DETECÇÃO DE QUEDAS VIA FEATURES VIDEOMAEV2 E SHALLOW NETWORKS

FALL DETECTION VIA VIDEOMAEV2 FEATURES AND SHALLOW NETWORKS

Alisson Chaves Ferreira*
Silas Santiago Lopes Pereira**

RESUMO

Quedas acidentais são uma das principais causas de morte no mundo, principalmente entre a população idosa. Em resposta a este desafio, diversos trabalhos explorando algoritmos de *machine learning* têm sido desenvolvidos, acompanhados pela construção de uma ampla variedade de *datasets* baseados em vídeos de monitoramento. Contudo, a eficácia desses sistemas depende intrinsecamente da qualidade das representações dos dados para assegurar uma classificação precisa. Neste contexto, e visando obter representações robustas para a construção de um classificador de quedas, o presente estudo concentra-se na exploração do *Vision Transformer VideoMAEv2* como principal extrator de *features* e de uma *Shallow Network* para avaliar esses dados. O modelo foi empregado para extrair as principais características dos vídeos do *dataset* HQFS. Os resultados demonstraram um alto poder discriminatório das *features* extraídas, evidenciado por uma AUC de $0,921$. O modelo resultante do *Grid Search* demonstrou maior confiabilidade preditiva, alcançando uma precisão de $0,607$, mesmo com um *recall* bastante modesto na classe de queda. Esta estratégia comprova a eficácia da abordagem para extrair *features* que podem ser utilizadas na construção de sistemas de detecção mais robustos.

Palavras-chave: Detecção de Queda. *Vision Transformer*. *Machine Learning*. *VideoMAEv2*.

ABSTRACT

Accidental falls are one of the leading causes of death worldwide, particularly among the elderly population. In response to this challenge, several works exploring machine learning algorithms have been developed, along with the construction of a wide variety of datasets derived from surveillance videos. However, the effectiveness of these systems intrinsically depends on the quality of data representations to ensure accurate classification. In this context, and aiming to obtain robust representations for the construction of a fall classifier, the present study focuses on exploring the Vision Transformer VideoMAEv2 as the primary feature extractor and a Shallow

* Graduando em Bacharelado em Ciência da Computação, Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE), Aracati, Ceará, Brasil. E-mail: chavesalisson.a@gmail.com
** Silas Santiago Lopes Pereira é professor do IFCE, doutor, mestre e bacharel em ciências da computação pela universidade estadual do ceará (UECE). Endereço eletrônico: silas.santiago@ifce.edu.br.

Network to evaluate these data. The model was used to extract key features from the videos in the HQFS dataset. The results demonstrated a high discriminatory power of the extracted features, evidenced by an AUC of 92.10%. The model resulting from the Grid Search demonstrated greater predictive reliability, achieving a precision of 60.70%, even with a modest recall in the fall class. This strategy proves the efficacy of the approach for extracting features that can be used in the construction of more robust detection systems.

Keywords: Fall Detection. Vision Transformer. Machine Learning. VideoMAEv2.

1 INTRODUÇÃO

As quedas acidentais representam a segunda causa de morte por lesões não intencionais no mundo, especialmente entre idosos. De acordo com a Organização Mundial da Saúde (OMS), as quedas ocupam a segunda posição entre as principais causas de morte por ferimentos não intencionais em todo o mundo. É estimado que cerca de 684.000 pessoas morram por essa causa anualmente e, ainda, cerca de $37,3$ milhões de quedas são suficientemente graves para exigir atendimento médico às vítimas OMS (2021). Além disso, a população idosa está aumentando em um ritmo muito acelerado; até 2030, estima-se que uma em cada seis pessoas terá 60 anos ou mais e, até 2050, 22% da população global terá mais de 60 anos (USMANI et al., 2021). Aplicando-se as taxas atuais ao cenário demográfico previsto, estima-se que o número de óbitos anuais possa ultrapassar $1,3$ milhão, com mais de 70 milhões de idosos necessitando atendimento médico caso medidas preventivas eficazes não sejam implementadas.

Os problemas que ocorrem em decorrência de quedas não apenas geram altos custos econômicos, como também comprometem a independência na vida cotidiana, que afeta com frequência a população mais idosa (NÚÑEZ-MARCOS; ARGANDA-CARRERAS, 2024). Uma queda pode ser causada por diversos fatores externos, como condições de trabalho perigosas, perda de equilíbrio, efeitos de medicação, uso de bebidas alcoólicas ou outras substâncias e até mesmo condições financeiras precárias (HA et al., 2023).

Diante da necessidade de monitorar o cotidiano humano, a área de reconhecimento de ações humanas, conhecida pela sigla HAR (*Human Action Recognition* em inglês), tem ganho destaque nos últimos anos. Por exemplo, um indivíduo pode sofrer quedas de diferentes tipos; devido à grande variedade de cenários, são utilizadas muitas ferramentas capazes de monitorar e avaliar continuamente se uma queda ocorreu, sejam estas por meio de sensores *wearables*, como acelerômetros e giroscópios ou câmeras de vigilância utilizando dados de RGB, profundidade e infravermelho, além de modelos baseados em esqueleto 3D. Conforme os algoritmos evoluíram, algumas técnicas se tornaram mais populares para esse tipo de detecção. Alguns dos modelos mais explorados utilizam técnicas de aprendizado supervisionado para classificar se houve ou não uma queda, como é o caso dos algoritmos de *Multi-layer Perceptrons* (MLPs), *Support Vector Machines* (SVMs) e *Convolutional Neural Networks* (CNNs) (ESPINOSA et al., 2019; GAYA-MOREY; MANRESA-YEE; BUADES-RUBIO, 2024; ZURBUCHEN; WILDE; BRUEGGER,

2021).

Em razão desta situação preocupante e visando aprimorar a robustez dos sistemas de detecção, o presente trabalho adota uma metodologia focada na classificação binária de eventos (queda vs. não queda). A abordagem baseia-se em um processo de duas etapas: primeiramente o Vision Transformer *VideoMAEv2* (WANG et al., 2023) é empregado como um extrator de *features*, convertendo os clipes de vídeo em representações vetoriais. Subsequentemente, essas representações são utilizadas para treinamento e avaliação de desempenho de modelos de aprendizado de máquina, que atuam como classificadores. Para realização dos experimentos, foram utilizados dados estrategicamente escolhidos do *dataset HQFS* (*High-Quality Fall Simulation Data*). O objetivo central é avaliar a eficácia desta arquitetura de extração de *features* para a tarefa de classificação binária, visando contribuir para o desenvolvimento de sistemas de queda mais precisos e confiáveis.

A estrutura do presente trabalho foi construída da seguinte forma: A Seção 2 apresenta, de forma concisa, uma revisão dos principais conceitos e ferramentas utilizadas na pesquisa durante o seu desenvolvimento. Na Seção 3 estão presentes os trabalhos relacionados a este estudo que envolvem aplicações para reconhecimento de quedas. Já a Seção 4 apresenta a metodologia com os principais materiais e métodos que explicam a construção do modelo para extrair *features* e suas métricas de avaliação. Na Seção 5 são apresentados os resultados dos experimentos. Por fim, a Seção 6, com as considerações finais, expõe as motivações da pesquisa e propõe futuros aprimoramentos.

2 FUNDAMENTAÇÃO TEÓRICA

Nos últimos anos, tecnologias de aprendizado de máquina tornaram-se ubíquas em sistemas dos mais diversos gêneros, possibilitando o monitoramento de ações humanas (MAHADEVKAR et al., 2022). Dessa forma, estas permitiram maior autonomia no monitoramento de quedas sem a necessidade de um agente externo humano ou sistemas invasivos como sensores. Tal método viabilizou que algoritmos fossem treinados e ferramentas criadas com esse propósito. Com uma abordagem focada em detecção de quedas, as seguintes subseções apresentam o referencial teórico da temática abordada.

2.1 Aprendizado de Máquina Supervisionado

Nasteski (2017) apresenta que as tarefas de aprendizado de máquina, ou *Machine Learning* (ML), são técnicas de inteligência artificial (IA) que foram criadas para, principalmente, representar a capacidade de aprendizado humano, na qual a aplicação é capaz de reconhecer padrões e construir modelos matemáticos com base nesses padrões. Essa técnica, junto do grande número de dados disponíveis e do poder de processamento atual, permite realizar processos de ML complexos, o que possibilita a execução de trabalhos com dados multidimensionais de diversas áreas.

O aprendizado supervisionado é um dos métodos mais comuns do ML. Esse método é utilizado no treinamento de modelos a partir de conjuntos de dados nos quais cada entrada está ligada a um resultado conhecido (rótulo) com finalidade de classificação ou regressão. Seu objetivo é que o modelo treinado consiga utilizar uma função capaz de mapear novas entradas para suas saídas (BURKART; HUBER, 2021; KHAN; LAGHARI; AWAN, 2018). Dentre os algoritmos de aprendizado de máquina supervisionado, os mais presentes na literatura, de acordo com Nafea et al. (2024), Usmani et al. (2021), são os algoritmos SVM (*Support Vector Machines*), *Random Forest* e *Naive Bayes* (CHANDRA; BEDI, 2021; CHOW; REYES-ALDASORO, 2021; MASWADI et al., 2021).

2.2 *Transfer Learning*

Transfer Learning é uma técnica do aprendizado de máquina que se concentra em aplicar o conhecimento adquirido na solução de um problema distinto e utilizá-lo em um problema diferente, porém com alguma relação. Essa estratégia é muito útil quando, por exemplo, os dados do domínio alvo são escassos. Um dos principais motivos do uso do *transfer learning* é a dificuldade de generalização, ou seja, aplicar seu conjunto de *features*, extraídas de um *dataset* específico, em outro domínio com padrões estatísticos diferentes (MARAY et al., 2023). Ainda segundo Maray et al. (2023), *transfer learning* pode ser dividido em duas categorias: *transfer learning* homogêneo e heterogêneo.

Técnicas de *transfer learning* homogêneo buscam garantir uma boa generalização com domínios de origem e de destino que se assemelham no mesmo tipo de características entre os dados; entretanto, em muitos campos, isso é inviabilizado pela dificuldade de segmentar alguns tipos de dados por empecilhos como casos raros, privacidade ou alto tempo de processamento (CHUI et al., 2023).

O *transfer learning* heterogêneo foi criado com o intuito de solucionar esses problemas causados por domínios homogêneos, trabalhando com características não similares, que se aplicam em áreas que exigem maior complexidade. O seu foco principal é alinhar os dados de entrada com o domínio em que serão aplicados (MIGNONE; PIO; CECI, 2024).

2.3 Representação de Atributos em Visão Computacional

A visão computacional consiste na capacidade dos computadores de extraírem e obterem, de forma automatizada, detalhes importantes de imagens e vídeos de forma semelhante à capacidade de assimilação e compreensão visual humana (XU et al., 2021). Os objetivos da visão computacional em conjunto com o *machine learning* são dar aos computadores a capacidade de coletar os dados, compreendê-los e tomar decisões com base no que já foi treinado e no que será visto (MAHADEVKAR et al., 2022). O processo de visão computacional transforma os dados em formas matemáticas, manipulando valores de *pixel* para realizar a análise e extrair as principais características, ou seja, seus padrões (PANERU; JEELANI, 2021).

Entender como representar atributos dos dados em vídeos é o ponto principal no desafio de visualizar computacionalmente o que ali está representado. A extração e representação dessas características (*features*) são fundamentais para o reconhecimento e segmentação de ações (BEHRMANN et al., 2021). Um dos métodos padrões de extração de *highlights*[1] em vídeos brutos consiste na seleção de *frames* que mostram momentos relevantes para classificar o problema. Esses momentos são cortes de alguns segundos do vídeo original que representam aspectos relevantes do conjunto de dados para se trabalhar no problema (XIAO; YIN; KANG, 2021).

Muitos métodos para representar *features* podem ser implementados com objetivos variados. Para detecção de objetos, trabalha-se com *bounding boxes* utilizando, por exemplo, a arquitetura *You Only Look Once* (YOLO) (TERVEN; CÓRRKOV-ESPARZA; ROMERO-GONZÁLEZ, 2023). Como também pode-se realizar segmentação, que é uma detecção mais detalhada que envolve criar máscaras que contornam todos os *pixels* do objeto, o que deixa os seus contornos mais detalhados utilizando R-CNN Xu et al. (2022).

2.4 *VideoMAEv2*

O *VideoMAEv2 (Video Masked Autoencoders V2)* (WANG et al., 2023) classifica-se como um *foundation model* baseado em pré-treinamento auto-supervisionado, desenvolvido com o intuito de escalar modelos de vídeo robustos com bilhões de parâmetros. Para treinamento deste modelo, os autores construíram o *dataset UnlabeledHybrid*, o qual consolida cerca de 1.35 milhão de vídeos públicos. A composição deste conjunto agrega amostras das bases de dados *Kinetics*, *Something-Something*, *AVA*, *WebVid2M*, além de um subconjunto proprietário extraído do *Instagram*. Sendo este o principal recurso para o pré-treinamento, no qual o modelo aprende representações gerais ao reconstruir autonomamente as informações.

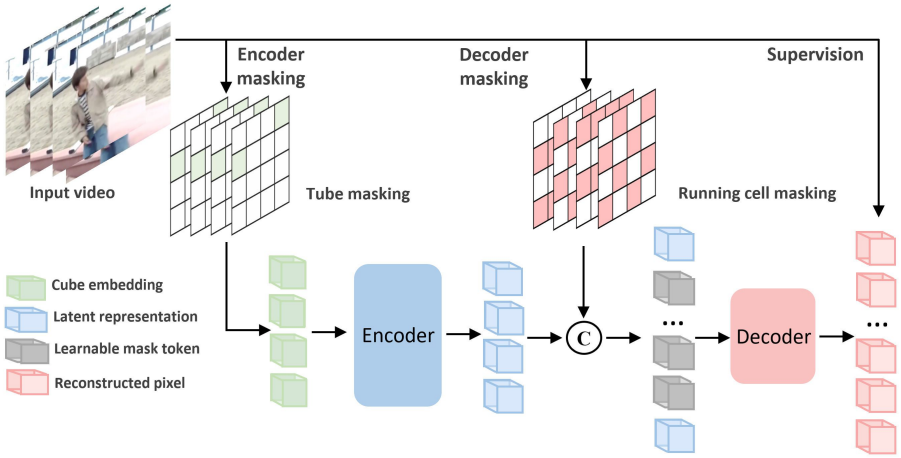
Em relação à arquitetura, o *VideoMAEv2* destaca-se ao adotar a estratégia de *dual masking* que aplica máscaras tanto no *encoder*, quanto no *decoder*, como demonstra a Figura 1. Essa abordagem propõe uma solução direta para o gargalo computacional de seu antecessor. No *VideoMAE v1*, embora o *encoder* processasse apenas 10% dos *tokens* visíveis utilizando a técnica de *tube masking*[2], o *decoder* exigia o processamento de 100% dos *tokens* para reconstruir as informações visuais.

Essa nova versão preserva a técnica de *tube masking* no *encoder*, herdada do *VideoMAEv1*. Contudo, a estratégia se diferencia ao implementar uma técnica de mascaramento diferente no *decoder* chamada *running cell masking*. Essa técnica faz com que ele processe um conjunto menor, com cerca de 50% dos *tokens* mascarados, percentual este que demonstrou um melhor equilíbrio entre precisão e eficiência durante os testes (TONG et al., 2022; GRUTSCHUS et al., 2024).

[1] Pontos importantes de um arquivo.

[2] Técnica que mascara as mesmas posições espaciais em todos os *frames*.

Figura 1 – Arquitetura do *VideoMAEv2*



Fonte: (WANG et al., 2023)

2.5 Reconhecimento de Atividades Humanas

O reconhecimento de ações humanas (*Human Action Recognition* - HAR) é uma subárea de ML e visão computacional que busca compreender essas ações por meio da análise de vídeos ou sequências de imagens com base em atributos espaciais ou temporais (GAMMULLE et al., 2023; DUONG; LE; HOANG, 2023; SUN et al., 2022).

O desenvolvimento de aplicações capazes de trabalhar com HAR possibilitou diversas soluções viáveis no mundo real. Sistemas baseados em HAR podem ser implementados em várias situações, como tarefas de monitoramento em câmeras de segurança ou detecção de quedas. O estudo e desenvolvimento de sistemas de HAR têm como foco vídeos RGB ou em escala de cinza, mas não se limitam apenas a isso. Para trabalhar com HAR, podem-se utilizar vários tipos de dados, como informações de sensores de *smartphones*, sensores de profundidade ou até por meio de ondas *Wi-Fi* (WANG et al., 2015; SUN et al., 2022).

2.6 Detecção de Quedas

Uma queda pode ser considerada como um evento imprevisível que leva o participante a um nível inferior, como chão ou piso. Para detectar este problema, duas abordagens mais comuns são utilizadas, sendo elas: sistemas não vestíveis (*Non-Wearables System*) e sistemas vestíveis (*Wearable Systems*) (USMANI et al., 2021).

Areeb et al. (2023) menciona que *Non-wearables System* são sensores não invasivos localizados ao redor do ambiente. Esses sensores são divididos em sistemas baseados em visão, focados no monitoramento contínuo que utiliza câmeras e sensores de pressão no chão. Eles são colocados em locais específicos e, caso a pessoa saia da área demarcada, será impossível detectar uma queda.

Já *wearables system* são sensores que funcionam anexados ao usuário, monitorando sequências de ações realizadas; estes são ainda mais invasivos e ficam presos junto ao corpo.

Sendo acelerômetros, giroscópios e magnetômetros os sensores mais utilizados para detectar quedas (KULURKAR et al., 2023).

3 TRABALHOS RELACIONADOS

De acordo com a necessidade de projetos que garantissem a autonomia e segurança de pessoas, muitos trabalhos na área de reconhecimento de atividades humanas foram desenvolvidos. A seguir, serão abordados alguns trabalhos nesta temática.

Espinosa et al. (2020) empregam Redes Neurais Convolucionais (*Convolutional Neural Networks* - CNN) para a classificação de quedas no *UP-Fall Detection Dataset*. *Dataset* coletado em ambiente laboratorial controlado que reúne dados de sensores *wearables* e sistemas de vídeo. O estudo comparou a eficácia de configurações baseadas em câmera única e sistemas multicâmera. Metodologicamente, aplicou-se a técnica de janelamento (*Windowing*) para capturar dependências temporais nas amostras. A arquitetura da CNN foi estruturada em três blocos convolucionais intercalados com camadas *max-pooling* $2D$ para extração de *features*, seguidos por três camadas *fully connected* responsáveis pela detecção. Para o treinamento, o *dataset* foi particionado em 67% (42.000 amostras) para treino e 33% (21.000 amostras) para teste, consistindo em imagens em escala de cinza (38x51 *pixels*), pré-processadas com *optical flow*. O treinamento durou 50 épocas, com o otimizador *Adam* e a função de perda *binary cross-entropy*[3]. A análise avaliou três cenários: **i.** comparação da CNN com modelos clássicos (SVM, *Random Forest*, MLP, KNN (*K-Nearest Neighbors*); **ii.** comparação com sistemas de detecção de quedas baseados em câmera única; **iii.** avaliação da CNN na classificação de atividades e quedas usando múltiplas câmeras. Os modelos clássicos demonstraram baixa eficácia na detecção de quedas, com um baixo *F1-score*. SVM obteve $14,06\%$, RF $14,37\%$, MLP $9,94\%$ e KNN, obtendo o melhor resultado com $15,27\%$. Em contraste, a CNN utilizando múltiplas câmeras, alcançou valores altos, chegando aos $97,43\%$ de *F1-score*.

Ha et al. (2022) propõe uma abordagem multimodal para detecção de quedas, combinando técnicas de extração de *features* de sensores e câmeras de monitoramento baseadas no *UP-Fall Detection Dataset*. A etapa de pré-processamento envolveu a limpeza dos dados, com a remoção de duplicatas e valores ausentes. Além de uma sincronização temporal entre os registros de sensores e os quadros de vídeo. O conjunto de dados resultante consolidou 258.113 amostras, divididas entre 28 atributos e seus rótulos. Para processamento dos 28 atributos, os autores desenvolveram uma rede neural composta por uma camada *fully connected*[4] com 2000 neurônios seguida por uma camada com 600 neurônios, ambas utilizando normalização e função de ativação *Relu*[5]. Para regularização, aplicou-se uma camada de *dropout* de $0,2$ antes da camada de saída *Softmax* de tamanho 12. Além da criação da rede neural, foram considerados dois algoritmos baseados em árvores de decisão, como o *XGBoost* e *CatBoost*, para a classificação baseada em

[3] Função de perda entre a quantidade de amostras reais e as preditas pelo modelo.
[4] Rede em que todas as camadas de aprendizado são conectadas.
[5] Unidade Linear Retificada: função de ativação que introduz não linearidade ao modelo.

sensores. No aspecto visual, modelos de CNN foram empregados para extrair características da câmera 1 e 2. A abordagem multimodal final consistiu na concatenação das *features* visuais e dos dados de sensores em uma única rede neural. Os experimentos avaliaram cenários unimodais e multimodais. Na análise com sensores, os algoritmos *XGBoost*, *CatBoost* e MLP superaram 99% de desempenho. Nos testes baseados em visão, a Câmera 1 e Câmera 2 alcançaram, respectivamente, $99,1\%$ e $99,3\%$ de acurácia. Além disso, a junção das duas câmeras alcançou $99,16\%$ de *F1-Score*.

Em (GALVÃO et al., 2021), conduziram uma investigação sobre eficiência de sistemas de detecção automática de quedas, focando no uso de modelos multimodais e redes neurais profundas para aprimorar o desempenho de sensores. O estudo utilizou imagens RGB e dados de acelerômetros dos conjuntos de dados *UR-Fall Detection* e *UP-Fall Detection*. Para o processamento das imagens, foram propostas três abordagens: uma CNN-2D, um método baseado em agrupamento de instâncias e uma arquitetura híbrida combinando CNN-2D com LSTM. A extração de *features* utilizou uma janela deslizante de 55 *frames* para cada amostra. No pré-processamento, as imagens foram convertidas para escala de cinza e redimensionadas para resolução 40*x*40 *pixels*. Já os dados de acelerômetros utilizados consistiram em gravações com frequência de 60Hz. Para o treinamento do modelo, o *dataset* foi dividido em 70% para treinamento e 30% para teste. Os resultados no *UR Fall dataset* utilizando apenas imagens destacaram a CNN-2D com o melhor desempenho ($95,58\%$ de acurácia), enquanto o modelo que utilizou apenas dados de acelerômetros obteve o pior resultado $93,37\%$ de acurácia. Resultados similares ao *UP-Fall Dataset*, em que a CNN-2D alcançou $99,99\%$ de acurácia, superando a abordagem de LSTM com acelerômetros, que registrou $94,40\%$.

O trabalho de Grutschus et al. (2024) investigou a aplicação do *VideoMAEv2* para a detecção de quedas no *dataset* HQFS. A metodologia proposta, chamada *Cutup and Detect*, segmenta os vídeos brutos por meio de uma janela deslizante, sem padrões complexos de amostragem. Também foi utilizado *Gaussian Sampling* para extrair pontos principais dos vídeos não editados. Clipes estes que foram amostrados e rotulados por prioridade: a classe de "queda"possui maior prioridade, seguida por "deitado"e "atividades do cotidiano"ou "outros". Desta forma, se um clipe possui os rótulos de "queda"e "deitado", ou "queda"e "atividades do cotidiano", o clipe inteiro será rotulado como queda. Antes de extrair as representações numéricas dos dados (*embeddings*) com o *VideoMAEv2*, foi realizado um pré-processamento em que são amostrados 16 *frames* por clipe, requisito do *backbone* do modelo. Durante o treinamento, os quadros foram redimensionados para escala *W x* 224, com aplicação de recortes aleatórios de 224 *x* 224 *pixels* e uma *augmentation* via *flip* horizontal com probabilidade de 50%. Na validação, o *flip* foi substituído por um *center-crop* para o tamanho dos *frames*. Durante a inferência, adotou-se a estratégia de *three-crops* horizontais de tamanho 224 *x* 224 a partir dos quadros redimensionados. A predição final é obtida pela média de 5 amostras por clipe e 3 recortes por amostra, totalizando 15 predições ao todo. Ao fim, foram calculadas a média e variância de cada *pixel* individualmente, com o objetivo de normalizar os *inputs* do modelo.

A arquitetura utilizou uma versão destilada do *backbone* ViT-B com 87 milhões de

parâmetros. A sua *Classification Head* é formada por uma única camada *fully connected* treinada com a função de perda de *cross-entropy* implementado por meio do *framework MMAction2*. Em termos de desempenho, o método *Cutup and Detect* alcançou um *F1-score* médio de $0,94$ e *recall* de $0,88$, enquanto o *Gaussian Sampling* teve $0,96$ de *F1-score* e $0,82$ de *recall*. O tempo de inferência foi otimizado, reduzindo de 5 para 2.46 segundos em comparação com outros trabalhos. Segundo os autores, os resultados experimentais superaram o estado da arte para classificação de vídeo utilizando o HQFS com a alta pontuação do *F1-score* de $0,96$. Contudo, o trabalho reconhece limitações ao sinalizar um possível vazamento de informações entre os dados de treino e teste devido aos vídeos gravados no mesmo cenário em ângulos diferentes que podem ter inflado os resultados.

O Quadro 1 apresenta uma análise comparativa entre o trabalho de (ESPINOSA et al., 2020) que utilizou CNNs para classificar quedas utilizando o *dataset UP-Fall* que engloba dados de dispositivos *wearables* e vídeos de monitoramento, o trabalho de (HA et al., 2022) que seguiu uma linha parecida, extraindo *features* do *Up-Fall* por meio de uma CNN utilizando os algoritmos *XGBoost* e *CatBoost*. (GALVÃO et al., 2021) propões uma metodolia onde serão usados apenas dados de imagens RGB e acelerômetros obtidos do *UR-fall* e *UP-Fall*, nesta abordagem foram utilizadas CNN2D e uma combinação da mesma com LSTM. Já (GRUTSCHUS et al., 2024) utiliza o *bakcbone* ViT-B do *VideoMAEv2* implementado através do *framework MMAction2* utilizando o *dataset* HQFS para realização de classificação multiclasse.

Quadro 1 – Principais abordagens dos trabalhos relacionados.

	Espinosa et al. (2020)	**Galvão et al. (2021)**	**Ha et al. (2022)**	**Grutschus et al. (2024)**	**Abordagem Proposta**
Dataset(s)	UP-Fall	UR Fall, UP-Fall	UP-Fall	HQFS	**HQFS**
Abordagem	Multi-câmera (RGB)	Multimodal (RGB+IMU)	Multimodal (Wearable + RGB)	Câmera Única, Vídeo	**Câmera Única, Vídeo**
Arquitetura	CNN 2D Simples	CNN 2D + LSTM/CNN 1D	CNN + MLP/XGBoost	VideoMAEv2 (ViT-B) & FC head	**Shallow Network**
Features	Optical Flow	CNN + LSTM (Temporal)	Embeddings CNN1D/CNN2D	Embeddings VideoMAEv2	**Features VideoMAEv2**
Tarefa	Detecção/Classif (11 atividades)	Detecção Binária (Queda vs. ADLs)	Classif. Multiclasse	Classif. Multiclasse (Fall, Lying, Other/ADL)	**Classif. Binária**
Métrica	Acc, Prec, Recall, Spec, F1-Score	Acc, Spec, Prec	Acc, F1-Score, Recall, F1-score	Recall, Spec, F1-Score	**Acc, Prec, Recall, Spec, F1-Score, AUC**

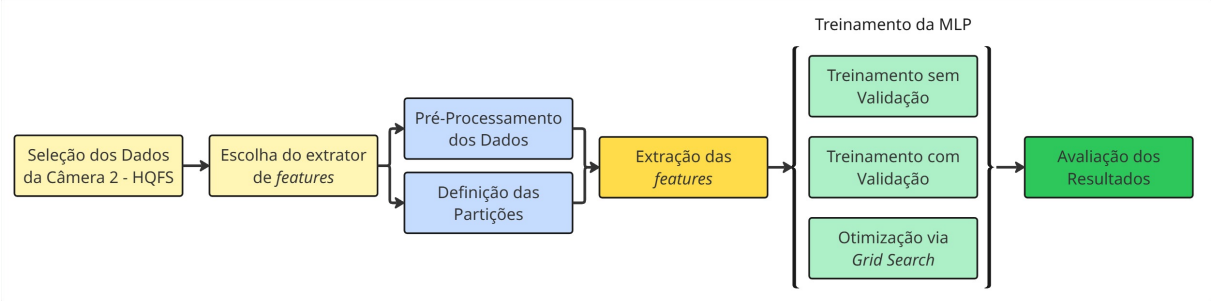
Fonte: Elaborado pelos autores.

O presente estudo trabalha com a implementação do *backbone* ViT-B do *VideoMAEv2*, implementado através do *framework* PyTorch e avaliação da qualidade das *features* e da capacidade preditiva da MLP na tarefa de classificação de quedas. O diferencial desta pesquisa dentre os trabalhos analisados está na utilização de um *Vision Transformer* para extrair atributos de vídeo e, com uma rede neural rasa (*shallow network*), avaliar a qualidade desta representação de atributos de vídeo. Além disso, diferentemente do estudo de (GRUTSCHUS et al., 2024), em que uma cena poderia estar tanto no conjunto de treino quanto no conjunto de teste através da visão de diversas câmeras, este trabalho garante que não haja essa ocorrência ao utilizar exclusivamente vídeos da Câmera 2 do HQFS, visando avaliar a capacidade de generalização do modelo, passo fundamental para viabilizar a construção de sistemas de detecção de quedas mais eficientes e robustos.

4 METODOLOGIA

Esta pesquisa propõe uma metodologia baseada em *deep learning* com o objetivo de desenvolver um modelo que seja capaz de identificar quedas. Com este objetivo, a seguinte seção detalha as estratégias, os materiais e os métodos empregados, abrangendo o conjunto de dados, a arquitetura do modelo e os critérios de avaliação. A Figura 2 apresenta as principais etapas seguidas para construção deste trabalho.

Figura 2 – Fluxograma Metodológico



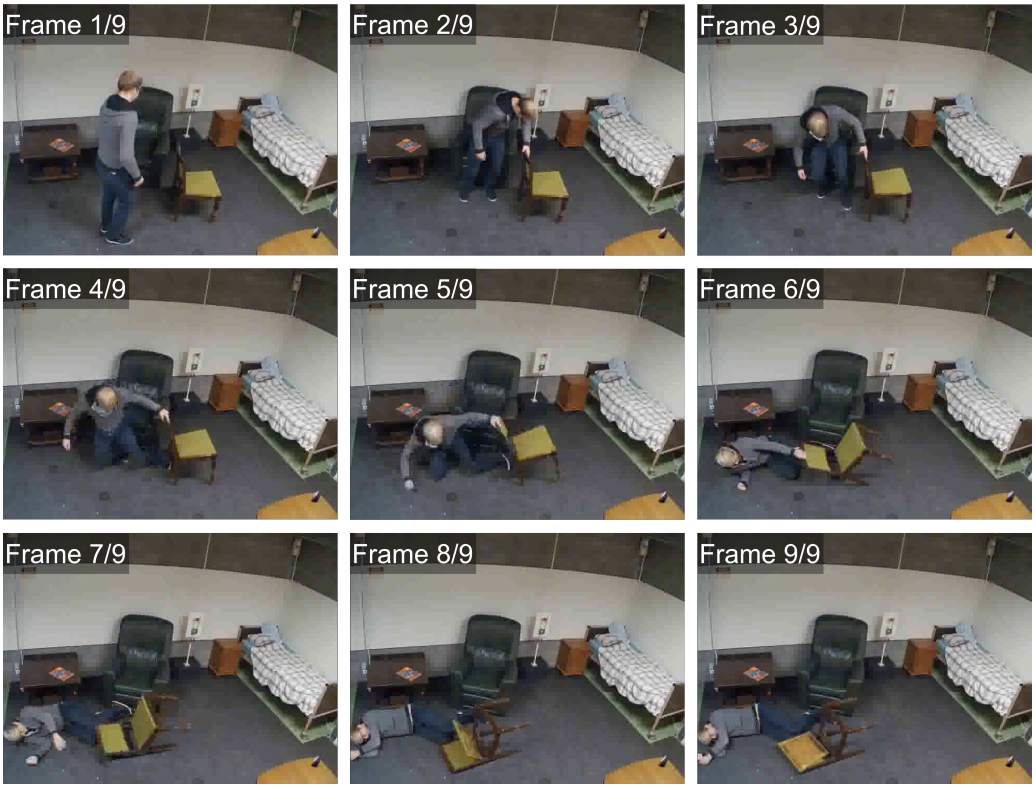
Fonte: Elaborado pelos autores.

4.1 Dataset

High Quality fall simulation data ou HQFS é um *dataset* desenvolvido em um ambiente laboratorial. O processo de criação envolveu mobiliar um quarto para se assemelhar a um ambiente típico de uma casa de repouso, com o objetivo de registrar cenas de quedas, como ilustrado na sequência de *frames* da Figura 3.

Durante a gravação, foram instaladas cinco *webcams* em pontos estratégicos, capturando imagens a 12 *fps* com resolução de 640x480. O *dataset* compreende 55 cenários distintos de quedas, distribuídos em 5 categorias, considerando fatores como necessidade de auxílio para locomoção, velocidade da queda, possibilidade de mover objetos durante a queda e caracterização

Figura 3 – Cenário de Queda 13 - Câmera 2.



Fonte: Adaptado de (BALDEWIJNS et al., 2016).

de pose inicial e final do indivíduo durante uma queda. Nos cenários que simularam a necessidade de auxílio para locomoção, foram realizadas gravações com andador, cadeira de rodas e também sem nenhum tipo de assistência. Em outros cenários em que o deslocamento de objeto durante a queda foi considerado, é possível observar elementos como cobertores, cadeiras, andadores e cadeiras de rodas. As posições iniciais das quedas foram categorizadas do seguinte modo: em pé, sentado, curvando-se, agachado e em transição de sentar e ficar em pé. Para a análise das posições finais, foram considerados dois cenários: a permanência no chão e levantar-se depois de uma queda. A duração média de cada cenário foi de 2 minutos e 24 segundos. Cada uma das cinco câmeras registrou um total de $2:25:54h$ de dados relacionados a quedas. Adicionalmente foram gravadas também atividades do cotidiano, como comer, dormir ou trocar de roupa, por exemplo. Foram gravados 17 cenários distintos de HAR, com duração média de $20:39min$ por cenário. Além disso, diante do desafio em simular com fidelidade quedas reais, foram adicionados alguns elementos como quedas parcialmente ou totalmente fora de visão e cenários com mais de uma pessoa em cena (BALDEWIJNS et al., 2016).

4.2 Extração de *Features* com VideoMAEv2

Para extração das *features*, emprega-se o *VideoMAEv2* com *backbone ViT-B*, versão base do modelo com 86.2 milhões de parâmetros, implementado a partir do repositório *OpenGVLab* da plataforma *Hugging Face*. Nesta configuração, adota-se uma abordagem metodológica de

Transfer Learning, em que o modelo atua como extrator de características, mantendo-se os pesos congelados conforme obtidos do pré-treinamento no conjunto *UnlabeledHybrid*. O *ViT-B* processa os clipes de vídeo para gerar as representações vetoriais que encapsulam tanto informações espaciais (o que está no quadro) quanto temporais (como o conteúdo muda ao longo do tempo), sendo estas cruciais para a modelagem e classificação de uma queda.

Escolha e Pré-processamento dos Dados: Para a composição do conjunto de dados, restringiu-se a análise exclusivamente aos vídeos da Câmera 2 do *dataset High Quality Fall Simulation Data (HQFS)*. O fluxo de pré-processamento abrangeu 53 vídeos do conjunto de treino e 18 vídeos do conjunto de teste. A leitura dos arquivos ocorreu sequencialmente, alimentando um *buffer* que armazena 16 *frames* RGB, no qual cada *frame* foi redimensionado para a resolução de 224*x*224 *pixels*. Estes *chunks* foram submetidos ao *VideoMAEImageProcessor*, e o tensor resultante, inicialmente com a dimensão ($Lote \times Frames \times Canais \times Altura \times Largura$), sofreu uma permutação dimensional para se adequar ao formato de entrada do modelo ($Lote \times Canais \times Frames \times Altura \times Largura$).

Implementação e Extração das *Features*: A extração das *features* foi conduzida através de um *pipeline* desenvolvido no *framework PyTorch*, executado em ambiente *Google Colab* configurado com a *GPU T4*. Utilizou-se a biblioteca *OpenCV (Open Source Computer Vision Library)* para o gerenciamento dos dados visuais, estruturando listas de 16 *frames* por *chunk*. Na etapa de inferência, o modelo projetou cada segmento em um tensor de saída com dimensão (1, 768), constituindo o *embedding* do respectivo *chunk*. Em seguida, esses vetores concatenados foram sequencialmente e convertidos para o formato *NumPy* e salvos em um arquivo *.npy* único para cada vídeo processado.

Aquisição e Adaptação dos *Ground Truths* e Construção dos *Splits*: Os *Ground Truths (GTs)* re-rotulados do *dataset* HQFS e a definição dos *splits* de treino e teste foram adotados conforme estabelecido por (PEREIRA; MAIA, 2025). Este trabalho prévio implementou uma segmentação de 16 *frames* por *chunks* de vídeo, alinhando-se diretamente aos requisitos de entrada do modelo extrator aqui utilizado. A estrutura de dados originais compreendia 69 sequências, sendo 17 delas referentes às atividades de vida diária (ADL) e 52 contendo eventos de *Fall*. No entanto, durante essa análise dos dados, foi observada uma ausência nos arquivos de *fall34* e *fall46* para a câmera 2, logo seus vídeos não serão utilizados nessa pesquisa. Para adaptar os GTs, construiu-se uma função para transformar os rótulos de *frame* em rótulos de *chunk*. O objetivo dessa função foi gerar um vetor de *labels Y*, em que cada elemento se alinhasse com o *chunk* de 768 *features* retornado pelo *VideoMAEv2*. Para isso, criou-se um *array* temporário de zeros com o tamanho exato necessário e, em seguida, os *frames* originais foram copiados para este *array*. Caso a dimensão fosse menor que a necessária, realizou-se um *padding*; se fosse maior, aplicou-se o truncamento para o limite necessário, garantindo assim a consistência dimensional entre o vetor de *features* e o vetor de *labels*.

Organização das Partições Finais: Para a definição da classe de cada *chunk*, aplicou-se uma estratégia de agregação baseada na presença de eventos. A matriz de rótulos original foi processada de modo que, para cada segmento, calculou-se o somatório das anotações dos

quadros correspondentes. Caso a soma resultasse positiva, indicando a ocorrência de queda em pelo menos um quadro do intervalo, o *chunk* inteiro foi categorizado como classe anômala (1); caso contrário, foi atribuído à classe normal (0). Concluído o processamento, os vetores de *features* e seus respectivos rótulos foram concatenados para constituir as matrizes finais de treino e teste, resultando nas seguintes dimensionalidades: $X_{\text{train}} \in \mathbb{R}^{42132\times768}$, $y_{\text{train}} \in \mathbb{R}^{42132}$, $X_{\text{test}} \in \mathbb{R}^{15174\times768}$ e $y_{\text{test}} \in \mathbb{R}^{15174}$.

Configuração do Fluxo de Dados: Para adequar os dados ao fluxo de entrada da rede, as matrizes definidas na Seção 4.2 foram convertidas em tensores e encapsuladas em objetos *TensorDataset*. O gerenciamento do fluxo de dados foi realizado por meio de iteradores (*DataLoader*) do *framework PyTorch*, configurados com *batch size* distintos: 16 amostras para o conjunto de treinamento e 8 para teste.

4.3 Experimentos

Esta pesquisa emprega redes neurais rasas (*shallow networks*), redes caracterizadas por possuírem apenas uma camada oculta (KŮRKOVÁ; SANGUINETI, 2016). Foram realizados experimentos distintos (Tabela 1) para o treinamento supervisionado de redes neurais *Multi-Layer Perceptron* (MLP).

Tabela 1 – Esquemas de treinamento empregados.

Exp	*Split* de Validação	Early Stopping	Grid Search
Exp 1 (*baseline*)			
Exp 2a (*baseline*)	✓		
Exp 2b (*baseline*)	✓	✓	
Exp 3	✓	✓	✓

Fonte: Elaborado pelos autores.

Os esquemas de treinamento empregados diferenciam-se quanto ao uso de um *split* de validação na etapa de treinamento da rede, regularização via *early stopping*, e otimização de hiperparâmetros via *Grid Search*.

4.3.1 Experimento 1 (Baseline): MLP Fixa com Splits de Treino e Teste

A arquitetura proposta consiste em uma rede neural MLP composta por uma camada densa (*shallow network*), com o objetivo de executar apenas treino e teste. A rede é estruturada em uma camada de entrada que recebe as 768 *features* e as projeta para 128 neurônios, seguida de uma função de ativação *ReLU*. A camada de saída retorna um único valor para as 128 *features* projetadas. Como *loss function*, utiliza-se a *Binary Cross Entropy* (BCE) (DR.A, 2020) juntamente com o otimizador *Adam* (SUN et al., 2024) e taxa de *learning rate* de $0,001$. A função de perda é definida matematicamente conforme a Equação (1):

$$L = -[y \cdot \log(p) + (1-y) \cdot \log(1-p)] \tag{1}$$

onde y representa o rótulo real (0 ou 1) e p a probabilidade prevista pelo modelo.

4.3.2 Experimento 2 (Baseline): MLP Fixa com Splits de Treino, Val, Teste

O segundo experimento incluiu a separação de 20% dos dados de treino para validação, permitindo o monitoramento da função de perda e aplicação do critério de parada. Neste cenário, ajustou-se a taxa de aprendizado para 10^{-4} e definiu-se uma paciência de 10 épocas para o *early stopping*. Ambos os *loops* de treinamento são definidos para um máximo de 100 épocas. Para analisar o impacto do *early stopping* no desempenho do modelo, executa-se o experimento sem a ativação desta regularização (experimento 2a) e com a ativação da mesma (experimento 2b).

4.3.3 Experimento 3: Otimização de hiperparâmetros via Grid Search

Nesta etapa, aplicou-se uma estratégia de otimização exaustiva via *grid search* para identificar a configuração ideal de hiperparâmetros. A configuração de hiperparâmetros otimizada, resultante deste processo, é apresentada na Tabela 2.

Tabela 2 – Tabela de Parametrização do Grid Search.

Hiperparâmetro	**Espaço de Busca**	**Valor Otimizado**
Neurônios (Camada Oculta)	{64, 128, 256}	64
Learning Rate	$\{10^{-3}, 10^{-4}\}$	10^{-4}
Dropout	{0,3, 0,5}	0.3
Paciência (*Early Stopping*)	{5, 10, 15}	15

Fonte: Elaborado pelos autores.

O espaço de busca contemplou 36 combinações distintas, variando: (i) a quantidade de neurônios da camada oculta (64, 128, 256); (ii) a taxa de *dropout* $(0,3, 0,5)$, inserida para mitigar o *overfitting*; (iii) o *learning rate* $(0,001, 0,0001)$; e (iv) a paciência do *early stopping* (5, 10, 15 épocas). Com o objetivo de priorizar a detecção de quedas, classe minoritária do *dataset*, o *f1-score* foi definido como a métrica central de desempenho, orientando tanto a otimização, quanto o critério de *early stopping*. Tal escolha deve-se à capacidade desta métrica em representar o equilíbrio harmônico entre *precision* e *recall*. A execução computacional desta foi conduzida nos ambientes *Google Colab* e *Kaggle*.

4.4 Métricas de Avaliação

Este trabalho emprega métricas consolidadas na literatura de classificação supervisionada, alinhando-se aos métodos de avaliação adotados em pesquisas similares (SATHYANARAYANAN, 2024; GRUTSCHUS et al., 2024; BALDEWIJNS et al., 2016). A análise de desempenho fundamenta-se na matriz de confusão, que compara as previsões do modelo com os valores reais (*ground truth*). Desta forma, todos os modelos foram avaliados considerando os valores

de: Verdadeiros Positivos (TP), Verdadeiros Negativos (TN), Falsos Positivos (FP) e Falsos Negativos (FN). As métricas derivadas destes valores são detalhadas a seguir:

$$\begin{aligned} \text{Acc} &= \frac{TP+TN}{TP+TN+FP+FN}, \qquad \text{Prec} = \frac{TP}{TP+FP}, \\ \text{Rec} &= \frac{TP}{TP+FN}, \qquad \text{Spec} = \frac{TN}{TN+FP}, \\ \text{F1} &= 2 \cdot \frac{\text{Prec} \cdot \text{Rec}}{\text{Prec} + \text{Rec}}. \end{aligned} \tag{2}$$

Além disso, utilizou-se a métrica AUC, comumente utilizada para avaliação de modelos em tarefas classificação binária para avaliar o poder discriminatório do classificador em diferentes limiares de classificação.

5 RESULTADOS

Esta seção apresenta os resultados obtidos a partir dos experimentos de classificação binária de quedas descritos na metodologia. Inicialmente são detalhados os desempenhos dos modelos (*baselines*) sem otimização de hiperparâmetros (Experimentos 1 e 2), seguidos pelas análises das curvas de aprendizado e das matrizes de confusão. A avaliação quantitativa baseia-se nas métricas de acurácia, precisão, *recall*, *F1-score*, *specificity* e AUC-ROC, permitindo aferir a eficácia das *features* extraídas pelo *VideoMAEv2* no *dataset* HQFS.

5.1 Experimento 1 (*Baseline*): MLP Fixa com *Splits* de Treino e Teste

Para estabelecer uma *baseline* de desempenho, a arquitetura inicial da MLP foi avaliada no conjunto de teste. O modelo alcançou uma acurácia de $0,994$, um valor que é alto, porém enganoso devido ao severo desbalanceamento das classes. As métricas para a classe de interesse (Queda) apresentadas na Tabela 3 evidenciam as limitações da abordagem inicial. O modelo alcançou uma *precision* de $0,320$, bem como um *recall* de $0,232$ e um *F1-score* de $0,269$. O baixo *recall* é o ponto mais crítico, que indica que o classificador falha em identificar a maioria dos eventos de queda. A matriz de confusão (Tabela 4) confirma essa análise, mostrando que apenas 16 quedas foram corretamente classificadas, enquanto 53 não foram detectadas. Em contrapartida, o classificador demonstrou alta eficácia na classe majoritária, identificando corretamente 15.071 instâncias ADL.

5.2 Experimento 2 (*Baseline*): MLP Fixa com *Splits* de Treino, Validação e Teste

5.2.1 Treino sem Early Stopping

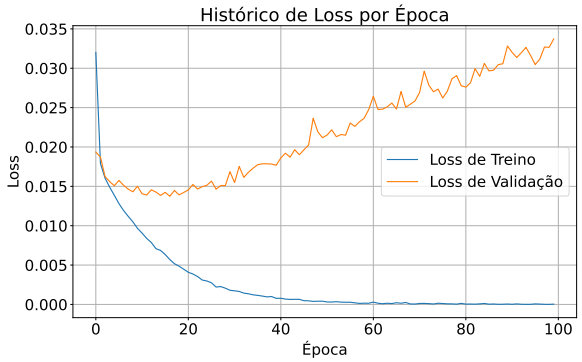
Na segunda configuração experimental, a MLP foi treinada com os dados particionados, 80% para treino e 20% para validação, e com *learning rate* ajustada para 10^{-4}. O early stopping não foi ativado nesta etapa, permitindo que o modelo treinasse por todas as 100 épocas.

A avaliação no conjunto de teste revelou um avanço em relação ao modelo *baseline*. O modelo alcançou uma *precision* de $0,571$, um *recall* de $0,290$ e um *F1-score* de $0,385$. O aumento mais significativo ocorreu na *precision*, indicando uma melhora nas predições de queda. A métrica AUC manteve-se elevada, o que corrobora o alto potencial discriminativo das *features* extraídas. A matriz de confusão quantifica essa melhoria: o número de Verdadeiros Positivos (quedas reais detectadas) aumentou de 16 para 20, enquanto o número de Falsos Positivos foi reduzido, caindo de 34 para 15.

5.2.2 Treino com Early Stopping

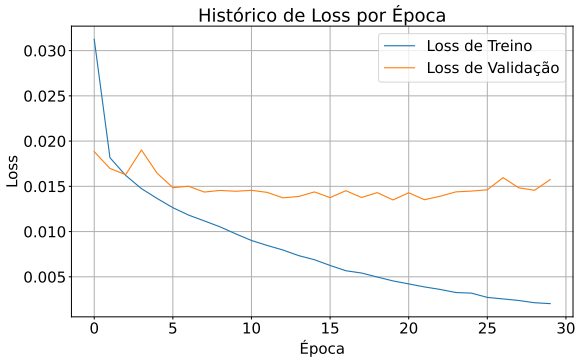
Para reduzir o risco de *overfitting* e otimizar o tempo computacional, o modelo foi treinado com a implementação de *early stopping*. A estratégia consistiu em monitorar o *loss* no conjunto de validação, definindo-se uma paciência de 10 épocas. Desta forma, o treinamento foi interrompido automaticamente quando não houve melhora durante as 10 épocas. A eficácia dessa abordagem, tanto na regularização quanto na otimização de recursos, é demonstrada visualmente pela comparação das curvas de aprendizado: a Figura 4 (que ilustra o treino completo de 100 épocas) e a Figura 5, que evidencia a interrupção do processo na época 30, ponto em que o modelo atingiu seu desempenho ótimo, antes de iniciar o *overfitting*.

Figura 4 – MLP Sem *Early Stopping*



Fonte: Elaborado pelos autores.

Figura 5 – MLP Com *Early Stopping*



Fonte: Elaborado pelos autores.

A utilização do *early stopping*, embora tenha otimizado o tempo de treinamento interrompendo na época 30, não resultou em melhora de desempenho no conjunto de teste. O modelo registrou uma *precision* de *0.408* e um recall de *0.290*, resultando em um *F1-score* de *0.339*. Resultados esses inferiores aos $0,385$ de *F1-score* obtidos pelo modelo anterior treinado por 100 épocas. A matriz de confusão (Tabela 4) confirma que não houve ganho na detecção da classe minoritária ao registrar 20 quedas corretamente identificadas. Isso sugere que, embora o *early stopping* tenha evitado o *overfitting* visível no *loss* de validação (Figura 5), o ponto ótimo para a métrica *loss* não coincidiu com o ponto ótimo para o *F1-score*.

5.3 Experimento 3: MLP com hiperparâmetros otimizados via *Grid Search*

Otimização de Hiperparâmetros via *Grid Search*: A etapa final de otimização, conduzida via *Grid Search* conforme detalhado na Seção 4.3.3, avaliou 36 combinações de hiperparâmetros. O critério de seleção foi o *f1-score* no conjunto de validação, monitorado também para o *early stopping*. A combinação que apresentou melhor desempenho foi: 64 neurônios na camada oculta, *early stopping* de 15 épocas, *learning rate* de $0,0001$ e *dropout* de $0,3$.

Treino e avaliação de desempenho a partir dos parâmetros selecionados: A avaliação do modelo otimizado no conjunto de teste (Tabela 3) apresentou um *F1-score* final de $0,351$. O resultado mais notável foi o aumento da *precision* para $0,607$, o que indica que, quando o modelo prevê uma queda, ele está correto em mais de 60% das vezes. Contudo, este ganho em confiabilidade foi alcançado às custas de uma redução no *recall*, que caiu para $0,246$.

Tabela 3 – Comparativo de desempenho entre os experimentos.

Exp	Acc	Precision	Recall	F1-score	Spec	AUC
Exp 1 *Baseline* (S/ Val.)	0,994	0,320	0,232	0,269	0,997	0,906
Exp 2a *Baseline* (C/ Val.)	**0,996**	0,571	**0,290**	**0,385**	**0,999**	0,907
Exp 2b *Baseline* (C/ Val. + ES)	0,995	0,408	**0,290**	0,339	0,998	**0,928**
Exp 3 *Grid Search*	**0,996**	**0,607**	0,246	0,351	**0,999**	0,921

Fonte: Elaborado pelos autores.

A matriz de confusão (Tabela 4) expõe esse comportamento, indicando que o modelo otimizado reduziu o número de falsos positivos para apenas 11, mas ao custo de aumentar o número de falsos negativos para 52. O que explica o porquê do *F1-score* permanecer em $0,351$.

Tabela 4 – Comparativo das Matrizes de Confusão entre os experimentos.

	Exp 1		Exp 2a		Exp 2b		Exp 3	
Classes	ADL	FALL	ADL	FALL	ADL	FALL	ADL	FALL
ADL	15071	34	15090	15	15076	29	15094	11
FALL	53	16	49	20	49	20	52	17

Fonte: Elaborado pelos autores.

5.4 Comparação com a Literatura

A Tabela 5 apresenta um comparativo com trabalhos relevantes do estado da arte. O estudo de (BALDEWIJNS et al., 2016), por exemplo, avaliou o *dataset* HQFS aplicando um algoritmo de subtração de *background* e filtro de partículas baseados em três funções de medição: imagem binária de primeiro plano, correlação com o histograma e detecção de corpo superior. Uma vez o individuo localizado, essas informações são submetidas a uma SVM para classificação. Destaca-se também o trabalho de (GRUTSCHUS et al., 2024) que aplicou *fine-tuning* no

VideoMAEv2 utilizando o HQFS, alcançando valores de *recall* superiores a $0,90$ ao utilizar abordagens complexas. Em contraste, o presente modelo, baseado em uma *Shallow Network* (MLP), apresentou um desempenho conservador quanto à detecção da classe Queda, que foi refletido em um *recall* de $0,246$.

Tabela 5 – Comparação com o estado da arte em pesquisas para classificação de quedas.

Trabalho	**Prec**	**Recall**	**F1-score**	**Spec**	**AUC**
(GRUTSCHUS et al., 2024)	0,960	0,930	0,940	0,880	-
(BALDEWIJNS et al., 2016)	0,380	0,510	-	-	0,350
Este Trabalho	**0,607**	**0,246**	**0,351**	**0,999**	**0,921**

Fonte: Elaborado pelos autores.

Em suma, a estratégia de otimização via *Grid Search* orientada pelo *F1-score* produziu um classificador de maior confiabilidade preditiva. Embora o modelo demonstre cautela na detecção, ele minimiza a ocorrência de falsos positivos (*specifity* de $0,999$). Adicionalmente, a AUC de $(0,921)$ confirma que a qualidade das *features* extraídas pelo *VideoMAEv2* foi preservada, independentemente da configuração do classificador. Desta forma, este estudo valida a arquitetura de dois estágios: extração de *features* com *VideoMAEv2* seguida de classificação por MLP, como uma abordagem promissora para análise de vídeos de câmera de monitoramento, pavimentando o caminho para o desenvolvimento de sistemas de detecção de quedas mais precisos.

6 CONCLUSÃO E TRABALHOS FUTUROS

O trabalho avaliou a solidez do *pipeline* proposto (*VideoMAEv2* + *Shallow Network*) como base para sistemas de monitoramento de quedas. As conclusões obtidas, juntamente com as limitações identificadas e as propostas de trabalhos futuros, são detalhadas a seguir.

Os resultados corroboram o principal pilar desta pesquisa: as *features* extraídas via rede *VideoMAEv2* possuem um altíssimo poder discriminatório para a detecção de quedas. Isso foi consistentemente evidenciado pela métrica AUC, que sempre se manteve acima de $0,90$ em todas as variações de modelos utilizados. Porém, a análise das métricas revelou desafios importantes devido ao alto desbalanceamento entre as classes. Mesmo que o *Grid Search* tenha elevado a *precision* $(0,607)$, o *F1-score* manteve-se limitado devido ao *recall* modesto. A progressão experimental partiu de uma MLP *baseline* e evoluiu com a introdução de um conjunto de validação, regularização por *early stopping* e otimização via *Grid Search*. Revelou-se que mesmo um classificador leve como uma MLP pode alcançar resultados notáveis quando opera sobre representações de alta qualidade.

O principal direcionamento futuro reside na expansão da classificação binária atual para uma abordagem multiclasse, visando diferenciar os tipos de queda (frontal, lateral, posterior), que é a lacuna motivadora deste estudo. Adicionalmente, a metodologia futura contemplará a diversidade das fontes de dados. Isso será realizado por meio da exploração de uma maior variedade de câmeras do *dataset* HQFS e da incorporação de novos conjuntos de dados externos,

como o *Up-Fall* e *UR-Fall*. Além disso, dado que o desbalanceamento de classes foi identificado como um possível obstáculo para o desempenho do classificador, pretende-se explorar técnicas de reamostragem de dados. Planeja-se a criação de um conjunto de dados balanceado utilizando métodos como o *oversampling*, com o objetivo de elevar substancialmente o *recall* e buscar um melhor *F1-score* sem comprometer a precisão.

REFERÊNCIAS

AREEB, S. et al. Fall Detection System using a single Accelerometer through Machine Learning. In: **2023 3rd International Conference on Digital Futures and Transformative Technologies (ICoDT2)**. Islamabad, Pakistan: IEEE, 2023. p. 1–7. ISBN 979-8-3503-3081-6.

BALDEWIJNS, G. et al. Bridging the gap between real-life data and simulated data by providing a highly realistic fall dataset for evaluating camera-based fall detection algorithms. **Healthcare Technology Letters**, v. 3, n. 1, p. 6–11, mar. 2016. ISSN 2053-3713, 2053-3713.

BEHRMANN, N. et al. Long Short View Feature Decomposition via Contrastive Video Representation Learning. In: **2021 IEEE/CVF International Conference on Computer Vision (ICCV)**. Montreal, QC, Canada: IEEE, 2021. p. 9224–9233. ISBN 978-1-6654-2812-5.

BURKART, N.; HUBER, M. F. A Survey on the Explainability of Supervised Machine Learning. **Journal of Artificial Intelligence Research**, v. 70, p. 245–317, jan. 2021. ISSN 1076-9757.

CHANDRA, M. A.; BEDI, S. S. Survey on SVM and their application in image classification. **International Journal of Information Technology**, v. 13, n. 5, p. 1–11, out. 2021. ISSN 2511-2104, 2511-2112.

CHOW, B.; REYES-ALDASORO, C. Automatic Gemstone Classification Using Computer Vision. **Minerals**, v. 12, n. 1, p. 60, dez. 2021. ISSN 2075-163X.

CHUI, K. T. et al. Facilitating innovation and knowledge transfer between homogeneous and heterogeneous datasets: Generic incremental transfer learning approach and multidisciplinary studies. **Journal of Innovation & Knowledge**, v. 8, n. 2, p. 100313, abr. 2023. ISSN 2444569X.

DR.A, U. R. Binary cross entropy with deep learning technique for Image classification. **International Journal of Advanced Trends in Computer Science and Engineering**, v. 9, n. 4, p. 5393–5397, ago. 2020. ISSN 22783091.

DUONG, H.-T.; LE, V.-T.; HOANG, V. T. Deep Learning-Based Anomaly Detection in Video Surveillance: A Survey. **Sensors**, v. 23, n. 11, p. 5024, maio 2023. ISSN 1424-8220.

ESPINOSA, R. et al. A vision-based approach for fall detection using multiple cameras and convolutional neural networks: A case study using the UP-Fall detection dataset. **Computers in Biology and Medicine**, v. 115, p. 103520, dez. 2019. ISSN 00104825.

ESPINOSA, R. et al. Application of Convolutional Neural Networks for Fall Detection Using Multiple Cameras. In: PONCE, H. et al. (Ed.). **Challenges and Trends in Multimodal Fall Detection for Healthcare**. Cham: Springer International Publishing, 2020. v. 273, p. 97–120. ISBN 978-3-030-38747-1 978-3-030-38748-8.

GALVÃO, Y. M. et al. A multimodal approach using deep learning for fall detection. **Expert Systems with Applications**, v. 168, p. 114226, abr. 2021. ISSN 09574174.

GAMMULLE, H. et al. Continuous Human Action Recognition for Human-Machine Interaction: A Review. **ACM Computing Surveys**, v. 55, n. 13s, p. 1–38, dez. 2023. ISSN 0360-0300, 1557-7341.

GAYA-MOREY, F. X.; MANRESA-YEE, C.; BUADES-RUBIO, J. M. Deep learning for computer vision based activity recognition and fall detection of the elderly: A systematic review. **Applied Intelligence**, v. 54, n. 19, p. 8982–9007, out. 2024. ISSN 0924-669X, 1573-7497.

GRUTSCHUS, T. et al. **Cutup and Detect: Human Fall Detection on Cutup Untrimmed Videos Using a Large Foundational Video Understanding Model**. [S.l.]: arXiv, 2024.

HA, T. V. et al. **Fall Detection Using Multimodal Data**. [S.l.]: arXiv, 2022.

HA, T. V. et al. Fall detection using mixtures of convolutional neural networks. **Multimedia Tools and Applications**, v. 83, n. 6, p. 18091–18118, jul. 2023. ISSN 1573-7721.

KHAN, A.; LAGHARI, A.; AWAN, S. Machine Learning in Computer Vision: A Review. **ICST Transactions on Scalable Information Systems**, p. 169418, jul. 2018. ISSN 2032-9407.

KULURKAR, P. et al. AI based elderly fall prediction system using wearable sensors: A smart home-care technology with IOT. **Measurement: Sensors**, v. 25, p. 100614, fev. 2023. ISSN 26659174.

KŮRKOVÁ, V.; SANGUINETI, M. Model complexities of shallow networks representing highly varying functions. v. 171, p. 598–604, 2016. ISSN 09252312. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0925231215009893>.

MAHADEVKAR, S. V. et al. A Review on Machine Learning Styles in Computer Vision—Techniques and Future Directions. **IEEE Access**, v. 10, p. 107293–107329, 2022. ISSN 2169-3536.

MARAY, N. et al. Transfer Learning on Small Datasets for Improved Fall Detection. **Sensors**, v. 23, n. 3, p. 1105, jan. 2023. ISSN 1424-8220.

MASWADI, K. et al. Human activity classification using Decision Tree and Naïve Bayes classifiers. **Multimedia Tools and Applications**, v. 80, n. 14, p. 21709–21726, jun. 2021. ISSN 1380-7501, 1573-7721.

MIGNONE, P.; PIO, G.; CECI, M. Distributed Heterogeneous Transfer Learning. **Big Data Research**, v. 37, p. 100456, ago. 2024. ISSN 22145796.

NAFEA, A. A. et al. A Short Review on Supervised Machine Learning and Deep Learning Techniques in Computer Vision. **Babylonian Journal of Machine Learning**, v. 2024, p. 48–55, fev. 2024. ISSN 3006-5429.

NASTESKI, V. An overview of the supervised machine learning methods. **HORIZONS.B**, v. 4, p. 51–62, dez. 2017. ISSN 18578578, 18579892.

NÚÑEZ-MARCOS, A.; ARGANDA-CARRERAS, I. Transformer-based fall detection in videos. **Engineering Applications of Artificial Intelligence**, v. 132, p. 107937, jun. 2024. ISSN 09521976.

OMS. **World Health Organization fact sheets: Falls**. 2021. <https://www.who.int/news-room/fact-sheets/detail/falls>.

PANERU, S.; JEELANI, I. Computer vision applications in construction: Current state, opportunities & challenges. **Automation in Construction**, v. 132, p. 103940, dez. 2021. ISSN 09265805.

PEREIRA, S. S. L.; MAIA, J. **I3D Video Features, Labels and Splits for Multicamera Overlapping Datasets Pets-2009, HQFS and Up-Fall**. Zenodo, 2025. Disponível em: <https://doi.org/10.5281/zenodo.14655606>.

SATHYANARAYANAN, S. Confusion Matrix-Based Performance Evaluation Metrics. p. 4023–4031, 2024. Disponível em: <https://africanjournalofbiomedicalresearch.com/index.php/AJBR/article/view/4345/3327>.

SUN, H. et al. An Improved Medical Image Classification Algorithm Based on Adam Optimizer. **Mathematics**, v. 12, n. 16, p. 2509, ago. 2024. ISSN 2227-7390.

SUN, Z. et al. Human Action Recognition from Various Data Modalities: A Review. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, p. 1–20, 2022. ISSN 0162-8828, 2160-9292, 1939-3539.

TERVEN, J.; CÓRRKOV-ESPARZA, D.-M.; ROMERO-GONZÁLEZ, J.-A. A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. **Machine Learning and Knowledge Extraction**, v. 5, n. 4, p. 1680–1716, nov. 2023. ISSN 2504-4990.

TONG, Z. et al. **VideoMAE: Masked Autoencoders are Data-Efficient Learners for Self-Supervised Video Pre-Training**. arXiv, 2022. Disponível em: <http://arxiv.org/abs/2203.12602>.

USMANI, S. et al. Latest Research Trends in Fall Detection and Prevention Using Machine Learning: A Systematic Review. **Sensors**, v. 21, n. 15, p. 5134, jul. 2021. ISSN 1424-8220.

WANG, L. et al. VideoMAE V2: Scaling Video Masked Autoencoders with Dual Masking. In: **2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. Vancouver, BC, Canada: IEEE, 2023. p. 14549–14560. ISBN 979-8-3503-0129-8.

WANG, W. et al. Understanding and Modeling of WiFi Signal Based Human Activity Recognition. In: **Proceedings of the 21st Annual International Conference on Mobile Computing and Networking**. Paris France: ACM, 2015. p. 65–76. ISBN 978-1-4503-3619-2.

XIAO, B.; YIN, X.; KANG, S.-C. Vision-based method of automatically detecting construction video highlights by integrating machine tracking and CNN feature extraction. **Automation in Construction**, v. 129, p. 103817, set. 2021. ISSN 09265805.

XU, S. et al. Computer Vision Techniques in Construction: A Critical Review. **Archives of Computational Methods in Engineering**, v. 28, n. 5, p. 3383–3397, ago. 2021. ISSN 1134-3060, 1886-1784.

XU, X. et al. Crack Detection and Comparison Study Based on Faster R-CNN and Mask R-CNN. **Sensors**, v. 22, n. 3, p. 1215, fev. 2022. ISSN 1424-8220.

ZURBUCHEN, N.; WILDE, A.; BRUEGGER, P. A Machine Learning Multi-Class Approach for Fall Detection Systems Based on Wearable Sensors with a Study on Sampling Rates Selection. **Sensors**, MDPI AG, v. 21, n. 3, p. 938, jan. 2021. ISSN 1424-8220.