical, respectivamente, e $P(\psi, \phi)$ a probabilidade conjunta de ocorrência do movimento em ambos os eixos. Essa métrica captura o grau de imprevisibilidade dos movimentos, onde valores mais baixos indicam comportamento mais regular e previsível, enquanto valores mais altos refletem maior variabilidade e diversidade nos padrões de movimento HMD. A Tabela 1 apresenta a caracterização dos conjuntos, com o número de sessões (n_s), usuários (n_u) e vídeos (n_v), bem como a distribuição percentual dos quartis da entropia observada em cada sessão n_s. A Figura 3 apresenta distribuição acumulada (FDA) da entropia, evidenciando diferenças estatísticas entre os conjuntos quanto à heterogeneidade e complexidade do movimento HMD, destacadas pela segmentação em quartis. Observa-se que o conjunto D3 concentra entropia em níveis mais baixos (no quartil Q1), indicando menor variabilidade dos movimentos, enquanto D5 apresenta maior concentração em níveis elevados (Q4), refletindo maior movimentação. Os conjuntos D1, D4 e D2 exibem comportamento intermediário, com transição gradual ao longo dos quartis. D4 e D5 apresentam maior representatividade devido ao maior número de sessões (n_s). O conjunto agregado (Todos) suaviza essas variações, resultando em distribuição balanceada entre os quartis.

5.5. Desempenho no Treinamento dos Modelos

A Figura 4 apresenta a perda (*loss*) observada durante o treinamento e validação dos modelos para cada horizonte de predição em W, sobre o conjunto agregado (Todos). O desempenho dos modelos indica comportamento consistente de convergência, com redução progressiva da perda em todos horizontes em W, conforme avanço das épocas. Observa-se que os modelos CNN–LSTM e CNN–GRU oferecem convergência semelhante, com

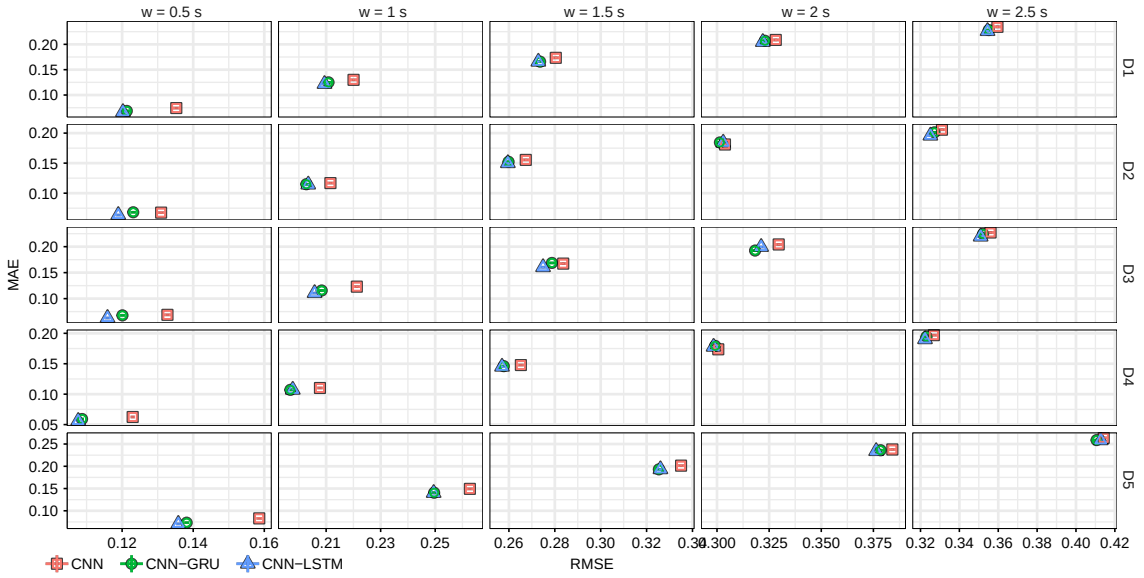
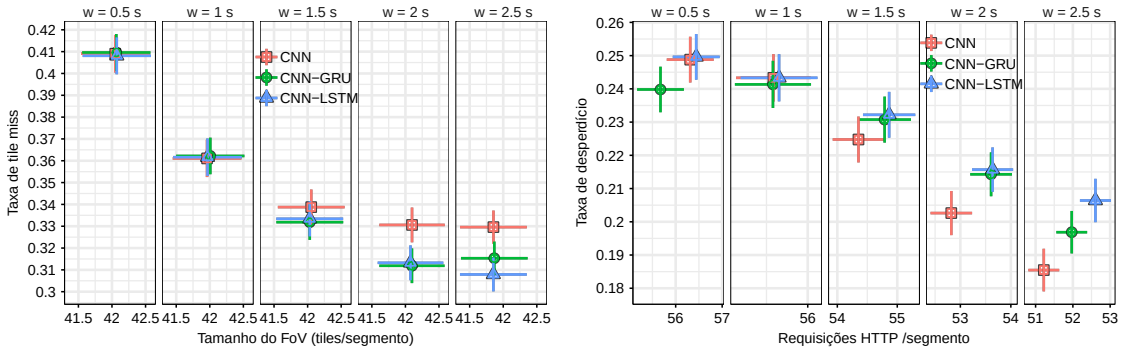


Figura 5. Erro dos modelos para diferentes conjuntos e horizontes w.

GRU ligeiramente inferior em termos de perda final, ambos tendendo a alcançar os menores valores de perda de forma mais estável, especialmente para horizontes menores, sugerindo melhor capacidade de captura de dependências temporais. Já o modelo CNN puro converge mais rapidamente nas primeiras épocas, mas tende a estabilizar em níveis de erro mais elevados, indicando menor capacidade de modelar padrões temporais complexos para movimentos HMD. Notadamente, maiores horizontes de predição em W impactam significativamente na perda, evidenciando, como esperado, que a incorporação de mais contexto temporal futuro piora o desempenho preditivo.

5.6. Erro de Predição Angular do Movimento HMD

A predição do movimento angular HMD é avaliada por MAE (*Mean Absolute Error*) e RMSE (*Root Mean Square Error*), métricas de erro de regressão amplamente utilizadas na literatura. A Figura 5 apresenta os resultados MAE e RMSE médios, evidenciando um padrão consistente, em que o erro aumenta com o horizonte de predição (w), independentemente do modelo e do conjunto de dados. No cenário mais curto ($w = 0.5\,\text{s}$), o CNN-LSTM apresenta o melhor desempenho, com reduções de MAE de aproximadamente 6–10% em relação ao CNN e 3–6% em relação ao CNN-GRU (e.g., em D4, com 0.0559 vs. 0.0625 e 0.0592), além de reduções similares em RMSE. À medida que w cresce (1–2 s), CNN-LSTM e CNN-GRU convergem, com diferenças marginais (tipicamente $<$3%). No pior cenário ($w = 2.5\,\text{s}$), todos os modelos degradam significativamente, com MAE até 0.26 (D5) e RMSE acima de 0.41, sendo o CNN consistentemente o pior, porém com diferenças reduzidas, quando CNN-GRU e CNN-LSTM apresentam ganhos de apenas 1–3% em MAE e $<$ 1% em RMSE, o que, na prática, representa melhoria marginal. Entre os conjuntos, D5 é o mais desafiador, enquanto D4 apresenta os menores erros, com diferenças de até 20–25%. A razão RMSE/MAE acima de 1.5 em todos os cenários indica a presença de erros extremos. Esse efeito é mais pronunciado em conjuntos com maior entropia (e.g., D5, com maior concentração em Q4), que apresentam maior dispersão de erro, enquanto conjuntos de menor entropia (e.g., D3) exibem comportamento mais estável. Assim, a variabilidade do erro está mais associada à imprevisibilidade dos dados,



(a) Taxa de *tile miss*. (b) Taxa de desperdício de requisições.

Figura 6. Métricas de eficiência espacial observadas sobre o conjunto D5.

que cresce com w, do que à arquitetura do modelo. Em síntese, CNN-LSTM é superior em horizontes curtos, CNN-GRU empata em cenários intermediários, e, para horizontes longos, os modelos tornam-se praticamente equivalentes, de modo que o custo adicional das arquiteturas recorrentes pode não se justificar.

5.7. Impacto da Predição na Transmissão

As métricas de erro em MAE e RMSE quantificam a discrepância angular entre os estados preditos e observados, porém não capturam diretamente o impacto da predição sobre sistemas de *streaming* baseados em *tiles*. Para esse fim, adota-se a métrica de eficiência espacial em *buffer*, conforme a modelagem do `Viewport-P` (§4.8). Dado o conjunto de *tiles* requeridos no segmento s, $\mathbf{V}(s)$, a taxa de erro de predição espacial (*tile miss*) é definida como $\eta_{\text{miss}}(s) = |\mathbf{T}_{\text{miss}}(s)| \;/\; |\mathbf{V}(s)|$,, onde $\mathbf{T}_{\text{miss}}(s)$ denota os *tiles* requeridos durante s e ausentes no *buffer*. Ainda, quantifica-se a taxa de desperdício de requisições HTTP, i.e, Requisições desperdiçadas correspondem àquelas em que erros de predição levam ao download de *tiles* que não são visualizados, implicando uso ineficiente de recursos de rede.

A Figura 6 apresenta as taxas de *tile miss* e de requisições desperdiçadas para o conjunto D5, escolhido por sua maior variabilidade de movimento HMD e, consequentemente, maior erro de predição angular. Os três modelos exibem comportamento semelhante. Contudo, diferentemente de MAE e RMSE, que aumentam com o horizonte de predição (w), ambas as métricas espaciais melhoram com w. A taxa de *tile miss* reduz-se de aproximadamente 41% para valores entre 31% e 33%, com o CNN-LSTM apresentando o melhor desempenho, seguido de perto pelo CNN-GRU, enquanto o CNN obtém os maiores valores. No maior horizonte ($w = 2.5$ s), apesar do pior erro angular, o CNN-LSTM reduz o *tile miss* em 6.6% em relação ao CNN (30.8% vs. 33.0%) e em 2.4% em relação ao CNN-GRU (31.5%). Esse ganho, entretanto, vem acompanhado de maior desperdício de requisições: o CNN-LSTM atinge até 20.6%, contra 18.5% do CNN e 19.7% do CNN-GRU, evidenciando um *trade-off* entre cobertura e eficiência. O tamanho médio do FoV (em número de *tiles*) permanece praticamente constante entre os modelos, indicando que as diferenças observadas decorrem principalmente da antecipação das requisições (horizonte w), e não da qualidade da predição. Além disso, o número total de requisições HTTP por segmento decresce com w para todos os modelos, com diferenças pouco significativas entre eles. Notavelmente, o CNN apresenta comportamento mais

conservador, resultando em menor desperdício à medida que w aumenta. Em síntese, o CNN-LSTM maximiza a cobertura, enquanto o CNN-GRU oferece melhor equilíbrio entre cobertura e eficiência. Já o CNN, embora inferior em *tile miss*, é mais eficiente em termos de desperdício, evidenciando o compromisso entre precisão e custo.

5.8. Discussão

Em contraste com a literatura, métricas de eficiência espacial em *buffer*, como *tile miss* e desperdício de requisições, mostram-se mais sensíveis e representativas do desempenho prático do que métricas de regressão. Em especial, diferenças em RMSE e MAE não se traduzem diretamente em acurácia espacial. Para horizontes curtos ($w \leq 1\,\mathrm{s}$), mesmo com maior erro angular, a CNN apresenta *tile miss* semelhante aos modelos híbridos, inclusive em cenários de alta variabilidade (e.g., D5). Para horizontes mais longos ($w \geq 1.5\,\mathrm{s}$), mais relevantes por viabilizarem requisições antecipadas, as métricas espaciais passam a diferenciar claramente os modelos. Em síntese, os resultados indicam que w é o principal fator determinante do desempenho. Tanto *tile miss* quanto desperdício tendem a reduzir com o aumento de w, devido à maior janela temporal para antecipação e recuperação de *tiles*. Além disso, erros em $t + w$ podem ser parcialmente compensados até o *playback*, pois *tiles* inicialmente incorretos podem vir a ser efetivamente utilizados.

6. Conclusões

Este trabalho propôs `Viewport-P`, uma modelagem formal do *pipeline* de predição de *viewport* em sistemas de *streaming* de vídeo 360. Tal modelagem foi implementada e avaliada em ambiente controlado, considerando arquiteturas estabelecidas (CNN, CNN-GRU e CNN-LSTM). A análise experimental combinou métricas de regressão (MAE e RMSE) com métricas sistêmicas de eficiência de *buffer*, evidenciando limitações das primeiras em capturar o impacto prático da predição. Os resultados mostraram que a incorporação de memória recorrente através dos modelos híbrido possibilita maior desempenho, principalmente para horizontes de predição mais longos, relevantes para QoE, onde as métricas espaciais, como *tile miss*, são mais representativas do desempenho sistêmico. Como trabalho futuro, a natureza modular do `Viewport-P` permite sua extensão para cenários mais complexos, como ambientes com alta mobilidade (e.g., redes veiculares).

Agradecimentos

Este trabalho foi parcialmente financiado pelas seguintes Agências: Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), processos #2024/21006-1, #2022/14503-3, #2015/18808-0; e *Swedish Knowledge Foundation*, *DRIVE project*.

Referências

Alidadi Shamsabadi, A., Yadav, A., Gadallah, Y., and Yanikomeroglu, H. (2025). Exploring the 6g potentials: Immersive, hyperreliable, and low-latency communication. *IEEE Vehicular Technology Magazine*, pages 2–10.

Ashida, H. and Fujimoto, K. (2022). Comparing measurements of head motion and centre of pressure for body sway induced by optic flow on a head-mounted display. *Frontiers in Virtual Reality*, Volume 3 - 2022.

Bentaleb, A., Lim, M., Hammoudi, S., Harous, S., and Zimmermann, R. (2025). Solutions, challenges, and opportunities in volumetric video streaming: An architectural perspective. *ACM Trans. Multimedia Comput. Commun. Appl.*, 21(7).

Corbillon, X., De Simone, F., and Simon, G. (2017). 360-degree video head movement dataset. In *Proceedings of the 8th ACM on Multimedia Systems Conference*, MMSys'17, page 199–204, New York, NY, USA. Association for Computing Machinery.

Dharmasiri, A., Kattadige, C., Zhang, V., and Thilakarathna, K. (2021). Viewport-aware dynamic 360° video segment categorization. In *Proceedings of the 31st ACM Workshop on Network and Operating Systems Support for Digital Audio and Video*, NOSSDAV '21, page 114–121, New York, NY, USA. Association for Computing Machinery.

Guan, Y., Zheng, C., Zhang, X., Guo, Z., and Jiang, J. (2019). Pano: optimizing 360° video streaming with a better understanding of quality perception. In *Proceedings of the ACM Special Interest Group on Data Communication*, SIGCOMM '19, page 394–407, New York, NY, USA. Association for Computing Machinery.

Kimura, B., Ferlin, S., Paiva, T., Mahmoodi, T., Brunstrom, A., and Alay, O. (2025). Evaluating adaptive video streaming over multipath quic with shared bottleneck detection. *ACM Trans. Multimedia Comput. Commun. Appl.*, 21(9).

Lo, W.-C., Fan, C.-L., Lee, J., Huang, C.-Y., Chen, K.-T., and Hsu, C.-H. (2017). 360° video viewing dataset in head-mounted virtual reality. In *Proceedings of the 8th ACM on Multimedia Systems Conference*, MMSys'17, page 211–216, New York, NY, USA. Association for Computing Machinery.

Mahmoud, M., Stamatia, S. R. R., Panayides, A. S., Lazaridis, P. I., Kantartzis, N. V., Karagiannidis, G. K., and Zaharis, Z. D. (2024). A comparative analysis of viewing prediction techniques for 360° video streaming applications. In *2024 Panhellenic Conference on Electronics Telecommunications (PACET)*, pages 1–4.

Nasrabadi, A. T., Samiei, A., Mahzari, A., McMahan, R. P., Prakash, R., Farias, M. C. Q., and Carvalho, M. M. (2019). A taxonomy and dataset for 360° videos. In *Proceedings of the 10th ACM Multimedia Systems Conference*, MMSys '19, page 273–278, New York, NY, USA. Association for Computing Machinery.

Perumal, T., Mustapha, N., Mohamed, R., and Shiri, F. M. (2024). A comprehensive overview and comparative analysis on deep learning models. *Journal on Artificial Intelligence*, 6(1):301–360.

Recommendation, I. (2023). Framework and overall objectives of the future development of imt for 2030 and beyond. *International Telecommunication Union (ITU) Recommendation (ITU-R)*.

Rosa, F., Ferlin, S., Brunstrom, A., da Costa, J. B. D., and Kimura, B. (2026). Enhancing 360° Video Streaming with Stream Scheduling Policies over HTTP/3. In *(to appear) Proceedings of the IEEE Wireless Communications and Networking Conference (IEEE WCNC 2026)*, Malaysia. IEEE.

Rosa, F., Ferlin, S., Brunström, A., and Kimura, B. (2025). End-to-End 360° Video Streaming over HTTP/3: Architecture and Implementation. In *Proceedings of the ACM/IRTF Applied Networking Research Workshop 2025 (ANRW'25)*, Spain. ACM.

Setayesh, M. and Wong, V. W. (2023). A content-based viewport prediction framework for 360° video using personalized federated learning and fusion techniques. In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pages 654–659.

Wan, Z., Hu, Y., Zhou, Y., Liu, X., and Zhao, S. (2024). Ebi360: An edge-assisted viewport prediction method for 360° video based on bilstm. In *2024 International Conference on Virtual Reality and Visualization (ICVRV)*, pages 19–24.

Wu, C., Tan, Z., Wang, Z., and Yang, S. (2017). A dataset for exploring user behaviors in vr spherical video streaming. In *Proceedings of the 8th ACM on Multimedia Systems Conference*, MMSys'17, page 193–198, New York, NY, USA. Association for Computing Machinery.