UNIVERSIDADE FEDERAL DE JUIZ DE FORA

INSTITUTO DE CIÊNCIAS EXATAS

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

André de Souza Brito

Uma arquitetura de redes neurais convolucionais para reconhecimento de ações humanas em vídeos utilizando ritmos de fluxo óptico

Juiz de Fora

2020

André de Souza Brito

Uma arquitetura de redes neurais convolucionais para reconhecimento de ações humanas em vídeos utilizando ritmos de fluxo óptico

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Orientador: Saulo Moraes Villela

Coorientador: Marcelo Bernardes Vieira

Juiz de Fora

2020

Ficha catalográfica elaborada através do Modelo Latex do CDC da UFJF
com os dados fornecidos pelo(a) autor(a)

Brito, André de Souza.
Uma arquitetura de redes neurais convolucionais para reconhecimento de ações humanas em vídeos utilizando ritmos de fluxo óptico / André de Souza Brito. – 2020.
94 f. : il.

Orientador: Saulo Moraes Villela
Coorientador: Marcelo Bernardes Vieira
Dissertação (Mestrado) – Universidade Federal de Juiz de Fora, Instituto de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação, 2020.

1. Reconhecimento de Ações Humanas. 2. Multi-fluxo. 3. Ritmo do Fluxo Óptico. I. Villela, Saulo Moraes, orient. II. Vieira, Marcelo Bernardes, coorient. III. Título.

André de Souza Brito

Uma arquitetura de redes neurais convolucionais para reconhecimento de ações humanas em vídeos utilizando ritmos de fluxo óptico

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do grau de Mestre em Ciência da Computação.

Aprovada em 30 de agosto de 2019.

BANCA EXAMINADORA

Prof. D.Sc. Saulo Moraes Villela - Orientador
Universidade Federal de Juiz de Fora

Prof. D.Sc. Marcelo Bernardes Vieira - Coorientador
Universidade Federal de Juiz de Fora

Prof. D.Sc. Heder Soares Bernardino
Universidade Federal de Juiz de Fora

Prof. Ph.D. Hélio Pedrini
Universidade Estadual de Campinas

AGRADECIMENTOS

Aos meus pais pelo apoio incondicional e pelas palavras de incentivo nas horas mais difíceis. Aos meus avós pelos ensinamentos passados, que impactaram muito no meu crescimento pessoal e a todos os meus parentes, pelo encorajamento e apoio.

Ao professor Saulo e ao professor Marcelo pela orientação, amizade e principalmente, pela paciência, sem a qual este trabalho não se realizaria. Aos professores do Departamento de Ciência da Computação e do PGCC pelos seus ensinamentos e aos funcionários do curso que, durante esses anos, contribuíram de algum modo para o meu enriquecimento pessoal e profissional.

Aos meus amigos do mestrado, pelo apoio e amizade ofertados durante o período em que convivemos. Gostaria de agradecer a todas as pessoas do GCG, em especial, meus amigos Hemerson e Hugo, que acompanharam todo o meu processo de aprendizado e dividiram comigo todas as conquistas e frustrações na grande tarefa que foi aprender *Deep Learning*. Agradeço também o pessoal do LIV, em especial à Helena, pela troca de conhecimentos e por todo o suporte fornecido durante o meu mestrado, sem isso dificilmente eu teria chegado até esse ponto.

Agradeço à CAPES pelo suporte financeiro sem o qual eu não teria conseguido realizar este mestrado. Agradeço também à NVIDIA pela doação das GPUs que possibilitaram a realização dos "infinitos" experimentos realizados a fim de materializar este trabalho.

"The most exciting phrase to hear in science, the one that heralds new discoveries, is not Eureka! (I've found it!), but That's funny..."(Isaac Asimov).

RESUMO

O problema de reconhecimento de ações humanas baseada em vídeos beneficiou-se significativamente do surgimento de modelos de aprendizado profundo. No entanto, este ainda é um problema em aberto devido à dificuldade associada ao processo de desenvolvimento de uma solução robusta e geral neste domínio. Neste ambiente, abordar o aspecto temporal dos vídeos é crucial para construir modelos realistas. Um passo fundamental nessa direção é a seleção de características dos vídeos que retratem a complexidade das ações humanas. Com este objetivo, uma solução baseada em uma arquitetura multi-fluxo formada por redes neurais convolucionais profundas é proposta neste trabalho. Um esquema multi-fluxo é uma forma interessante de agregar informações de diferentes origens com um custo de treinamento inferior ao de outros métodos. Uma nova característica temporal, chamada ritmo do fluxo óptico, foi incorporada à arquitetura para melhorar o seu desempenho. Os experimentos realizados sugerem que o ritmo do fluxo óptico é complementar às outras informações geralmente usadas nessas arquiteturas, como imagens em RGB, fluxo óptico e ritmo visual, potencializando os resultados da abordagem. Para combinar os vários fluxos de informação dessa arquitetura, é introduzido um novo método de fusão por média ponderada, onde os pesos dos classificadores são definidos pela meta-heurística resfriamento simulado. Além disso, é especificada uma nova estratégia de treinamento e teste para os dois novos fluxos introduzidos neste trabalho. Esse esquema é baseado na extração de múltiplos planos dos ritmos visuais e do fluxo óptico dos vídeos. Os resultados indicam um aumento na eficácia da arquitetura usando esta estratégia. Os experimentos realizados em dois conjuntos de dados desafiadores, UCF101 e HMDB51, demonstram que o método desenvolvido é comparável às abordagens estado da arte.

Palavras-chave: Reconhecimento de Ações Humanas. Multi-fluxo. Ritmo do Fluxo Óptico.

ABSTRACT

The human action recognition problem based on video classification has significantly benefited from the introduction of deep learning models. However, this is still an open problem due to the inherent difficulty in developing a general and robust solution. In this domain, addressing the temporal aspect of the videos is crucial in order to build realistic models. A key step in this direction is the selection of features that characterize the complexity of human actions. With this goal, we propose a solution for the video classification problem based on a multi-stream deep convolutional neural network architecture. A multi-stream is an interesting way to aggregate information from different sources with a lower training cost if compared to other methods. A new stream called optical flow rhythm was incorporated to improve accuracy rates. Our experiments suggest that optical flow rhythm is complementary to other streams, such as RGB, optical flow and visual rhythm, boosting the results of the approach. To combine the various streams in our architecture, we introduce a new weighted average fusion scheme where the weights of the classifiers are defined by a simulated annealing metaheuristic. Furthermore, we propose a training and test protocol based on the extraction of multiple planes of the visual and optical flow rhythms. The results indicate a performance augmentation using this protocol. Experiments conducted on the challenging UCF101 and HMDB51 datasets demonstrate that our method is comparable to state-of-the-art approaches.

Keywords: Human Action Recognition. Multi-stream. Optical Flow Rhythm.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AVR	*Adaptive Visual Rhythm*
CNN	*Convolutional Neural Network*
CV	*Computer Vision*
DCNN	*Deep Convolutional Neural Network*
DL	*Deep Learning*
DNN	*Deep Neural Network*
EEP	*Eigen Evolution Pooling*
HAR	*Human Action Recognition*
IDT	*Improved Dense Trajectories*
LSTM	*Long Short-Term Memory*
ML	*Machine Learning*
MOFR	*Multiple Optical Flow Rhythms*
MVR	*Multiple Visual Rhythms*
NN	*Neural Network*
OF	*Optical Flow*
OFF	*Optical Flow guided Feature*
OFR	*Optical Flow Rhythm*
PoTion	*Pose moTion*
PSO	*Particle Swarm Optimization*
ReLU	*Rectified Linear Units*
RGB	*Red Green Blue*
RNN	*Recurrent Neural Network*
SA	*Simulated Annealing*
SGD	*Stochastic Gradient Descent*
TSN	*Temporal Segment Networks*

SOTA	*State Of The Art*
SVM	*Support Vector Machine*
VR	*Visual Rhythm*
WVR	*Weighted Visual Rhythm*

LISTA DE SÍMBOLOS

$\vec{d_\tau}(u, v)$	campo de deslocamentos de fluxo óptico no ponto (u, v) do *frame* τ
d_τ^x	componente horizontal do campo de deslocamentos de fluxo óptico
d_τ^y	componente vertical do campo de deslocamentos de fluxo óptico
h	altura
w	largura
I_τ	pilha de fluxos ópticos no *frame* τ
F_i	*frame* i formado pela matriz $h \times w$.
p_i	ponto $p = (u, v)$ no *frame* i
P	um conjunto de coordenadas 2D $P = \{p_1, \cdots, p_n\}$ nas imagens
V	video $V = \{F_1, F_2, \cdots, F_f\}$
A_i	$A_i = [F_i(p_1)\ F_i(p_2)\ \cdots\ F_i(p_n)]^T$, sendo A_i a contribuição 1D do i-ésimo *frame* com $F_i(p_j)$ representando o valor RGB do ponto p_j no *frame* F_i.
VR_P	ritmo visual $\mathrm{VR}_P = [A_1\ A_2\ \cdots\ A_f]$ dado um conjunto P de coordenadas 2D
O	volume de fluxo óptico $O = \{D_1, D_2, \ldots, D_f\}$
$D_i(p_j)$	vetor 2D de deslocamentos de fluxo óptico no ponto p_j do i-ésimo *frame*
C_i	$C_i = [D_i(p_1)\ D_i(p_2)\ \cdots\ D_i(p_n)]^T$, sendo C_i um vetor coluna $n \times 2$ com a contribuição 1D do i-ésimo *frame*
OFR_P	ritmo do fluxo óptico $\mathrm{OFR}_P = [C_1\ C_2\ \cdots\ C_f]$ dado um conjunto P de coordenadas 2D.
OFR_P^x	componente horizontal do ritmo do fluxo óptico OFR_P
OFR_P^y	componente vertical do ritmo do fluxo óptico OFR_P
acc	valor de acurácia
reg	termo referente ao fator de regularização agregado à função objetivo do SA
β	constante de regularização

a_i	peso associado à i-ésima *stream*, onde $0 \leq a_i \leq 1$ e $\sum_{i=0}^{n} a_i = 1$ em cada linha da matriz de pesos, ou no vetor de pesos inteiros, usando uma arquitetura de n *streams*
$a_{i,j}$	peso associado à i-ésima classe da j-ésima *stream* de uma arquitetura

SUMÁRIO

1 INTRODUÇÃO

Técnicas de aprendizado profundo (*deep learning* – DL) vêm alcançando resultados expressivos em vários campos de pesquisas. Esses resultados foram impulsionados, principalmente, por avanços técnicos e tecnológicos que tornaram possível o processamento de bases de dados mais densas. Unidades de processamento gráfico mais potentes, o aumento considerável da quantidade de dados digitais produzidos e avanços técnicos no campo de aprendizado de máquina (*machine learning* – ML), são alguns destes fatores que contribuíram para difundir o uso de arquiteturas profundas (POUYANFAR et al., 2018).

Visão computacional (*computer vision* – CV), por exemplo, foi uma das áreas que se beneficiou da introdução destes modelos. Vários tópicos dessa área, como classificação de imagens (KRIZHEVSKY; SUTSKEVER; HINTON, 2012; SIMONYAN; ZISSERMAN, 2014b; SZEGEDY et al., 2016; MAHAJAN et al., 2018), reconhecimento facial (WANG; DENG, 2018), e colorização de imagens (ZHANG; ISOLA; EFROS, 2016), atualmente são abordados quase que exclusivamente por meio de modelos profundos. Dado o sucesso substancial alcançado nestes tópicos, era natural pressupor que parte destas conquistas pudessem ser transpostas e reproduzidas em campos correlatos como, por exemplo, o de classificação de vídeos. Dentro desse domínio, especificamente, há um problema conhecido como reconhecimento de ações humanas (*human action recognition* – HAR) que passou por uma verdadeira renovação com a emergência de técnicas de DL. Esse problema consiste em classificar ações executadas por um humano em um vídeo. Muitos estudos têm esse tópico como foco principal e suas potenciais aplicações que incluem sistemas de vigilância, interfaces humano-máquina e recuperação de vídeos (KONG; FU, 2018).

Apesar dos avanços decorrentes dos últimos anos, o HAR é um tema que permanece desafiador. Os principais entraves associados a este domínio têm relação com o tipo de dado analisado. Vídeos são estruturas de dados heterogêneas extremamente complexas. A alta dimensionalidade, redundância de informação, e problemas associados à própria natureza desses dados, como oclusões, movimentos de câmera, variações bruscas de iluminação e *background*, são alguns dos aspectos que justificam a utilização de modelos de ML mais robustos para abordar o problema. Outra dificuldade reside na complexidade inerente ao processo de modelagem da estrutura temporal dos movimentos.

Buscando mitigar esses problemas, Simonyan e Zisserman (2014a) introduziram o conceito de um modelo de DL baseado em múltiplos fluxos (canais) de informação denominado arquitetura *two-stream*. Os autores apresentam uma nova maneira de abordar o HAR. A ideia é bem simples: substituir um processo de aprendizado geral focado em abordar vários aspectos dos vídeos ao mesmo tempo, por um processo mais fragmentado, baseado em fluxos paralelos de aprendizado, onde cada aspecto da informação é analisado em separado por um modelo especializado. A arquitetura proposta originalmente pelos autores

contava com dois desses fluxos (*streams*): uma rede neural convolucional profunda (*deep convolutional neural network* – DCNN) "espacial", focada em extrair padrões relacionados à aparência dos vídeos como objetos, texturas, *background*, entre outros, e uma segunda DCNN responsável por aprender uma assinatura temporal presente nos movimentos realizados em determinadas ações. Os conhecimentos desses vários fluxos são mesclados, em um passo posterior, por meio de um mecanismo denominado fusão. Essa forma de trabalhar o problema tem muitos pontos em comum com uma outra técnica de ML chamada *ensemble learning* (ROKACH, 2010). De modo que grande parte dos desafios decorrentes da utilização desta abordagem tem relação com desafios também encontrados em métodos baseados em *ensembles*. Fatores como diversidade entre os componentes e formas eficientes de combinar as informações são algumas das premissas encontradas em ambos os domínios. Apesar de vários trabalhos posteriores proporem soluções visando esses pontos como, por exemplo, formas de representar os diversos aspectos do vídeo com um alto poder discriminativo e complementaridade (KONG; FU, 2018), várias lacunas ainda precisam ser preenchidas. Uma dessas questões em aberto tem relação com uma particularidade que determinadas ações apresentam. Essas ações só fazem sentido se observadas por meio de uma perspectiva global. De forma que, para identificá-las corretamente, os métodos empregados devem ser capazes de extrair relações nos vídeos denominadas relações temporais de longo alcance que exprimem esta propriedade.

Um outro fator crucial que aparentemente parece ter sido negligenciado nos trabalhos mais recentes é a escolha mais criteriosa de um método de fusão adequado à suas arquiteturas. Conforme levantado por Wu et al. (2015), as escolhas dos autores geralmente se resumem a métodos extremamente simples, já provados ineficientes, ou a métodos exaustivos. Esse fato justifica o estudo de técnicas mais eficientes para fundir as informações, levando em consideração fatores como, por exemplo, a complementaridade entre os fluxos e os diferentes níveis de contribuições de cada modelo no entendimento de uma ação.

1.1 DEFINIÇÃO DO PROBLEMA

O problema de reconhecimento de ações humanas baseado em vídeos consiste em identificar e classificar a ação principal realizada por humanos em uma cena. Conforme mencionado, os grandes desafios atribuídos a este tópico residem na complexas dinâmicas associadas a ações humanas. Por exemplo, algumas ações são definidas exclusivamente pela movimentação dos atores, outras são marcadas por interações com objetos ou outras pessoas, e não é raro encontrar ações em que todos esses elementos contribuem para o processo de reconhecimento. Nesse sentido, costuma-se dizer que essas ações mais complexas, definidas normalmente como atividades, são um dos maiores desafios a serem enfrentados dentro deste domínio.

Nesta dissertação, o HAR é abordado via aprendizado supervisionado. O que significa dizer que são utilizadas bases de dados com vídeos anotados, ou seja, para cada vídeo nessas bases sabe-se com exatidão qual ação está sendo realizada. Esses vídeos de treinamento servirão como base de aprendizado para um modelo, que, posteriormente, será utilizado para reconhecer ações em vídeos novos não vistos, ou seja, vídeos para os quais não existem anotações. Neste contexto, o grande desafio é construir um modelo capaz de reconhecer adequadamente as ações nestes vídeos desconhecidos, utilizando somente o conhecimento abstraído do conjunto de vídeos utilizado no processo de treinamento.

1.2 OBJETIVOS

O principal objetivo deste trabalho é propor uma arquitetura multi-fluxo (*multi-stream*) aplicada ao reconhecimento de ações humanas em vídeos. Várias modificações são propostas para alcançar esse objetivo, focando em pontos considerados frágeis em arquiteturas desse tipo utilizadas na literatura. Entre as possíveis melhorias, podem ser citadas, por exemplo: novas *streams* de informação visando diversidade de conhecimento, um mecanismo de fusão capaz de combinar satisfatoriamente o conhecimento extraído por cada *stream* e um protocolo geral de utilização (treinamento e teste) bem conciso, objetivando transparência e reprodutibilidade aos experimentos realizados. Como objetivos específicos do presente trabalho, podem ser mencionados:

- Mensurar o impacto das alterações propostas à arquitetura multi-fluxo;
- Discorrer sobre características inerentes ao problema que possam facilitar ou direcionar melhor o processo de classificação dos vídeos;
- Analisar o comportamento da meta-heurística resfriamento simulado (*simulated annealing* – SA) como cerne do método de fusão desenvolvido.

1.3 CONTRIBUIÇÕES

As principais contribuições deste trabalho são:

- O ritmo do fluxo óptico (*optical flow rhythm* – OFR) como representação (*feature*) temporal para classificação de vídeo;
- Os múltiplos ritmos visuais (*multiple visual rhythms* – MVR) e os múltiplos ritmos do fluxo óptico (*multiple optical flow rhythms* – MOFR), respectivamente, como fluxo espaço-temporal e temporal em arquiteturas multi-fluxo para reconhecimento de ações humanas;
- Uma nova estratégia para treinar e testar os ritmos visuais e do fluxo óptico;

- Um novo método de fusão baseado na utilização da meta-heurística SA;
- Uma arquitetura 4-*stream* para o HAR, com resultados comparáveis aos métodos estado da arte, fundamentada em diversas descobertas provenientes deste trabalho.

1.4 ORGANIZAÇÃO

O presente trabalho está estruturado da seguinte forma: o Capítulo 2 apresenta o HAR de forma mais detalhada, além de uma visão geral dos trabalhos presentes na literatura que se relacionam de alguma forma com este trabalho. Os conceitos fundamentais que compõem o método proposto são explicados no Capítulo 3. O Capítulo 4 descreve o método proposto neste trabalho. Os experimentos realizados com o intuito de avaliar a abordagem desenvolvida, acompanhado de uma descrição dos detalhes de implementação são apresentados no Capítulo 5. O Capítulo 6 resume os resultados e contribuições deste trabalho, além de introduzir possíveis melhorias a serem abordadas em trabalhos futuros.

2 RECONHECIMENTO DE AÇÕES HUMANAS

O termo ação humana estudado em CV pode se referir a simples movimentos de partes do corpo ou a interações complexas entre humanos e objetos. Dessa forma, é extremamente difícil encontrar uma única definição formal que sintetize esse termo. Turaga et al. (2008), por exemplo, definem uma ação humana como um conjunto de padrões de movimento usualmente executados por uma única pessoa com duração média de poucos segundos. Estendendo essa definição, Weinland, Ronfard e Boyer (2011) afirmam que uma ação é essencialmente uma série de movimentos gerados por um ator durante a execução de uma tarefa, que pode ser uma interação com outros humanos ou objetos. Para Chaquet, Carmona e Fernández-Caballero (2013), uma ação pode ser considerada uma sequência de micro-ações, atômicas, que cumprem uma função ou um propósito.

Contrário às particularidades relacionadas a cada uma dessas definições, um ponto em comum é descrito por todos os autores. Todas as definições demonstram que existe uma hierarquia de conceitos atrelada a ideia de ação. De fato, Moeslund, Hilton e Krüger (2006) já haviam estabelecido uma terminologia para categorizar ações de acordo com sua complexidade. Com base nesses conceitos, os movimentos humanos são agrupadas em gestos, ações e atividades. Gestos, elementos atômicos, correspondem a movimentos simples de partes do corpo como "levantar uma mão" e "esticar uma perna". Uma ação, nesta hierarquia, equivale a um movimento corporal mais complexo que consiste de vários gestos organizados temporalmente, "correr" e "secar o cabelo" são exemplos de ações. Atividades, por outro lado, são movimentos corporais ainda mais complexos compostos por sequências de ações. Atividades usualmente são relacionadas à interações com diversas pessoas ou objetos, eventos, como uma partida de futebol ou um grupo de pessoas dançando. No entanto, é importante notar que nem sempre há uma distinção clara entre esses níveis hierárquicos. Uma pessoa correndo de um lugar para outro pode ser interpretado como uma simples ação, ou como uma atividade se for levado em consideração, por exemplo, que ela possa estar fugindo de uma possível situação de risco. É por isso que as bases de dados (*datasets*) descritas neste trabalho não fazem distinção entre ações e atividades. De uma maneira genérica, diz-se que elas armazenam ações.

Essa dificuldade de fornecer uma definição formal precisa para conceitos básicos associados ao problema já dá uma dimensão dos desafios a serem enfrentados dentro desse domínio. Conforme mencionado, é extremamente complexo o processo de identificar uma ação a partir de um vídeo. Uma série de fatores podem interferir nesta tarefa e é comum a utilização de algum tipo de trabalho manual e conhecimento prévio do domínio para facilitar o processo de reconhecimento. Por exemplo, é possível encontrar ações que podem ser potencialmente executadas de diversas formas diferentes por uma mesma pessoa. É usual também encontrar em alguns *datasets*, classes com sentido muito amplo, muitas vezes ambíguo, como "abrir" que podem descrever várias ações distintas (KONG; FU, 2018).

Para superar esses e outros problemas já descritos no Capítulo 1, várias abordagens pré-processam os vídeos para construir representações para as ações e, em seguida, usar as mesmas como entrada em suas arquiteturas (KONG; FU, 2018). Por esse ponto de vista, o HAR pode ser reduzido a um problema de encontrar uma representação efetiva e inferir a ação realizada através dela (classificação). É uma prática comum usar, ao mesmo tempo, várias dessas representações em abordagens HAR. Geralmente, a arquitetura empregada recombina esses elementos em algum momento para criar um único descritor do vídeo, um único vetor numérico multidimensional que descreve todo o vídeo, na tentativa de melhorar o processo de reconhecimento. Partindo desse descritor, um modelo de classificação como uma rede neural (*neural network* – NN), uma rede neural convolucional (*convolutional neural network* – CNN) ou uma máquina de vetores suporte (*support vector machine* – SVM), infere uma "etiqueta" que descreve a ação realizada.

2.1 REPRESENTAÇÕES

Conforme já apresentado por Kong e Fu (2018), o primeiro e mais importante problema no HAR é o de como representar uma ação em um vídeo. De fato, parte significante do sucesso de um método está na seleção adequada de seus dados de entrada. Encontrar *features* extraídas dos vídeos que sejam, ao mesmo tempo, discriminantes e gerais, ou seja, capazes de caracterizar as ações e, simultaneamente, maximizar a discrepância entre classes diferentes, diminuindo com isso o erro de classificação. Existe uma vasta literatura relacionada a esse tema (DALAL; TRIGGS, 2005; SCOVANNER; ALI; SHAH, 2007; ZACH; POCK; BISCHOF, 2007; TORRES; PEDRINI, 2016) e algumas representações interessantes foram derivadas desses estudos como o ritmo visual e o fluxo óptico (*optical flow* – OF).

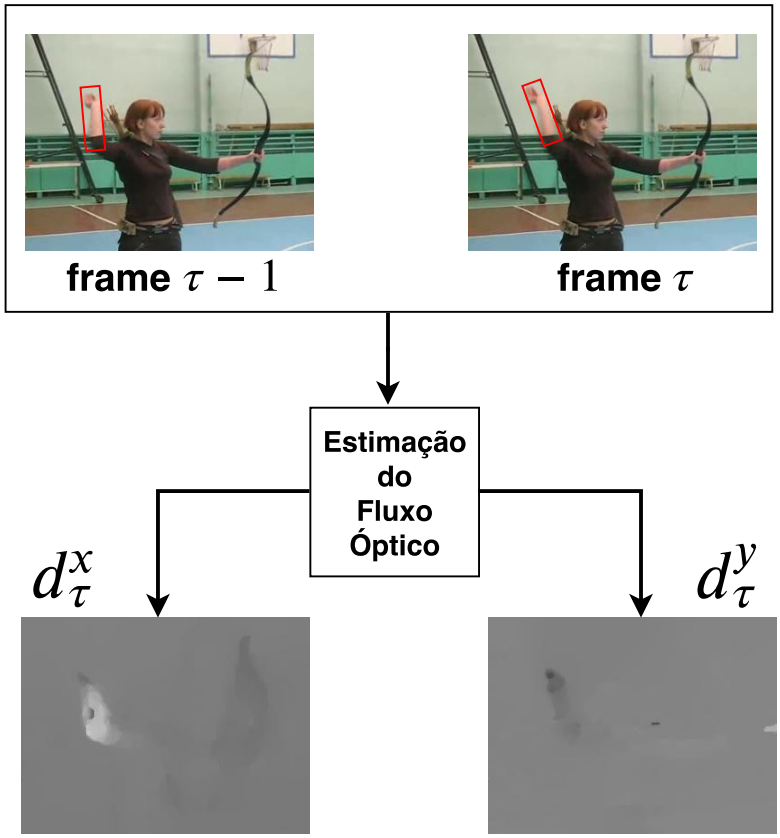
Essas representações usualmente abordam um ou mais aspectos do vídeo. O OF, por exemplo, é uma das representações que abordam o aspecto temporal dos movimentos. O VR, por outro lado, é uma *feature* que busca retratar aspectos espaço-temporais das ações. As seções a seguir definem brevemente as representações utilizadas neste trabalho, OF e VR, destacando as suas principais características no ambiente de classificação de vídeos.

2.1.1 **Fluxo óptico**

Horn e Schunck (1981) definem o fluxo óptico como a distribuição das velocidades aparentes dos padrões de brilho em uma imagem. Isto é, essencialmente, o OF é um método para caracterizar e quantificar o movimento de objetos em um vídeo. Mais precisamente, é um campo vetorial de deslocamento 2D onde cada vetor $\vec{d_t}(u,v) = [d_t^x(u,v) \;\; d_t^y(u,v)]^T$ indica o movimento do respectivo ponto $p = (u,v)$ entre dois quadros consecutivos t e $t+1$. A Figura 1, por exemplo, ilustra alguns dos padrões de movimento deixados nas

componentes verticais e horizontais do OF pelo deslocamento de pontos relativos à mão do ator (delimitados pela área em vermelho). O OF tem várias aplicações na área de CV, como detecção de objetos (ASLANI; MAHDAVI-NASAB, 2013), estabilização de vídeo (LIU et al., 2014) e segmentação de imagens (SEVILLA-LARA et al., 2016).

Figura 1 – O OF estima a variação de movimento em ambas as direções, X e Y, entre dois quadros (*frames*) consecutivos.



Fonte: elaborado pelo autor.

No escopo de classificação de vídeos, existem fortes evidências de que esta estimativa de movimento, gerada a partir do OF, é uma informação valiosa para caracterizar determinadas ações. Johansson (1973), por exemplo, demonstrou que várias ações podem ser reconhecidas por observadores humanos usando somente o movimento descrito por poucos pontos específicos nas articulações dos atores. Em outro trabalho, Jhuang et al. (2013) mostraram que o OF, além de capturar a movimentação dos objetos, fornece uma outra informação útil no processo de reconhecimento de ações, que seriam as assinaturas de movimento geradas perto da silhueta de objetos se deslocando. Esse fato sugere que o OF pode também ser útil na extração desses contornos que possivelmente identifiquem algumas ações.

Mais recentemente, Sevilla-Lara et al. (2017) argumentaram que o OF é útil nesse domínio por ser uma característica extraída dos vídeos invariante à aparência, mais do que por ser um descritor de movimento. Para sustentar essa afirmação, os autores

realizaram alguns experimentos com vetores de fluxo inacurados e verificaram que, mesmo sob essas circunstâncias, o desempenho dos modelos não era severamente prejudicado. Em um trabalho ainda mais recente, Sun et al. (2017) propuseram uma nova forma de representar compactamente o movimento nos vídeos inspirado nos conceitos do OF. Essa nova representação guiada pelo fluxo óptico (*optical flow guided feature* – OFF), proposta pelos autores, aplica os conceitos de OF na diferenciação de informações extraídas de camadas de uma rede neural obtidas durante o processamento de quadros (*frames*) consecutivos de um vídeo. Essa representação derivada do OF, pode então ser repassada a um segundo modelo responsável, por exemplo, por definir ações em vídeos com base nessa representação compacta de movimento.

Estes resultados sugerem que o OF pode desempenhar um papel importante no problema de reconhecimento de ações. A literatura relacionada a esse tema é tão ampla que aborda desde métodos baseados exclusivamente em *features* extraídas previamente dos dados por especialistas de domínio (*handcrafted features*) (FATHI; MORI, 2008; GUO; ISHWAR; KONRAD, 2010; WANG et al., 2011; WANG; SCHMID, 2013) até arquiteturas mais recentes baseadas em técnicas de DL (SIMONYAN; ZISSERMAN, 2014a; CARREIRA; ZISSERMAN, 2017; SEVILLA-LARA et al., 2017; FAN et al., 2018a; CONCHA et al., 2018).

2.1.2 **Ritmo visual**

O termo ritmo visual foi introduzido no contexto de representação de vídeo por Kim et al. (2001). Segundo os autores, o ritmo visual é um tipo de abstração bidimensional gerada de um vídeo, por meio de um processo predeterminado e sistemático empregado em seu conjunto de *frames*. O resultado desse processo é uma superfície 2D arbitrária imersa no volume 3D de um vídeo. Em suma, é uma imagem única que representa o conteúdo completo do vídeo. Ou, ainda mais importante, uma imagem que pode conter padrões visuais relevantes para vários problemas de CV.

Kim et al. (2001), por exemplo, propuseram o uso dessa representação 2D para identificar efeitos em vídeos como cortes, *fades*, movimentos de câmera, dentre outros. A premissa adotada pelos autores é a de que diferentes efeitos nos vídeo implicam em diferentes padrões nas imagens geradas. Com base em descobertas semelhantes, Valio, Pedrini e Leite (2011) desenvolveram um método baseado em VR para detectar legendas em filmes. O padrão retangular bem definido produzido pelas legendas facilita o seu processo de segmentação. Em um trabalho mais recente, Torres e Pedrini (2016) empregaram o VR no problema de reconhecimento de ações humanas. Os autores usaram filtros passa-alta para obter regiões de interesse nos VR dos vídeos, sob a justificativa de que os padrões referentes às ações estão concentradas nessas regiões específicas.

Levantadas as possíveis aplicações, faz-se necessária uma definição mais formal da

ideia de VR. No âmbito do presente trabalho, foi utilizada, com mínimas adaptações, a formulação apresentada por Concha et al. (2018). Nesse trabalho, o ritmo visual VR_P de um vídeo $V = \{F_1, F_2, \ldots, F_L\}$ com L *frames* F_i, tendo $h \times w$ pontos, é formalmente dado pela matriz $n \times L$:

$$VR_P = [A_1 \ A_2 \ \cdots \ A_L], \tag{2.1}$$

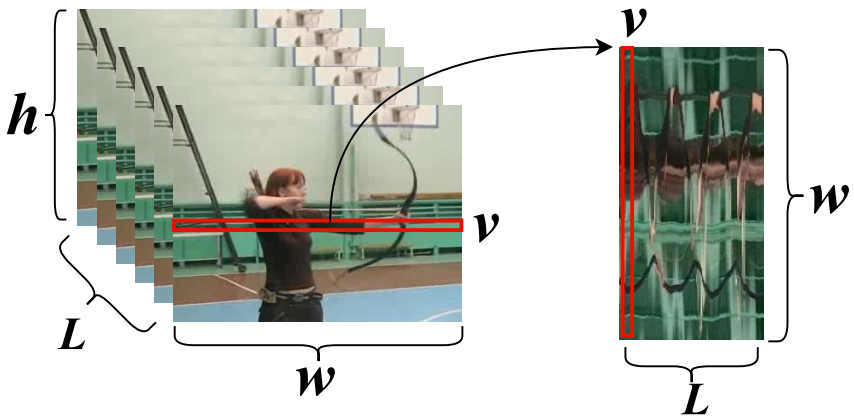
onde $A_i = [F_i(p_1) \ F_i(p_2) \ \ldots \ F_i(p_n)]^T$, com $F_i(p_j)$ representando o valor RGB do *pixel* em p_j do *frame* F_i, e

$$P = \{p_1, \ldots, p_n\}, \tag{2.2}$$

é um conjunto de coordenadas 2D da imagem (u, v), com $u \in \{1, \ldots, w\}$ e $v \in \{1, \ldots, h\}$. Os pontos em P geralmente formam um trajetória compacta, fazendo com que o VR correspondente represente um subespaço 2D imerso no volume UVT do vídeo. Assim, se P é um conjunto de pontos de uma única linha de cada *frame* ou uma única coluna, o VR resultante, horizontal ou vertical, é um plano paralelo, respectivamente, a UT e VT.

Contrariando o que foi proposto por Concha et al. (2018) e Souza (2018), onde são usados VR horizontais e verticais "médios", no desenvolvimento deste trabalho foram utilizados VR horizontais baseados na formulação clássica descrita pela Equação (2.1), e representados pela Figura 2. Optou-se por VR horizontais por eles apresentarem resultados superiores de acurácia quando comparados aos verticais, conforme observado por Concha et al. (2018) em seus experimentos.

Figura 2 – Ritmo visual horizontal: v é a linha de referência.



Fonte: elaborado pelo autor.

2.2 CLASSIFICAÇÃO DE AÇÕES

Uma vez que as sequências de vídeo estão representadas, é necessário decidir quais ações elas contêm. Ou, mais precisamente, classificar quais ações estão sendo descritas por uma das representações mencionadas anteriormente. É nesse sentido que entram os

algoritmos de ML. Esses métodos aprendem aspectos relevantes, de um conjunto de dados fornecido, que permitem que os mesmos possam extrapolar esse conhecimento adquirido de dados nunca vistos anteriormente. É esse processo de aprendizado, ou simplesmente, treinamento, que faz com que os algoritmos de ML usados em arquiteturas HAR possam inferir ações em vídeos desconhecidos. Quando se conhece as ações dos vídeos usados no treinamento, diz-se que esse processo de aprendizado é supervisionado. Neste contexto, um modelo HAR aprende a "classificar" as ações identificando, principalmente, padrões que ajudem a caracterizar esses elementos nos vários vídeos usados de exemplo. Padrões esses que podem ter relação com assinaturas temporais e/ou espaciais encontradas nos vídeos utilizados.

Os primeiros algoritmos de classificação utilizados em arquiteturas HAR geralmente recebiam algumas das *hand-crafted features*, descritas na Seção 2.1, pré-processadas e transformadas em um descritor. Esse pré-processamento geralmente era feito em diversas etapas e podia incluir a aplicação de diferentes técnicas como a extração de regiões de interesse, filtragens, entre outros (KONG; FU, 2018). SVM, NN e árvores de decisão (*decision trees*) (SHARMA; KUMAR, 2016) eram alguns dos classificadores mais utilizados na literatura desse período (DALAL; TRIGGS, 2005; KLASER; MARSZAŁEK; SCHMID, 2008; FATHI; MORI, 2008; WANG et al., 2011; MAIA et al., 2015; TORRES; PEDRINI, 2016).

Embora progressos notáveis tenham sido registrados em CV pela utilização dessas abordagens baseadas exclusivamente em *hand-crafted features*, esses métodos geralmente envolviam muito trabalho manual e conhecimento especializado de domínio em seu processo de desenvolvimento. Além disso, era notável que essas técnicas não apresentavam um desempenho satisfatório em grandes conjuntos de dados. Com o surgimento de técnicas de aprendizado profundo, parte desses problemas foram mitigados. Abordagens de DL impulsionaram consideravelmente o domínio HAR, graças a sua capacidade de projetar *features* que generalizavam muito bem sob conjuntos de dados maiores. *Features* que possibilitaram o surgimento de classificadores cada vez mais robustos para vários problemas de CV, conforme demonstrado no Capítulo 1. Outra possibilidade interessante apresentada por essas técnicas era fornecer mecanismos para o uso destas representações de vídeo como entrada de suas arquiteturas, aprimorando o desempenho dessas *hand-crafted features* em conjuntos de dados mais densos e, consequentemente, melhorando o processo de reconhecimento de ações. Além disso, esses métodos mostraram novas maneiras de abordar com eficiência o problema de modelagem temporal, como modelos híbridos (WANG et al., 2017a; ULLAH et al., 2018; ARIF et al., 2019) ou abordagens multi-fluxo baseadas em DCNNs (SIMONYAN; ZISSERMAN, 2014a; WANG et al., 2017; CHOUTAS et al., 2018).

2.3 TRABALHOS RELACIONADOS

Dado os avanços significativos no campo de classificação de imagens proporcionados, principalmente, pelo surgimento da LeNet-5 (LECUN et al., 1998) e da AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), era natural esperar que arquiteturas inspiradas nesses dois trabalhos inovadores surgissem em campos correlatos como o de classificação de vídeos. As primeiras arquiteturas derivadas nesse domínio geralmente adicionavam uma dimensão ao modelo 2D de CNN, para acomodar a dimensão extra, temporal, presente nos vídeos. Esses modelos tratavam espaço e tempo como dimensões equivalentes na entrada, executando as convoluções nos dados, temporalmente e espacialmente (BACCOUCHE et al., 2011; JI et al., 2013; TRAN et al., 2015). Em um desses trabalhos pioneiros, Baccouche et al. (2011) realizaram convoluções 3D em *frames* adjacentes dos vídeos para extrair *features*, temporais e espaciais, que eram utilizadas no treinamento de uma rede neural recorrente (*recurrent neural network* – RNN). A RNN então classificava a ação representada por cada uma dessas sequências, analisando a evolução temporal das *features* aprendidas pela CNN 3D. Ainda que os autores desse trabalho tenham demonstrado que o método proposto por eles superava as abordagens estado da arte, os resultados ainda eram muito próximos dos apresentados por arquiteturas baseadas em *hand-crafted features*, que, em geral, demandavam um esforço computacional bem inferior ao necessário para treinar uma rede 3D. Até esse momento, o *tradeoff* entre custo computacional e desempenho ainda não favorecia a utilização de modelos profundos no ambiente HAR.

Os primeiros resultados verdadeiramente expressivos da aplicação de técnicas de aprendizado profundo para resolver o HAR foram alcançados basicamente por CNNs 2D adaptadas para extrair informações de vídeos. Esse truque teve que ser empregado devido à alta complexidade relacionada ao treinamento e avaliação de uma rede 3D, que limitava muito o desempenho dos trabalhos iniciais, baseados em arquiteturas profundas 3D. Um dos primeiros trabalhos nessa direção foi o de Karpathy et al. (2014). Nesse trabalho, os autores chegaram a conclusão de que um único *frame* processado de cada vídeo já era suficiente para identificar grande partes das ações nas base de dados utilizadas. A CNN 2D proposta por eles era basicamente a arquitetura AlexNet, pré-treinada com a base de dados Imagenet, com ligeiras adaptações para funcionar no ambiente de reconhecimento de vídeos. Os resultados dos experimentos demonstraram que essa arquitetura apresentava um desempenho similar às redes que usavam pilhas de *frames* como entrada, com bem menos parâmetros a serem otimizados, o que acelerava o processo de treinamento e tornava a rede menos suscetível a sobre-ajuste (*overfitting*) (KONG; FU, 2018). Em suas conclusões, os autores cogitaram que o desempenho inferior dos modelos 3D também pudesse estar relacionado ao fato de que as *features* aprendidas por essas arquiteturas poderiam não estar capturando corretamente as informações de movimento necessárias para identificar as ações.

Embora o trabalho de Karpathy et al. (2014) ressalte a importância da informação espacial no processo de reconhecimento das ações, os próprios autores destacam que informação temporal é crucial para processar corretamente as ações nos vídeos.

Para trabalhar simultaneamente com os dois aspectos do vídeo, temporal e espacial, Simonyan e Zisserman (2014a) propuseram o conceito de uma arquitetura de dois fluxos (*two-stream*). Nesse esquema, o aspecto espacial era abordado por uma rede treinada com informações estáticas dos *frames*. Para a rede temporal, o fluxo óptico era usado para reconhecer as ações por meio de estimativas locais de movimento. A arquitetura proposta pelos autores é considerada um marco da área, sendo a base de uma série de trabalhos posteriores (WANG et al., 2015; CONCHA et al., 2018; TACON et al., 2019). Entre as principais contribuições desse trabalho estão as constatações dos autores de que uma informação explícita de movimento, fornecida pelos fluxos ópticos, pode melhorar o desempenho dos modelos, facilitando o processo de identificação das ações. A rede temporal proposta nesse trabalho superava com folga os outros modelos temporais que geralmente usavam pilhas de *frames* para inferir indiretamente essas estimativas de movimento necessárias para definir certas ações (KARPATHY et al., 2014; RAVANBAKHSH et al., 2015; KAHANI; TALEBPOUR; MAHMOUDI-AZNAVEH, 2017).

O modelo original *two-stream* não permite que informações sejam compartilhadas diretamente entre as *streams*. Essa falta de comunicação impede que esses modelos aprendam características espaço-temporais que podem ser úteis para aprender algumas ações (KONG; FU, 2018). Com base nessas afirmações, várias *features* foram propostas, posteriormente, tentando correlacionar as informações processadas por essas duas *streams* (DIBA; SHARMA; GOOL, 2017; WANG; TRAN; HOAI, 2017; FEICHTENHOFER; PINZ; WILDES, 2017; CONCHA et al., 2018; CHOUTAS et al., 2018).

Wang, Tran e Hoai (2017), por exemplo, apresentaram o *eigen evolution pooling* – EEP, uma *feature* espaço-temporal obtida dos vídeos, criada com o intuito de aproximar o efeito dessa interação entre as *streams* de uma arquitetura. A ideia principal por trás do EEP é criar uma representação única para um grupo de *features* extraídas dos *frames* RGB. Cada grupo é mapeado para um conjunto ordenado de funções unidimensionais que é posteriormente fornecido como entrada de uma CNN. Os autores demonstram que essa forma de sintetizar as informações de um vídeo apresenta uma performance superior a de outros métodos criados com essa finalidade.

Visando esse mesmo aspecto espaço-temporal dos dados, Choutas et al. (2018) introduziram o PoTion. O PoTion é uma representação capaz de capturar a movimentação de certos pontos chave nos vídeos. Esses pontos, usualmente associados às articulações dos atores são calculados por meio de estimativas de pose e representados na forma de mapas de calor. Os mapas de calor de toda a extensão do vídeo são calculados e posteriormente somados para formar uma única representação final (uma imagem) que será fornecida

como entrada de uma outra rede. Uma arquitetura rasa, formada por poucas camadas, foi desenvolvida pelos autores para processar os PoTions. Embora o desempenho isolado dessa *feature* não seja satisfatório, os autores demostraram que em conjunto com a I3D (CARREIRA; ZISSERMAN, 2017), o PoTion supera os trabalhos estado da arte desse domínio.

Mais recentemente, Concha et al. (2018) introduziram o ritmo visual ao ambiente *multi-stream*. Ao fornecer uma estimativa da trajetória percorrida por um objeto em uma única imagem, o ritmo visual se torna uma *feature* capaz de representar ambos os aspectos do vídeo de uma só vez, permitindo com que esses dois aspectos sejam analisados em conjunto. Ainda que o VR apresente resultados inferiores às outras *streams* usadas geralmente nesse domínio, como o RGB e o fluxo óptico, sua aplicação combinada ao modelo *two-stream* original alcançou resultados comparáveis ao estado da arte. Um fato que demonstra o aspecto complementar dessa *feature* às outras informações utilizadas neste domínio. Além disso, como o VR é uma representação compacta de todo o vídeo, ele pode ser utilizado para abordar outra questão deixada em aberto pela arquitetura *two-stream* original, que é como representar de forma eficiente as relações temporais de longo alcance nos vídeos. Alguns experimentos nesse sentido foram realizados pelos autores, mostrando que o VR possivelmente pode ser capaz de modelar essas relações temporais longas nos vídeos. Porém, testes mais completos ainda terão que ser realizados para comprovar a aplicabilidade do VR com essa finalidade.

Representações temporais de longo alcance capturam as variações de movimento executadas ao longo de todo o vídeo associadas a realização de certas ações. Para capturar essas possíveis alterações de movimentação, Ng et al. (2015) avaliaram várias estratégias para criar pilhas de informações extraídas dos vídeos (RGBs e fluxos ópticos) de tamanhos anteriormente nunca testados. A arquitetura proposta por esses autores chegava a processar 120 *frames* por vídeo. Com esse procedimento, os autores tentavam combinar e processar informações de períodos maiores dos vídeos, visando codificar, principalmente, possíveis relações temporais mais longas, relacionadas a execuções de certas ações. Para agregar o conhecimento dessas sequências de informação, eles examinaram duas abordagens: camadas de agrupamento (*pooling*) e redes recorrentes LSTMs (*long short-term memory*).

Outra solução nesse contexto foi apresentada por Wang et al. (2016). A rede *two-stream* de segmentos temporais (*temporal segment network* – TSN), arquitetura criada por esses autores, aborda o problema de modelagem temporal de longo alcance nos vídeos, utilizando um esquema de amostragem esparsa, onde clipes curtos (segmentos) são extraídos das amostras de vídeo e usados como entrada de uma arquitetura. Nesse esquema, uma função de consenso entre segmentos produz a predição final de uma amostra. Essa função relaciona esses vários segmentos para construir uma representação temporal que explore relações temporais longas nos vídeos. Além disso, esse método tem a vantagem de não

ser limitado temporalmente, ou seja, vídeos com quantidades arbitrariamente grandes de *frames* podem ser utilizados.

Uma nova abordagem foi introduzida por Carreira e Zisserman (2017). Nesse trabalho, os autores inflaram algumas camadas de uma arquitetura para possibilitar o uso de convoluções 3D. Para isso, os autores expandem uma dimensão dos filtros de algumas camadas elevando os mesmos a um espaço tridimensional. Com isso, os autores argumentam que relações temporais mais longas e aspectos espaço-temporais passam a ser parte do conhecimento possível de ser extraído desse modelo. Os resultados expressivos fizeram da arquitetura derivada nesse trabalho, a I3D, o estado da arte do problema de reconhecimento de ações humanas em vídeos. No entanto, pode-se notar que a rede proposta foi originalmente treinada para o conjunto de dados Kinetics (KAY et al., 2017), que contém cerca de 300 mil vídeos, uma quantidade muito mais expressiva de dados que a presente na maior parte dos *datasets* anteriormente utilizados nesse domínio (KONG; FU, 2018). Desde então, os melhores resultados nos *datasets* usualmente empregados no domínio HAR foram alcançados explorando redes pré-treinadas no conjunto de dados Kinetics (CARREIRA; ZISSERMAN, 2017; CHOUTAS et al., 2018; ZHU; ZHU; ZOU, 2018; WANG et al., 2018). Apesar desses resultados, o tamanho do *dataset* Kinetics ainda é um fator desafiador, já que uma quantidade substancial de recursos computacionais é necessária para processá-lo.

3 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo estão descritos os principais conceitos relacionados ao ferramental teórico utilizado para abordar o tema proposto. Conceitos básicos referentes à rede neurais nas Seções 3.1 e 3.2, para que o leitor consiga compreender o funcionamento básico de uma arquitetura *multi-stream*, conforme apresentado na Seção 3.3, e conceitos referentes à meta-heurística aplicada na construção do método de fusão proposto neste trabalho, *simulated annealing*, na Seção 3.4.

3.1 REDES NEURAIS

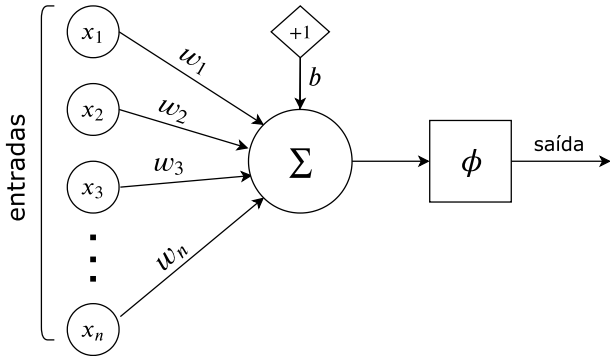
No paradigma convencional de programação, ao modelar um certo processo computacionalmente, é necessário definir todas as possíveis interações e listar cada passo necessário para realização dessa tarefa (NIELSEN, 2015). Entretanto, certos problemas são tão complexos que não permitem uma especificação tão precisa. Nestes casos, geralmente, tem-se pouca informação sobre o fenômeno a ser modelado, salvo observações coletadas do mesmo. Um dos problemas que encaixam nessa descrição é o de classificação de vídeos.

Redes neurais apresentam-se como alternativas viáveis para modelar esses tipos de fenômenos (HAYKIN, 1994). Esses modelos aprendem a construir uma solução com base no conjunto de amostras fornecidas do problema analisado, em princípio, sem qualquer informação prévia de como solucioná-lo. Uma rede neural, resumidamente, é um sistema computacional formado por um conjunto de estruturas mais simples que processam informação e armazenam o conhecimento extraído nesse processo. Cada neurônio de uma NN, nome dado a essas pequenas unidades de processamento, funciona como um modelo matemático simples que mapeia para cada conjunto de entradas x fornecido, um valor de saída y. Rosenblatt (1958) aprimorou esse conceito inicial de neurônio de McCulloch e Pitts (1943), adicionando uma espécie de ponderação a estes cálculos. Esses pesos adicionados à formulação dos neurônios, $\{w_1, w_2, \ldots, w_n\}$, são números reais que expressam a importância das respectivas entradas para o valor de saída. Esse novo tipo de neurônio, Perceptron, é considerado um verdadeiro marco na área de ML. Essa formulação de neurônio apresentada por Rosenblatt (1958), inspirou o conceito de neurônio artificial usado no modelo de NN comumente adotado na literatura, representado pela Figura 3.

Com base nessas ideias, um neurônio é essencialmente formado por entradas, pesos e uma função de ativação. O processo de computação de uma amostra nesses modelos consiste de um passo inicial onde as entradas são ponderadas pelos seus respectivos pesos. Todos esses valores ponderados são somados, em uma etapa posterior, e acrescidos de uma constante (viés), como especificado a seguir:

$$y = \sum_{i=1}^{n} w_i x_i + b, \tag{3.1}$$

Figura 3 – Neurônio artificial.



Fonte: elaborado pelo autor.

onde w_i corresponde ao peso associado a i-ésima entrada x_i, b é o viés (*bias*).

Esse resultado, então, passa por uma função de ativação que calcula o valor de saída do neurônio da seguinte forma:

$$f = \phi(y), \tag{3.2}$$

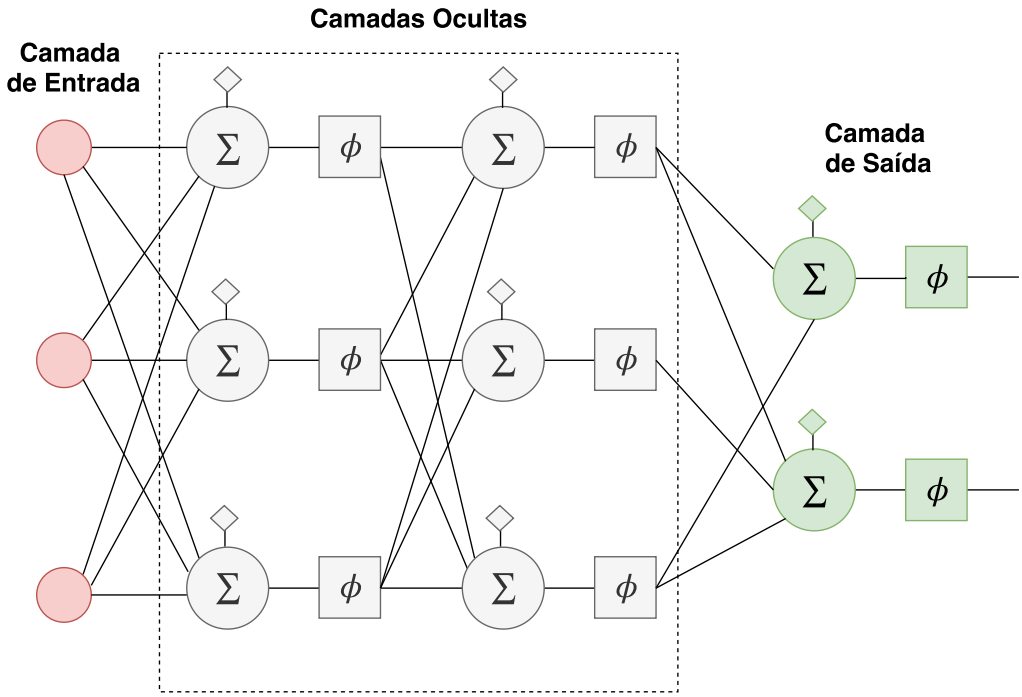
onde f corresponde ao valor da saída y transformado por uma função de ativação ϕ.

A simplicidade e eficiência do perceptron na solução de problemas linearmente separáveis disseminou o seu uso na época. No entanto, não demorou muito para os pesquisadores perceberem as restrições severas desse modelo frente a problemas reais, não-lineares em sua quase totalidade (MINSKY; PAPERT, 1969). Essas conclusões interromperam as pesquisas em NN por vários anos até que foi demonstrado que essas limitações poderiam ser contornadas adicionando mais camadas e funções não lineares à arquitetura desses modelos (HAYKIN, 1994). Neste contexto, são introduzidas as redes neurais de múltiplas camadas.

Em uma rede multicamadas, os neurônios estão dispostos paralelamente em grupos ou simplesmente camadas. Os neurônios na primeira camada recebem os dados de entrada que são encaminhados para as camadas intermediárias, ocultas. Essas camadas ocultas processam e repassam essas informações, tentando abstrair algum conhecimento desse cálculo. O processamento em uma rede flui até a última camada. Essa camada de saída é responsável por sintetizar os conhecimentos extraídos em uma ou mais saídas. Quando cada neurônio de uma determinada camada está conectado a todos os neurônios de uma camada anterior, diz-se que essa camada é totalmente conectada. Uma rede com essas características está ilustrada na Figura 4.

O ato de simplesmente adicionar mais camadas à arquitetura não confere ao modelo o caráter não linear necessário para abordar determinados problemas. De fato, uma rede composta por neurônios lineares permanece linear, não importa quantas camadas ela tenha (HAYKIN, 1994). Uma NN de múltiplas camadas, portanto, deve fazer o uso de funções de

Figura 4 – Rede neural multicamadas (camadas totalmente conectadas em destaque).



Fonte: elaborado pelo autor.

ativação não lineares nos neurônios, visando solucionar problemas dessa natureza. Embora várias funções desse tipo tenham sido propostas na literatura, no âmbito do presente trabalho, o foco será em duas funções de ativação muito utilizadas no campo de DL: a ReLU e a softmax.

Uma maneira simples e efetiva de criar uma rede não linear é por meio de unidades lineares retificadas conhecidas popularmente como ReLUs. A ReLU é basicamente uma função identidade para valores positivos e que mapeia todos os valores negativos para o eixo das abscissas, ou seja:

$$\phi(y) = \max\ (0, y). \tag{3.3}$$

Ela é uma função fácil de computar e diferenciável em todo o domínio, exceto no zero, um detalhe que demonstrou ter pouca importância na prática (LECUN; BENGIO; HINTON, 2015). A função ReLU se tornou uma alternativa a outras funções mais lentas para calcular, como a sigmoide. Funções clássicas que normalmente sofrem com problemas graves, como a saturação do gradiente, quando aplicadas em arquiteturas mais profundas (LECUN; BENGIO; HINTON, 2015).

A outra função descrita nesse trabalho, softmax, é usada nos neurônios de saída das redes em problemas de classificação multiclasse (BISHOP, 2006). É uma função que mapeia um vetor de valores quaisquer arbitrariamente grandes em um vetor normalizado, de mesmo tamanho, com valores que variam no intervalo $[0, 1]$ e que somam 1. A saída de

uma unidade softmax pode ser interpretada como um vetor com a respectiva probabilidade de pertencimento de uma determinada amostra a cada uma das C classes utilizadas. Seu cálculo é realizado da seguinte forma:

$$\phi(y_i) = \frac{\exp(y_i)}{\sum_{c=1}^{C} \exp(y_c)}, \tag{3.4}$$

para $i = 1, \ldots, C$.

Essas duas funções, ReLU e softmax, assim como outras usadas nesse domínio, devem ser diferenciáveis para poderem ser aplicadas ao algoritmo usado para estimar o conjunto de pesos de uma NN. A próxima seção apresenta uma breve descrição desse algoritmo de treinamento denominado algoritmo de retropropagação de erro, sucintamente conhecido como *backpropagation*.

3.1.1 ***Backpropagation***

Uma rede neural é treinada ajustando os pesos de todos os neurônios de modo que a rede aprenda a aproximar as saídas de seus valores reais, auferidos diretamente da função a ser modelada. Definir esses pesos de forma analítica é praticamente impossível, mesmo em arquiteturas com poucos neurônios, justificando o uso de técnicas que tornem esse procedimento viável ainda que em condições bem controladas. O algoritmo de *backpropagation* provê uma forma iterativa de estimar esse conjunto de pesos. Sua versão clássica usa o gradiente descendente como método de otimização. Usando o conjunto de hiperparâmetros apropriado, percebe-se que esse algoritmo funciona suficientemente bem na prática, embora não haja garantia de que o mínimo global será encontrado nesse processo.

Na primeira fase do algoritmo, um vetor de entradas é propagado através da rede neural produzindo uma saída. Esse valor fornecido pela rede é comparado com a saída esperada, por meio de uma função que mensura a discrepância desses valores, chamada função de perda. O gradiente dessa função é então computado. Esses valores de gradiente, erros, são retropropagados pela rede para calcular esses mesmos valores nos neurônios das camadas ocultas. Nessas camadas, os gradientes são calculados usando a regra da cadeia para derivadas. Os pesos dos neurônios, então, são finalmente atualizados levando em consideração seus respectivos erros ponderados por uma constante que controla essas alterações. Essa constante, também chamada de taxa de aprendizado, é um parâmetro bem sensível nesses métodos, sendo que taxas muito baixas retardam o processo de convergência do algoritmo, e taxas muito altas podem fazer a função divergir. A taxa ideal depende do problema e geralmente é fruto de muita análise empírica. É comum usar uma taxa dinâmica, ajustável às várias etapas desse processo (HAYKIN, 1994).

Com o decorrer do tempo, os valores de saída tendem a se aproximar dos seus valores reais. Ou seja, os erros tendem a diminuir até que um ótimo da função de perda

seja alcançado. O algoritmo continuará a ser executado com entradas diferentes até atingir esta etapa do processo, onde espera-se que o conjunto de pesos convirja para valores bem próximos aos obtidos via solução analítica.

3.2 REDES NEURAIS CONVOLUCIONAIS

Não é usual abordar problemas de CV por meio de NN tradicionais como a apresentada na Figura 4. Isso tem relação com o tipo de dado utilizado nessa área. Os vários obstáculos já descritos anteriormente, associados à natureza dessas informações, fazem com que esse modelo padrão de NN não apresente um desempenho satisfatório quando aplicado a esse domínio. Problemas como, por exemplo, a maldição da dimensionalidade dificultam a convergência do processo de treinamento desses modelos (BISHOP, 2006). Além disso, todo esse procedimento consumiria uma quantidade significativa de recursos computacionais, inviável, na maior parte das situações. Utilizar as representações descritas na Seção 2.1 é uma boa forma de amenizar esses problemas, porém vários autores já demonstraram que existem modelos de ML mais eficientes que uma NN padrão nessas circunstâncias (KONG; FU, 2018). Outro ponto que não favorece a utilização desses modelos é o fato deles não serem invariantes à operação de translação (LECUN; BENGIO; HINTON, 2015). Assim, uma rede desse tipo deveria ser treinada para reconhecer separadamente o mesmo padrão em inúmeras posições de uma imagem, o que seria computacionalmente impraticável.

Uma CNN, por outro lado, é um tipo de NN adaptado a este ambiente de CV. CNNs são modelos fundamentados por um conceito de biologia denominado campo receptivo. Campos receptivos são áreas do córtex visual animal que funcionam como detectores. Eles respondem a certos estímulos no campo visual relacionados, por exemplo, a bordas ou quinas (HUBEL; WIESEL, 1968). No processamento de imagens, esse comportamento pode ser simulado, conforme apresentado por Marr e Hildreth (1980), por meio de uma operação matemática chamada convolução, ilustrada na Figura 5. Convoluções podem ser usadas na filtragem de imagens e são a base de funcionamento das CNNs.

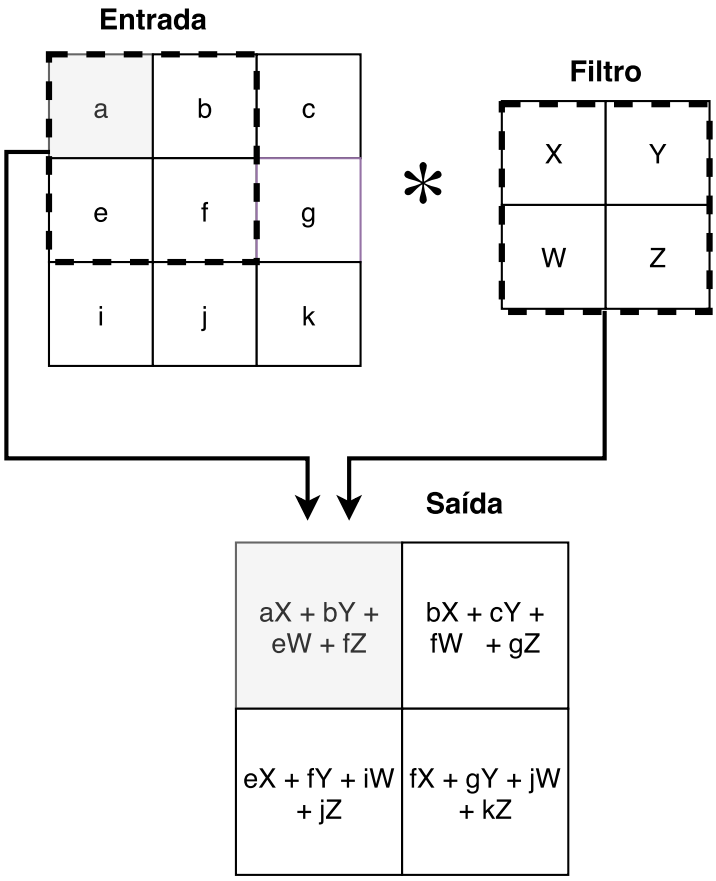
A convolução entre uma imagem F e um filtro g é realizada da seguinte forma:

$$h(u,w) = F(u,w) * g(u,w) = \sum_n \sum_m F(n,m) \cdot g(u-n, w-m) \tag{3.5}$$

onde $h(u,w)$ representa o valor do *pixel* no ponto $p = (u,w)$ produzido pelo produto vetorial de uma subimagem $F(u,w)$ e um filtro $g(u,w)$ centrados nesse mesmo ponto p. m e n são as dimensões da imagem original.

Nas bordas da imagem, um tratamento especial deve ser realizado para que essa operação permaneça válida. Isso pode ser feito simplesmente completando os *pixels* faltantes com um valor constante, ou por meio de uma operação mais elaborada como uma reflexão (LECUN; BENGIO; HINTON, 2015). Às vezes, uma convolução é operada somente sob certos *pixels* de uma imagem. Um parâmetro denominado *stride* controla esse

Figura 5 – Um exemplo de convolução 2D sem *stride* sendo realizada em uma matriz de entrada. Nenhum tipo de tratamento de bordas está sendo aplicado nessa situação. Com isso, o sinal de saída está restrito às posições onde o filtro está inteiramente dentro da matriz.



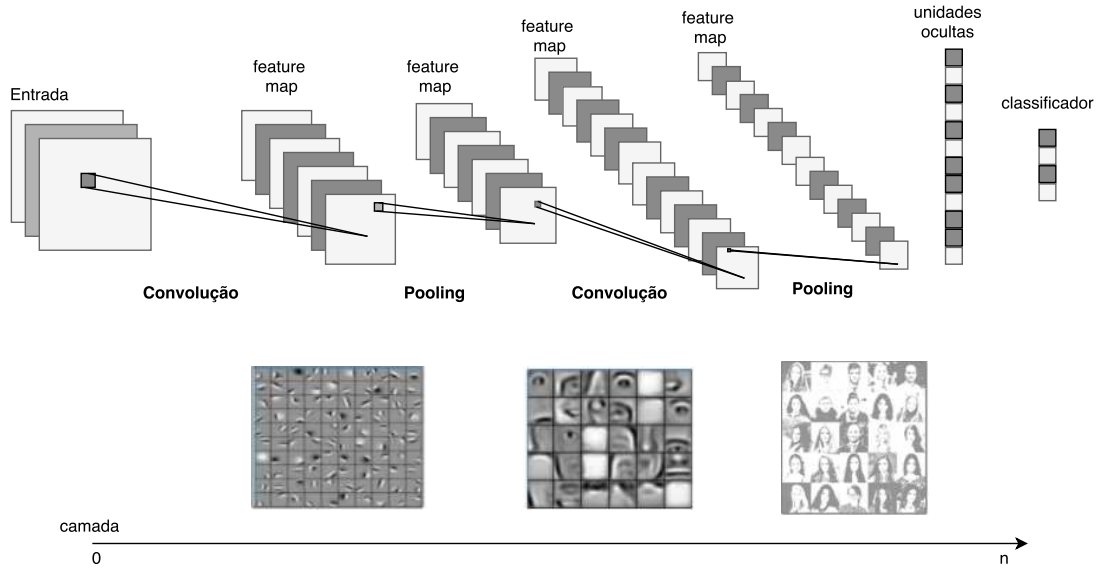
Fonte: elaborado pelo autor.

processo de subamostragem da imagem original. A aplicação sucessiva do filtro a cada possível subimagem extraída de F produz a matriz final h correspondente à imagem filtrada. Essa matriz final, no contexto de uma CNN, é conhecida como mapa de características (*feature map*) (LECUN; BENGIO; HINTON, 2015). Uma camada convolucional de uma CNN é essencialmente uma combinação de vários desses filtros convolucionais.

Camada convolucional: em uma camada convolucional, os pesos são agrupados localmente na forma de filtros convolucionais. Esses pesos são ajustados utilizando a mesma estratégia de treinamento aplicada à uma NN tradicional. Embora o mesmo procedimento de treino seja empregado em ambas arquiteturas, os conhecimentos adquiridos durante esse processo são diferentes. Em uma CNN, os filtros em camadas mais próximas à entrada aprendem a reconhecer detalhes mais simples (*low-level features*) das imagens, como bordas e cantos. Camadas posteriores aprendem a combinar essas *features* com o intuito de criar abstrações de mais alto nível, detectando padrões mais complexos nos dados, como rostos, penas, entre outros (LECUN; BENGIO; HINTON, 2015). A operação de convolução é sempre acompanhada de uma função de ativação em uma camada convolucional padrão.

A Figura 6 dá uma noção de como funciona todo esse procedimento.

Figura 6 – Fluxo de aprendizado de uma arquitetura CNN. *Features* em múltiplos níveis de abstração. Padrões mais simples abstraídos pelas camadas mais próximas a entrada e padrões complexos como rostos nas camadas mais próximas ao classificador.



Fonte: elaborado pelo autor.

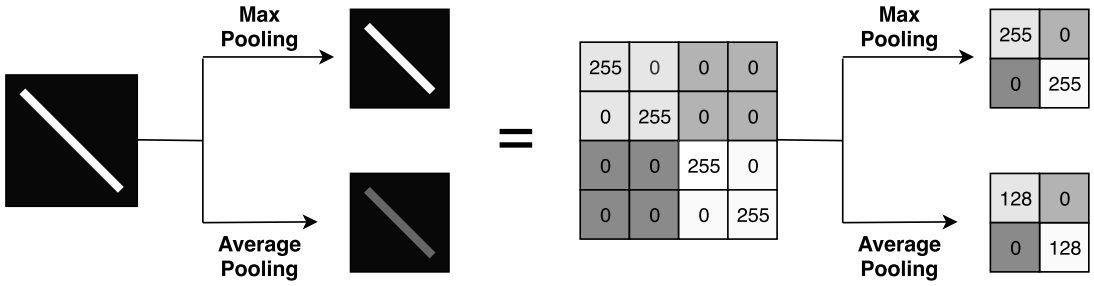
Os filtros aprendidos por uma camada convolucional são aplicados em todas as partes da imagem. Dessa forma, há uma redução considerável no número de parâmetros a serem otimizados. Isso só é possível porque alguns neurônios compartilham parâmetros. Em certas regiões da matriz de entrada, um conjunto de parâmetros é comum a vários neurônios ligados localmente a essas regiões. É esse fato que garante a equivariância de uma CNN em relação à operação de translação. Uma função é dita equivariante se uma alteração em sua entrada provocar uma alteração idêntica em sua saída. Essa propriedade permite que um mesmo padrão, extraído de um conjunto de *pixels* por uma CNN, possa ser aplicado a várias regiões de uma entrada (LECUN; BENGIO; HINTON, 2015). As bordas detectadas nas primeiras camadas convolucionais, por exemplo, aparecem por toda a imagem. Com isso, é extremamente útil que parâmetros relacionados à identificação desse padrão sejam compartilhados e aplicados a diversas regiões do sinal de entrada.

Camada de *pooling*: Krizhevsky, Sutskever e Hinton (2012) definem a operação de *pooling* como um processo que sintetiza a presença de uma determinada *feature* em uma vizinhança $N \times N$, para um dado *feature map* h. A adição de uma camada de *pooling*, torna a rede resultante invariante a pequenas translações nas entradas. Essa invariância a translações locais é útil quando a posição precisa de uma determinada *feature* não é relevante para identifica-lá. Parte do processo de detecção de uma face, por exemplo, pode simplesmente ser executado por meio da identificação de um olho na parte esquerda e um na parte direita da face. Localizar com exatidão os olhos não melhora esse processo de

detecção (LECUN; BENGIO; HINTON, 2015).

Camadas de *pooling* mapeiam um *feature map* h em uma outra matriz, subamostrada de h, que indica a presença de uma determinada *feature* em uma região. Esse *pooling map* é criado destacando as maiores contribuições de uma *feature* (*max pooling*) a uma determinada região ou mensurando sua contribuição média (*average pooling*). A operação de *pooling* é uma maneira de reduzir o volume dos dados trafegados por uma camada sem que o conhecimento abstraído se perca totalmente neste processo (LECUN; BENGIO; HINTON, 2015). Um exemplo de um *pooling* aplicado sobre o *feature map* de uma imagem pode ser encontrado na Figura 7.

Figura 7 – Técnicas de *Pooling* aplicadas a um *feature map*. Imagem, à esquerda, e sua respectiva matriz de valores RGB à direita.



Fonte: (YU et al., 2014).

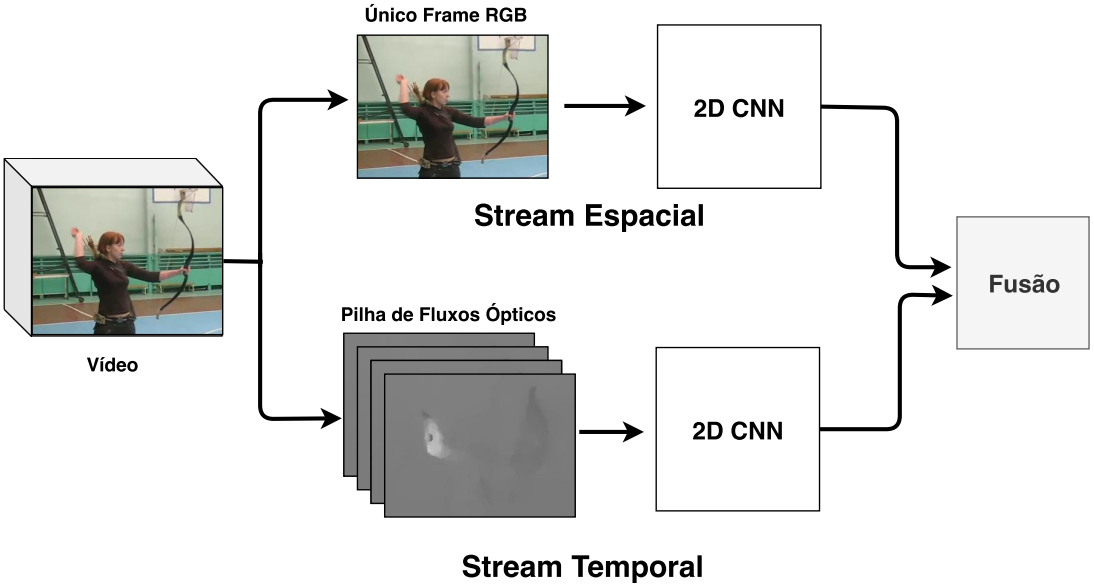
3.3 ARQUITETURAS *MULTI-STREAM*

Vídeos podem ser compreendidos por meio de seus aspectos espaciais e temporais. A parte espacial contém informações sobre a aparência dos *frames*, como cenas de fundo e objetos representados no vídeo. A parte temporal retrata o movimento entre os *frames*. Desenvolver uma arquitetura que funcione eficientemente nesses dois aspectos é um grande desafio na área de reconhecimento de ações humanas (KONG; FU, 2018).

Simonyan e Zisserman (2014a) propuseram o conceito de redes *two-stream*, com o intuito de trabalhar separadamente estes dois aspectos de um vídeo. Nesse esquema, o aspecto espacial é abordado por uma CNN treinada com informações estáticas dos *frames*. Na rede temporal, o OF é usado para reconhecer ações por meio de uma estimativa local de movimento. Os conhecimentos dessas redes são combinados em um passo posterior por um método de fusão. A arquitetura proposta pelos autores, ilustrada pela Figura 8, é considerada um marco na área de CV, sendo a base de uma série de trabalhos posteriores (WANG et al., 2015; CONCHA et al., 2018).

Com o decorrer do tempo, essa arquitetura padrão passou por vários incrementos. Novas *streams* foram propostas visando outros aspectos do vídeo como assinaturas temporais de longo alcance e espaço-temporais (CONCHA et al., 2018). Informações de

Figura 8 – Arquitetura *two-stream*. Uma CNN cuja entrada é um único *frame* e uma CNN que recebe uma pilha de fluxos ópticos como entrada. Um método de fusão combina os conhecimentos dessas duas redes.



Fonte: elaborado pelo autor.

outra natureza foram acopladas a estrutura desses modelos, como informações sonoras, textuais e de sensores (HWANG; CHA; OH, 2017). Novos tipos de CNN com maior poder de generalização foram propostas (CARREIRA; ZISSERMAN, 2017). Estas modificações tornaram modelos baseados em múltiplos fluxos de informação, o estado da arte no reconhecimento de ações em vídeos e acabaram por cunhar o termo *multi-stream* amplamente difundido atualmente.

3.3.1 *Stream* espacial

Dificilmente uma ação duraria pouco a ponto de ser capturada por um único *frame*. Entretanto, em certos casos, esse único *frame* pode fornecer informações suficientes para delimitar um espaço de possíveis ações sendo realizadas em uma cena. Certos ambientes, por exemplo, são característicos de determinadas ações. Por exemplo, a água em ações relacionadas a esportes aquáticos como "surfe" e "*jet ski*". Outras ações estão fortemente associadas a determinados objetos (SIMONYAN; ZISSERMAN, 2014a). "Tocar um instrumento musical" é um exemplo de ação com essa característica.

Essa forte associação entre ações e aspectos visuais, aliada a avanços significativos no âmbito de classificação de imagens, reforçam que o reconhecimento de ações pode alcançar resultados satisfatórios usando somente informação estática dos vídeos. A CNN usada como *stream* espacial por Simonyan e Zisserman (2014a), por exemplo, alcançou resultados competitivos em HAR processando somente um única imagem RGB extraída dos vídeos.

Concha et al. (2018) notaram que mesmo ações definidas pela presença de objetos e/ou cenários poderiam apresentar tantas variações durante o vídeo que somente um *frame* não seria capaz de caracterizá-las. Buscando contornar esse problema, eles propuseram uma alteração na *stream* espacial usada por Simonyan e Zisserman (2014a). Essa *stream* aperfeiçoada consiste de uma CNN treinada com dois *frames* por vídeo. Um *frame* extraído aleatoriamente da primeira metade e outro extraído da segunda metade do vídeo. Uma estratégia que, segundo os autores, aumentou a possibilidade dos *frames* selecionados definirem a ação realizada. Porém, mesmo essa abordagem não é capaz de distinguir ações com uma forte assinatura temporal, como é o caso em atividades como correr e andar. Esse fato destaca o papel fundamental que a *stream* temporal desempenha nas abordagens *multi-stream*.

3.3.2 *Stream* temporal

No contexto de classificação de vídeo, Simonyan e Zisserman (2014a) introduziram o uso de sequências de fluxo óptico como *features* temporais, a fim de complementar os métodos baseados exclusivamente em informação RGB usados neste domínio. Os autores argumentam que é possível obter uma representação do movimento em uma sequência de *frames*, empilhando os canais de fluxo d_t^x e d_t^y de L *frames* consecutivos. A pilha $I_t \in R^{h \times w \times 2L}$ para um *frame* arbitrário t é composta por $2L$ canais de fluxo como:

$$\begin{aligned} I_t(u, v, 2l-1) &= d^x_{t+l-1}(u, v) \\ I_t(u, v, 2l) &= d^y_{t+l-1}(u, v), \end{aligned} \tag{3.6}$$

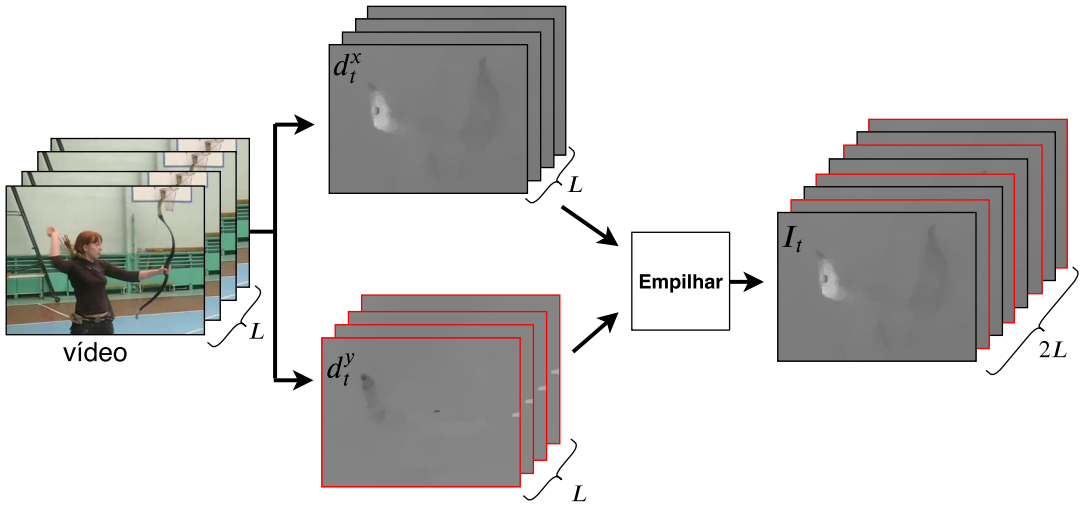
onde $l = \{1, \ldots, L\}$ representa *frames* consecutivos depois de t. Partindo de qualquer ponto $p = (u, v)$, com $u = \{1, \ldots, h\}$ e $v = \{1, \ldots, w\}$, o elemento $I_t(u, v, z)$, onde $z = \{1, \ldots, 2L\}$, codifica o movimento de p sobre uma sequência de *frames* L definidos no volume do vídeo V. A Figura 9 ilustra o processo de construção de I_t.

Como essa pilha descreve explicitamente o movimento entre os *frames*, a rede não precisa inferir essas informações de forma implícita a partir dos *frames*, o que facilita o processo de reconhecimento. Além disso, essa abordagem permite que a rede atinja um desempenho satisfatório mesmo em pequenos conjuntos de dados (SIMONYAN; ZISSERMAN, 2014a). Essas características direcionaram a escolha dessa formulação para compor o fluxo temporal da arquitetura proposta no presente trabalho.

3.3.3 *Stream* espaço-temporal

A arquitetura *two-stream* original falha em não considerar que o reconhecimento de algumas ações está atrelado ao aprendizado de *features* espaço-temporais nos dados (KONG; FU, 2018). A falta de comunicação entre as *streams* desse modelo dificulta o processo de identificação desse tipo de ação caracterizada simultaneamente pelas dimensões espaciais do vídeo (X e Y) e pelo tempo.

Figura 9 – Construção da pilha de Fluxos Ópticos I_t utilizada como entrada da rede temporal de Simonyan e Zisserman (2014a). A parte de geração das componentes d_t^x e d_t^y dos fluxos foi omitida para facilitar o entendimento do processo.



Fonte: elaborado pelo autor.

Tentando contornar esse problema, Concha et al. (2018) introduziram o VR como *feature* espaço-temporal em arquiteturas *multi-stream*. Com isso, as interações entre espaço e tempo necessárias para definir algumas ações são agregadas ao conhecimento extraível dos modelos. Buscando validar suas observações, Concha et al. (2018) adicionaram uma nova *stream* espaço-temporal a uma arquitetura bem conhecida e demonstraram que seu desempenhou melhorou após a inclusão da *stream* baseada no VR proposta por eles. O modelo adicionado era fundamentalmente uma CNN, similar à utilizada na *stream* espacial, cuja entrada era composta de VRs dos vídeos.

3.3.4 **Fusão**

A escolha precisa das *streams* que irão compor uma arquitetura *multi-stream* não garante o seu bom desempenho como solução para o HAR. De fato, um outro passo fundamental na construção de uma arquitetura deste tipo está na definição de como combinar os conhecimentos das *streams* que a constituem. Métodos ineficientes podem subestimar os resultados ao não extrair ou não combinar satisfatoriamente o conhecimento obtido dos fluxos, e métodos muito complexos podem tornar o processo inviável computacionalmente.

Uma fusão pode ser realizada, por exemplo, com base no treinamento de um segundo modelo que recebe como entrada os descritores de cada *stream* da arquitetura. Um modelo auxiliar que fica responsável por definir e pesar as características que cada *stream* apresenta que mais influenciam positivamente o processo de reconhecimento de uma determinada ação. Porém, é comum encontrar na literatura relacionada a esse tema, métodos mais

rudimentares. Métodos como uma média simples dos descritores das *streams*. A principal justificativa para isso é que essas técnicas atingem bons resultados praticamente sem custo computacional (KONG; FU, 2018). Entretanto, Wu et al. (2015) já demonstraram que esse tipo de fusão pode levar a resultados subótimos, mascarando o real desempenho da arquitetura. Como alternativa, Wu et al. (2015), apresentaram uma técnica de fusão adaptativa denominada *adaptive multi-stream fusion*, um método que aprende os pesos de fusão de cada classe de forma adaptativa, explorando correlações nos dados que, segundo os autores, carregam significado semântico. Em suas conclusões, os autores mostraram que seu método de fusão supera a fusão por média simples e outras abordagens muito utilizadas, como a *kernel fusion* e a fusão baseada em um modelo de regressão logística.

Um ponto em comum que vários desses métodos de fusão mais elaborados tentam endereçar é a noção de que as *streams* contribuem de forma heterogênea para o resultado final de uma rede *multi-stream*. É natural pensar assim, uma vez que, certas representações descrevem melhor as ações e acabam por ter um peso maior em seu processo de identificação se comparadas a outras *features* utilizadas. É por isso que uma média ponderada se apresenta como um método interessante neste domínio. Ponderar a contribuição de cada *stream* é uma técnica mais robusta que uma média simples e que, dependendo do método empregado na escolha dos pesos, pode ter um custo computacional inferior às estratégias baseadas em modelos.

Trabalhos recentes propõem formas inteligentes para definir estes pesos utilizados no processo de fusão. Técnicas alternativas a métodos exaustivos como o *grid search*. Meta-heurísticas são algumas dessas técnicas. Chernbumroong, Cang e Yu (2014), por exemplo, criaram um método evolucionário baseado em algoritmos genéticos para calibrar os pesos da fusão de uma arquitetura multissensor. Com conclusões similares ao do presente trabalho, os autores mostraram que técnicas bio-inspiradas podem ser alternativas viáveis como métodos para fundir múltiplas *streams*.

3.4 *SIMULATED ANNEALING*

A Seção 3.1 apresenta problemas para os quais não se conhece um algoritmo polinomial eficiente que gere soluções exatas (HAYKIN, 1994). Nessa situação, os métodos geralmente constroem soluções que, em condições controladas, podem ser consideradas aproximações suficientemente "boas". Um dos métodos desenvolvidos com essa finalidade são as meta-heurísticas. Ao atuar em situações onde se tem pouca informação disponível sobre o problema e/ou sob restrições severas de recursos computacionais, elas se tornaram uma ferramenta poderosa na solução desses problemas, ditos de otimização combinatória, em ambientes reais (BLUM; ROLI, 2001).

Uma meta-heurística, segundo Osman e Laporte (1996), compreende um processo iterativo encarregado de guiar heurísticas especializadas subordinadas a ele, e que mescla

várias técnicas com o objetivo de explorar, de forma eficiente, o espaço de soluções de um determinado problema. Essa redução no espaço de busca é feita direcionando o processo de obtenção das soluções. Para isso, elas usam uma função a ser otimizada como uma espécie de guia. Com base em alterações nessa função, uma meta-heurística percorre sistematicamente esse espaço de busca, evitando regiões com um grande número de soluções homogêneas ou próximas a muitos ótimos locais.

Uma das áreas que meta-heurísticas tem apresentado resultados interessantes é a de ML. Neste campo, elas geralmente são aplicadas com o intuito de auxiliar algoritmos de ML, atuando em tarefas como, por exemplo, a de construção e hiperparametrização de modelos (ACEVEDO et al., 2007; KHALIFA et al., 2017; WANG et al., 2018; MATTIOLI et al., 2019), ou no refinamento das soluções geradas por esses métodos (RODRIGUEZ et al., 2006; MARAVALL; LOPE; H., 2009). Algoritmos genéticos (GOLDBERG, 1989), resfriamento simulado (*simulated annealing* – SA) e o algoritmo de otimização por enxame de partículas (*particle swarm optimization* – PSO) (KENNEDY, 2010) são algumas das meta-heurísticas utilizadas nessas arquiteturas híbridas.

A meta-heurística selecionada para compor o método proposto, o *simulated annealing*, associa um conceito de energia à qualidade das soluções obtidas. Toda mudança em uma solução acarreta em um aumento ou diminuição dessa energia. Neste procedimento, a melhor solução é encontrada maximizando uma função relacionada a essa energia (KIRKPATRICK; GELATT; VECCHI, 1983).

O algoritmo parte de uma solução inicial qualquer, geralmente, randômica, que a cada iteração é modificada, provocando uma perturbação no equilíbrio energético do sistema. Alterações que elevam essa energia são aceitas imediatamente. Caso contrário, uma diminuição é permitida condicionalmente segundo uma probabilidade $p = e^{\frac{-\Delta f}{KT}}$, onde Δf denota a variação na função de energia, T é a temperatura corrente e K é uma constante usada para relacionar temperatura e energia. Nos estágios iniciais deste processo, o sistema é submetido a uma temperatura inicial altíssima que, durante o procedimento, diminui por meio da Equação (3.7) até o ponto de cristalização do sistema. Em cada temperatura, o espaço de busca do problema é explorado por um número definido de iterações até que o sistema atinja um estágio conhecido como equilíbrio térmico. Neste ponto, o sistema é resfriado em um esforço para aumentar sua energia.

$$T_n = \alpha\, T_{n-1}, \tag{3.7}$$

onde T_n é a temperatura em um estágio de resfriamento n e α é um fator associado à estratégia de resfriamento adotada.

Este algoritmo é executado até que a temperatura se aproxime de zero, indicando que o sistema "cristalizou". Neste estado, qualquer alteração feita não causará qualquer melhoria na função objetivo, e nenhuma solução com um valor inferior à melhor pode ser

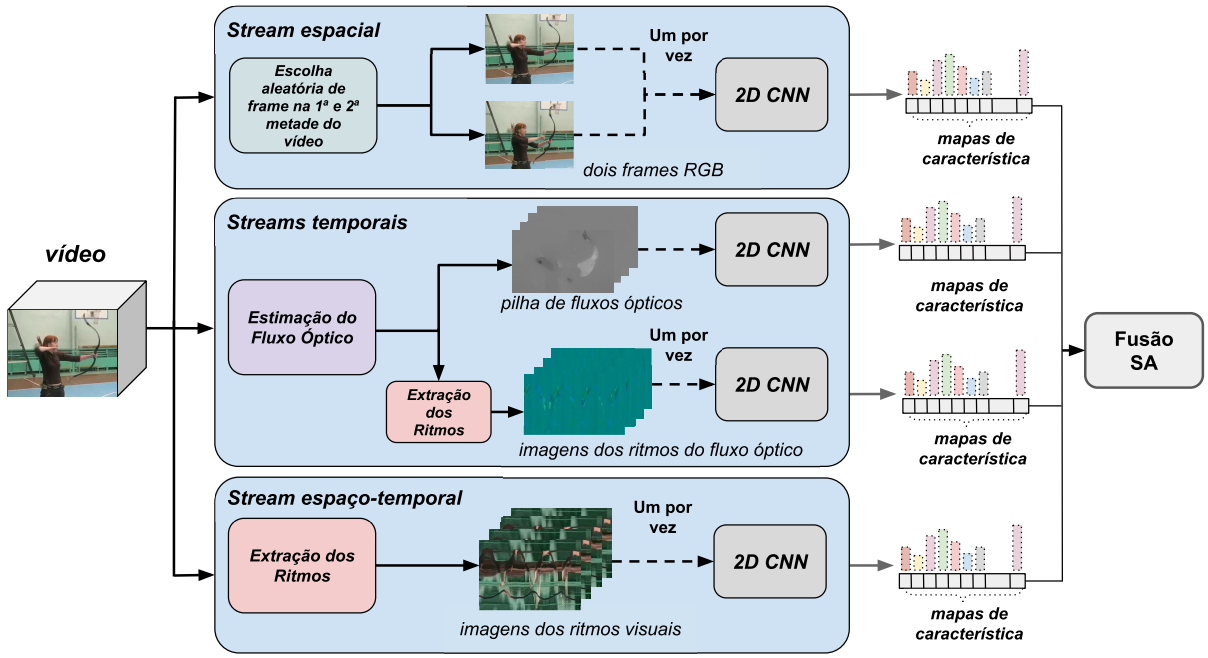
aceita. Espera-se que neste momento, a melhor solução corresponda a um ótimo global da função utilizada ou esteja bem próxima dessa solução ótima.

Devido à sua flexibilidade e simplicidade, o SA é amplamente usado em vários problemas de otimização (KOULAMAS; ANTONY; JAEN, 1994). No âmbito de ML, por exemplo, Acevedo et al. (2007) propuseram um método de seleção de hiperparâmetros para o L1-SVM baseado nessa meta-heurística. Em suas conclusões, os autores demonstraram que o desempenho do método desenvolvido foi satisfatório mesmo em situações onde um número considerável de hiperparâmetros era utilizado, ou quando os parâmetros eram variados em intervalos maiores. Todos estes aspectos direcionaram a escolha dessa meta-heurística para compor o método de fusão desenvolvido no presente trabalho.

4 MÉTODO PROPOSTO

A arquitetura proposta consiste basicamente de quatro CNNs profundas treinadas em diferentes modalidades: espacial (RGB), espaço-temporal (ritmo visual) e temporal (fluxo óptico e ritmo do fluxo óptico). Duas dessas *streams*, a espaço-temporal (MVR) e a temporal baseada no OFR (MOFR), são contribuições desse trabalho. O MVR é uma versão modificada da *stream* espaço-temporal introduzida originalmente por Concha et al. (2018), que utiliza múltiplos VRs ao invés de um único na entrada. A outra *stream*, MOFR, baseia-se em uma CNN, similar à utilizada na *stream* espacial, que recebe como entrada múltiplos ritmos do fluxo óptico (*optical flow rhythms* – OFRs). É importante mencionar ainda, que um protocolo de treinamento e teste foi especificado exclusivamente para essas duas novas *streams*. A Figura 10 fornece uma visão geral do método proposto.

Figura 10 – Visão geral do método proposto.



Fonte: adaptado de Concha et al. (2018).

Cada *stream* desta nova abordagem *multi-stream* é treinada de antemão e depois combinada através de um novo método de fusão desenvolvido em conjunto com a arquitetura, a fusão SA. Essa fusão é realizada ponderando a camada da rede que precede a softmax. O conjunto de pesos usado nessa ponderação é definido por meio da meta-heurística SA aplicada ao resultado de uma validação cruzada no conjunto de treinamento.

Como diversos trabalhos sugerem que profundidade é essencial ao problema de

reconhecimento de ações humanas (KONG; FU, 2018), foi decidido que um modelo de CNN profundo, bem conhecido na literatura, seria utilizado no escopo deste trabalho como modelo base das *streams*. Optou-se pela InceptionV3 (SZEGEDY et al., 2016) dada a sua eficácia superior em classificar imagens se comparada a outras arquiteturas DL como a AlexNet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012) e a VGG16 (SIMONYAN; ZISSERMAN, 2014b). Estudos demonstraram empiricamente que realizar uma transferência de aprendizado (*transfer learning*) a partir de um conjunto de dados de imagens através do ajuste fino (*fine-tuning*) de uma rede pré-treinada melhora o desempenho do HAR (SIMONYAN; ZISSERMAN, 2014a). Com base nesse fato e considerando o desempenho da InceptionV3 no conjunto de dados Imagenet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), espera-se que os mesmos resultados positivos observados na classificação das imagens dessa base se reproduzam no ambiente HAR. Além disso, a InceptionV3 tem a vantagem de realizar menos operações e utilizar menos parâmetros que outras redes de melhor desempenho.

4.1 *STREAM* ESPACIAL

A Subseção 3.3.1 destaca a importância da informação visual no processo de identificação de ações. Por isso, os ótimos resultados reportados por Concha et al. (2018) tiveram um papel fundamental na escolha da *stream* espacial proposta por esses mesmos autores, assim como de seu protocolo de treinamento e teste, para compor a arquitetura deste trabalho.

Na fase de treinamento dessa *stream*, a rede recebe dois *frames* coletados por vídeo: um na primeira metade e outro na sua segunda metade, ambos escolhidos aleatoriamente. O protocolo de teste leva em conta várias imagens do vídeo: 25 *frames* são amostrados uniformemente de cada vídeo de teste e 10 novas amostras são produzidas a partir deles para aumento de dados. Essas novas amostras são geradas usando uma das estratégias de aumento por cortes (*cropping*) apresentadas por Wang et al. (2015). Nessa técnica, novas imagens são geradas por meio de cortes *crops* (centrais e nos cantos das imagens) e giros (*flips*) horizontais, totalizando 250 amostras geradas por vídeo. Essas imagens, então, são fornecidas uma a uma para a CNN, que gera os seus respectivos mapas de características. Uma média simples desses descritores, acompanhada da aplicação da função softmax, define a classe final de uma amostra.

4.2 *STREAM* TEMPORAL

A *stream* temporal baseada em OFs introduzida por Simonyan e Zisserman (2014a) é considerada um marco na área de CV. Seus resultados individuais superiores aos de outras *streams*, conforme reportado por uma série de trabalhos (SIMONYAN; ZISSERMAN, 2014a; WANG et al., 2015; CARREIRA; ZISSERMAN, 2017; CONCHA et al., 2018),

fizeram com que esse modelo fosse frequentemente escolhido para compor o fluxo temporal de arquiteturas *multi-stream*, o que inclui a arquitetura proposta.

Neste trabalho, uma versão ligeiramente modificada da InceptionV3 é utilizada como rede temporal. Ela é essencialmente uma CNN com 20 canais de entrada ao invés dos 3 usuais, presentes no modelo de classificação de imagens. No processo de treinamento dessa CNN temporal, 10 pares de imagens de OFs, extraídos de quadros consecutivos dos vídeos, são utilizados, formando os 20 canais de fluxo correspondentes à entrada da rede. Todas essas imagens são estimadas em um estágio prévio ao treinamento para evitar qualquer gargalo na arquitetura.

No procedimento de teste, 25 pilhas de OFs são selecionadas de cada vídeo. Cada uma dessas pilhas gera mais 10 novas amostras usando as mesmas técnicas de aumento de dados introduzidas por Wang et al. (2015). Isso faz com que a predição de um vídeo seja dada por uma média dos 250 mapas de característica extraídos a partir dele.

Uma segunda *stream* temporal baseada no ritmo do fluxo óptico é proposta neste trabalho com o objetivo de complementar os conhecimentos da *stream* temporal original.

4.3 RITMO DO FLUXO ÓPTICO

O ritmo do fluxo óptico é uma representação temporal introduzida como informação complementar às imagens RGB e a pilha de OFs usualmente utilizadas em arquiteturas *multi-stream*. Ela combina o poder discriminante do fluxo óptico e a redução de dimensionalidade com preservação da taxa de quadros (*frame rate*) dos ritmos visuais. O OFR é um corte bidimensional na sequência de OFs estimados para um vídeo. É uma representação temporal de longo alcance do fluxo óptico de certas partes do vídeo.

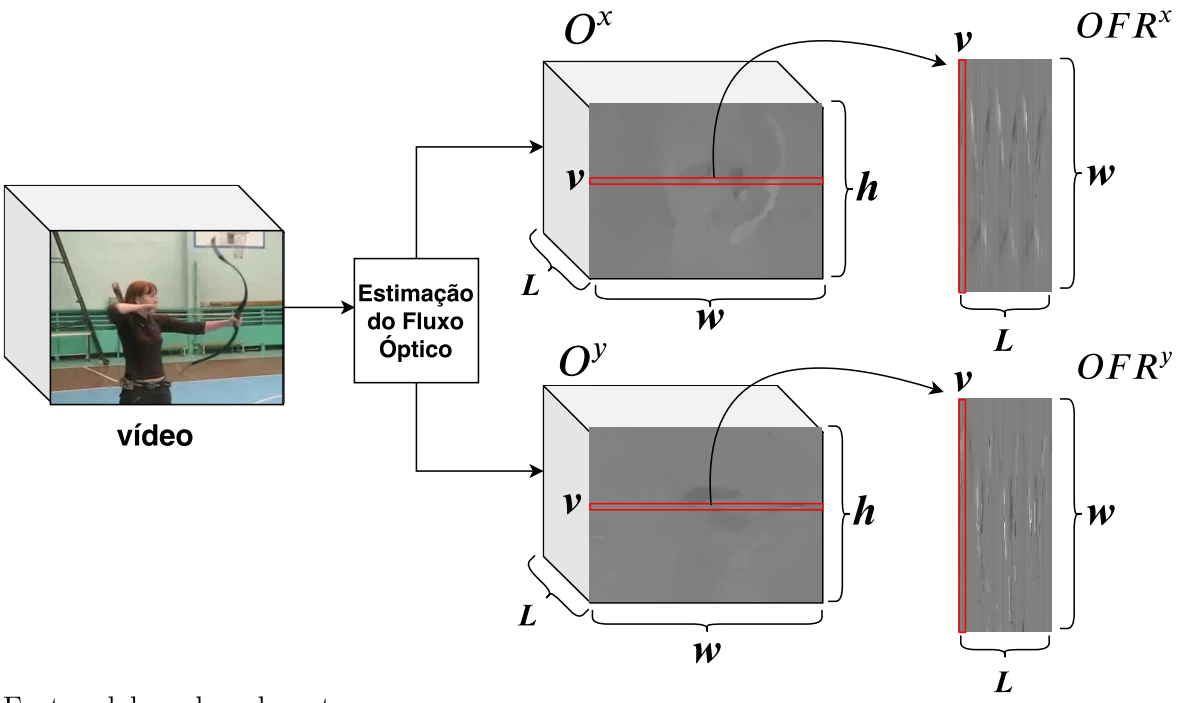
Seja $O = \{D_1, D_2, \ldots, D_L\}$ todo o volume de OFs obtidos de um vídeo V, onde cada $D_i(u,v) = [d_i^x(u,v) \;\; d_i^y(u,v)]^T$ representa o fluxo $h \times w$ estimado em um *frame* F_i, e $P = \{p_1, \cdots, p_n\}$ um conjunto de coordenadas 2D como definido na Equação (2.2). Cada elemento $D_i(p_j)$ é um vetor 2D de deslocamentos de fluxo óptico no ponto p_j do i-ésimo *frame*. Assim, a contribuição 1D desse i-ésimo *frame* é um vetor coluna $n \times 2$ denominado C_i, onde $C_i = [D_i(p_1) \; D_i(p_2) \; \cdots \; D_i(p_n)]^T$. Então, o OFR do vídeo V inteiro é o corte $n \times L \times 2$:

$$OFR_P = [C_1 \; C_2 \; \cdots \; C_L]. \tag{4.1}$$

Como ilustrado na Figura 11, essa nova informação funciona como um retrato das variações de velocidade de *pixels* específicos ao longo de todo o vídeo. Seu objetivo é fornecer mecanismos que permitam que modelos possam inferir padrões de ordem superior nos dados. Padrões que podem estar relacionados à aceleração da variação de brilho em algumas regiões do vídeo. Na Figura 11, por exemplo, os padrões característicos da movimentação do arco e das mãos do ator ao disparar a flecha ficam evidentes em ambas componentes do OFR.

Embora essas informações de mais alta ordem pareçam de suma importância para definir certas ações, são raros os métodos HAR que exploram esse tipo de informação (KONG; FU, 2018). Sendo esse um dos principais indicativos de que o OFR pode ser complementar às representações usadas nesse domínio. Complementaridade que possivelmente reflete em novos conhecimentos a serem agregados à arquitetura. Conhecimentos sobre aspectos até então não explorados nos dados.

Figura 11 – O ritmo do fluxo óptico é um recorte 2D retirado do volume de OFs de um vídeo inteiro. Note o padrão regular gerado em ambas componentes do OFR pela movimentação do arco e das mãos do ator ao disparar a flecha.



Fonte: elaborado pelo autor.

4.4 PROTOCOLO DE EXTRAÇÃO DOS RITMOS

Um ritmo pode ser definido como um plano recortado de um volume tridimensional de informação. Extrair um ritmo é o mesmo que comprimir um vídeo em uma representação 2D. O VR e o OFR são dois desses recortes extraídos de diferentes representações do vídeo, respectivamente, *frames* e OFs. Embora os trabalhos de Concha et al. (2018) e Tacon et al. (2019) reforcem as diversas vantagens desse tipo de representação, alguns desafios decorrem da utilização dessa *feature* no ambiente *multi-stream*. Definir a estratégia de extração dos ritmos é uma dessas questões.

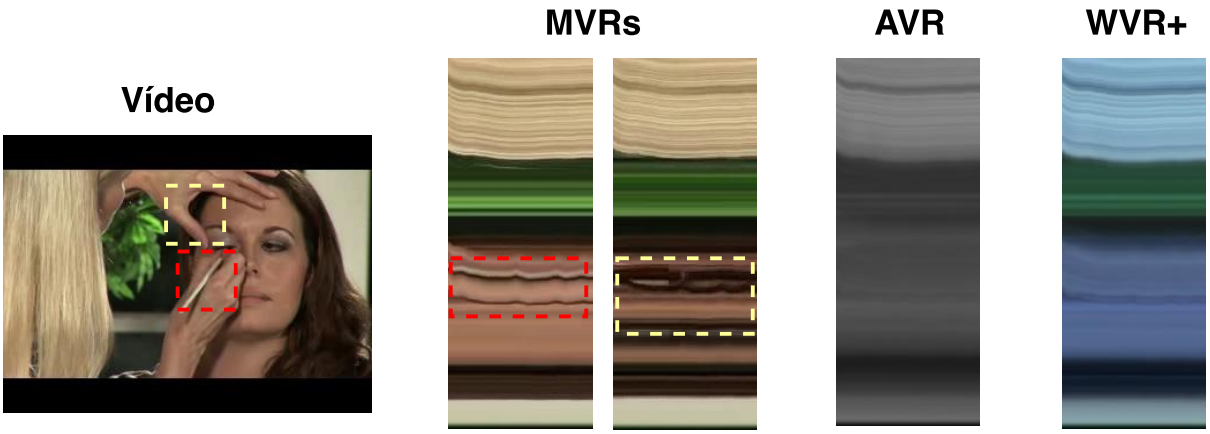
Concha et al. (2018) sugeriram que este recorte 2D fosse construído sob a média das linhas ou colunas de cada *frame*, de acordo com a direção usada em sua extração. Tacon et al. (2019) substituiu essa média por uma gaussiana centrada no meio dos vídeos sob a justificativa de que padrões de movimento poderiam ser perdidos, pesando igualmente

regiões distantes do objeto principal com regiões vizinhas. Tanto Concha et al. (2018) quanto Tacon et al. (2019) buscavam um único ritmo que sintetizasse toda a informação útil de um vídeo e que pudesse ser utilizado como *feature* espaço-temporal em uma arquitetura *multi-stream*.

Porém, por mais que exista uma predisposição das ações ocorrerem na parte central dos vídeos (TACON et al., 2019), é muito improvável que um único plano retenha toda a informação necessária para caracterizar uma ação. De fato, em um vídeo presume-se que haja movimento em diversas partes e direções. A adoção de uma estratégia tão brusca de compressão levaria à perda de muita informação, o que poderia prejudicar seriamente o processo de identificação das ações. Em condições ideais, toda a extensão do vídeo deveria ser analisada por meio de ritmos. E cada um desses planos deveria ser levado em conta para definir uma ação, o que seria extremamente ineficiente do ponto de vista computacional.

Aproximar esse efeito por meio de uma operação matemática, como nas soluções de Concha et al. (2018) e Tacon et al. (2019), também não parece ser o mais indicado, como demonstrado na Figura 12, onde fica evidente que os diferentes padrões relacionados à movimentação do lápis de olho (em vermelho), da mão e dos olhos do outro ator (em amarelo) não aparecem no VR médio e no VR gaussiano propostos por esses autores.

Figura 12 – Comparação entre os métodos de extração do VR. Padrões específicos deixados pela movimentação de alguns pontos, marcados em amarelo e vermelho no vídeo original e nas imagens do MVR, AVR e WVR^{+}gerados.



Fonte: elaborado pelo autor.

Ainda que um processo tão completo de extração não seja viável computacionalmente, é possível que uma versão sua mais simples possa ser aplicada. Nesse procedimento são prefixados: o número de planos e as direções utilizadas no processo de extração dos ritmos. Com base nessas ideias, neste trabalho, vários planos horizontais são amostrados do vídeo. Ritmos definidos por suas formulações clássicas (Equações (2.1) e (4.1)). Dessa

forma, um único vídeo passa a ser representado não por um, mas por vários ritmos. Com isso, mais aspectos do vídeo podem ser capturados e perdas decorrentes da definição de um único plano para representá-lo podem ser amenizadas. A ideia é que essas várias imagens de OFRs e VRs introduzam mais características dos dados analisados, enriquecendo o processo de treinamento e melhorando a taxa de reconhecimento das *streams*.

4.5 NOVA *STREAM* TEMPORAL (MOFR)

Modelar relações temporais de longo alcance nos vídeos é um dos passos fundamentais para o entendimento das dinâmicas associadas às ações humanas em vídeos. Ações mais complexas, por exemplo, tipicamente são executadas em múltiplos passos e podem durar centenas de *frames*. Definir esse tipo de ação, representada pela Figura 13, envolve uma análise do vídeo como um todo. Analisar localmente provavelmente levaria a subótimos, prejudicando o seu processo de reconhecimento (VAROL; LAPTEV; SCHMID, 2017). Por isso, existe uma preocupação recorrente na literatura em encontrar formas eficientes de explorar essas estruturas temporais de longo alcance, responsáveis por caracterizar estas ações (KONG; FU, 2018).

Figura 13 – Vários trechos retirados de um vídeo da ação "Ginástica de solo". Analisar isoladamente cada um desses trechos dificulta o processo de reconhecimento e pode levar a erros de classificação. O segundo e terceiro trechos, da esquerda para direita, podem, por exemplo, incorretamente definir a ação como "Salto em distância", uma vez que, essa mesma ação também envolve uma etapa de impulso para realização dos movimentos.

Fonte: elaborado pelo autor.

Diversas estratégias foram empregadas na tentativa de incorporar este aspecto ao domínio *multi-stream*, como pilhas maiores de fluxos ópticos, modelos temporais (ULLAH et al., 2018) ou novas *streams* (CONCHA et al., 2018; WANG et al., 2017). Parte dessas abordagens esbarram no alto custo computacional associado ou na dificuldade de representar corretamente esse aspecto temporal (KONG; FU, 2018), mostrando que esse problema continua desafiador.

Como a *stream* temporal usada na arquitetura proposta é capaz de modelar somente correlações temporais curtas nos vídeos, levando em consideração a taxa de amostragem e o tamanho da pilha usados nessa abordagem, uma stream temporal adicional é proposta neste trabalho. A MOFR ou *stream* dos múltiplos OFRs é uma *stream* criada para codificar

estas correlações temporais longas nos vídeos. Ao apresentar uma visão geral da variação de movimento de todo o vídeo, o OFR se torna uma *feature* capaz de modelar essas dinâmicas de movimento de longo alcance. A ideia não é substituir, mas sim complementar a *stream* temporal original.

Essa nova *stream* temporal está estruturada sob uma CNN similar a que foi utilizada na *stream* espacial. Como as informações manipuladas originalmente por essa rede eram *frames* em formato RGB, uma pequena alteração foi realizada nas imagens do OFR para adequá-las à entrada dessa CNN. Um canal adicional, preenchido com zeros, foi concatenado aos dois canais (d^x e d^y) referentes à contribuição de cada *frame* ao volume de OFs. Já que todas as amostras terão esse mesmo canal extra, espera-se que a rede aprenda a ignorá-lo, focando as análises nos dois canais que realmente contêm informação.

O processamento dessa *stream* consiste de um passo inicial onde são estimadas as imagens dos OFs. Em seguida, esse volume de OFs serve de base para a extração de diversos OFRs, assim como especificado na Seção 4.4. Todos esses planos gerados são fornecidos um a um para a CNN que definirá a ação sendo realizada por meio desses elementos. O protocolo de treinamento e teste empregado nessa *stream* está descrito em detalhes na Seção 4.7.

4.6 *STREAM* ESPAÇO-TEMPORAL (MVR)

A abordagem de Concha et al. (2018) usa um ritmo visual adaptativo médio como a entrada da CNN, que consiste em estimar a direção com a maior quantidade de movimento e extrair um corte perpendicular a essa direção. Os autores também propuseram que esse corte fosse construído sobre a média de todas as linhas ou colunas dos *frames*. Abordar o problema dessa forma tem sérias implicações como descrito na Seção 4.4.

No presente trabalho, uma adaptação foi feita na *stream* proposta por Concha et al. (2018). Ao invés de um único VR para representar um vídeo, é proposta a utilização de múltiplos VRs extraídos de vários planos do vídeo. A premissa é a de que múltiplos planos podem retratar melhor as complexas interações relacionadas às ações humanas.

4.7 PROTOCOLO DE CLASSIFICAÇÃO MVR E MOFR

Ainda hoje, são raros os modelos que trabalham diretamente sobre os vídeos em sua forma bruta. Entre os fatores que restringem o uso desse tipo de representação está a sua alta dimensionalidade. Ainda que existam algumas arquiteturas que consumam essa *feature* como entrada, em geral, a quantidade de informação agregada pela adição de uma dimensão extra, inviabiliza o amplo uso desse tipo de abordagem (KONG; FU, 2018). Além disso, vale a pena salientar que vídeos não possuem a dimensão temporal de tamanho fixo. De fato, eles podem ter um número arbitrário de *frames* podendo ser

mais curtos ou mais longos dependendo do tipo de ação, dos atores ou de vários outros fatores. Entretanto, certos modelos, como CNNs 3D, têm como pré-requisito dimensões de tamanho fixo, fazendo com que certas adaptações, como subamostragens, tenham que ser realizadas para encaixar as dimensões do vídeo à entrada dessas arquiteturas. Isso pode afetar o *frame rate* e distorcer os vídeos a ponto das ações ficarem irreconhecíveis, conforme apresentado por Tacon et al. (2019).

O procedimento de classificação adotado nas duas novas *streams*, MVR e MOFR, consiste de um protocolo próprio de treinamento e teste inspirado nas ideias apresentadas por Simonyan e Zisserman (2014a), inicialmente, para contornar as dificuldades introduzidas pelo uso do vídeo bruto nesse domínio. No protocolo original, os autores recomendam que, ao invés de analisar o vídeo inteiro de uma só vez, que o mesmo seja particionado em vários blocos que são processados individualmente. Com isso, o esforço computacional necessário para processar os vídeos é reduzido, uma vez que esses microvolumes podem ter um tamanho definido de acordo com os recursos disponíveis ou com as dimensões de entrada de determinado modelo. Esse mesmo procedimento também pode ser utilizado para aumentar a capacidade de generalização de um modelo, já que cada um desses fragmentos pode conter um aspecto bem específico de uma determinada ação que pode ser útil no seu processo de identificação (KONG; FU, 2018).

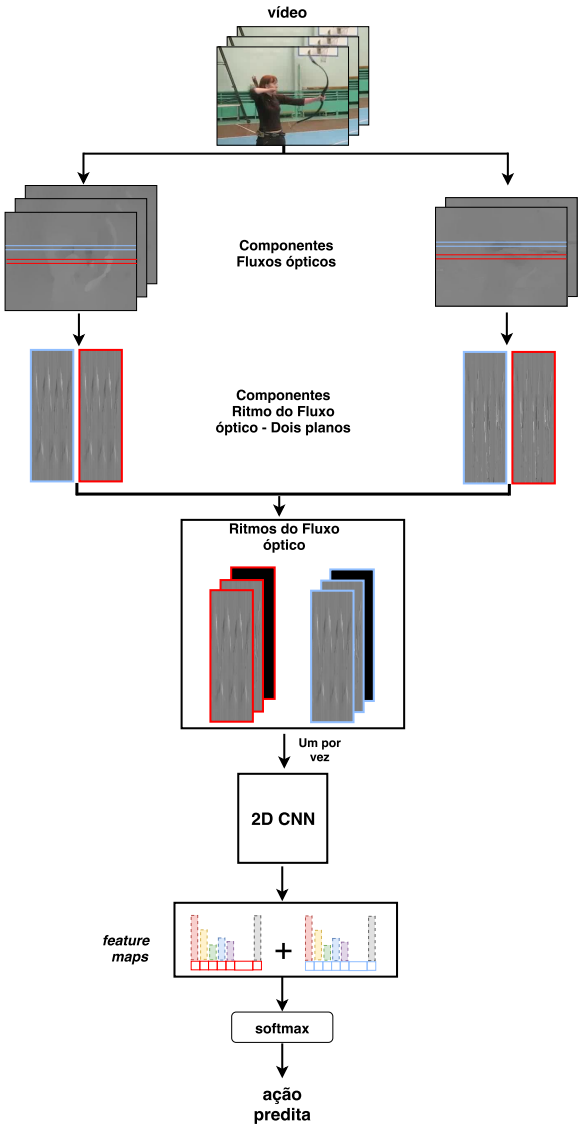
No treinamento adotado por Simonyan e Zisserman (2014a), cada um desses blocos de informação é considerado uma nova amostra. Blocos de informação que podem ser utilizados todos ao mesmo tempo para definir uma ação no caso de um vídeo de teste. Os autores usam esse procedimento para cobrir e analisar toda a dimensão temporal de um vídeo. No presente trabalho, esses blocos de informação são planos dos ritmos visuais e do fluxo óptico. O tempo é uma dimensão que já está inerentemente incluída em cada um desses blocos, e que consequentemente será levada em consideração pela rede neural. Então, o protocolo projetado fica encarregado de explorar outros aspectos do vídeo, como certas assinaturas espaciais e espaço-temporais não capturadas pelas outras *streams* da arquitetura, por exemplo.

Durante o treinamento da MVR ou da MOFR, múltiplos planos uniformemente extraídos das linhas do vídeo são utilizados. Cada plano é um ritmo processado e posteriormente classificado como uma nova amostra pela CNN. Na prática, usa-se a expressão invariante à altura $u = \alpha_u \cdot h$, com $0 \leq \alpha_u \leq 1$, para extrair o ritmo da linha u. Com base em exaustivos experimentos e nas conclusões de Tacon et al. (2019), assume-se que as ações tendem a ocorrer na parte central do vídeo. Com isso, o processo de extração dos planos é restrito ao intervalo $0,4 \leq \alpha_u \leq 0,6$.

No protocolo de teste, vários ritmos são uniformemente extraídos do mesmo intervalo usado no treinamento. Em seguida, é calculada a média de todos os *feature maps* obtidos em cada um desses ritmos. Esses *feature maps* são extraídos da camada que antecede a

softmax em cada uma das redes usadas. Os experimentos realizados demonstraram que essa estratégia de fundir os *feature maps* antes da normalização do softmax apresentava melhores resultados. A função de ativação softmax então é aplicada a esse *feature map* médio que finalmente é usado para prever a classe da amostra. Conjectura-se que esse protocolo de treinamento/teste pode produzir melhores predições com base na suposição de que os múltiplos ritmos podem conter partes distintas da ação sendo realizada. A Figura 14 apresenta uma visão geral do protocolo proposto aplicado na nova *stream* temporal.

Figura 14 – Uma visão geral do protocolo de classificação adotado na *stream* MOFR. Inicialmente são estimadas as componentes do OF. Dois planos, representados pelas linhas azul e vermelha, são então extraídos desse volume de OFs, gerando os OFRs correspondentes. Todos os OFRs são submetidos um a um para CNN. As *features* resultantes passam por uma média, e a predição final é dada por meio de uma camada softmax.



Fonte: elaborado pelo autor.

4.8 MÉTODO DE FUSÃO

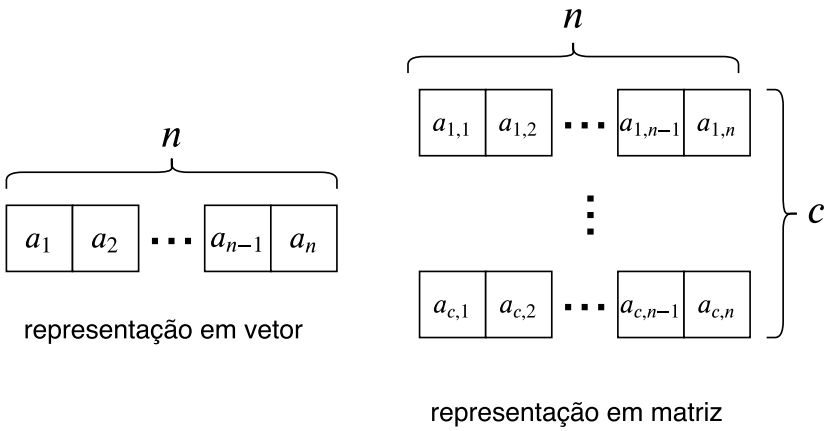
Um método de fusão tardia (*late fusion*) por média ponderada foi implementado para combinar as várias *features* da arquitetura *multi-stream* desenvolvida. Nesse esquema, a predição final de uma amostra é dada ponderando-se a saída de cada *stream*. Para determinar esses pesos, propõe-se o uso da meta-heurística *simulated annealing* – SA. O SA é uma abordagem alternativa às técnicas exaustivas geralmente aplicadas ao domínio estudado, que se mostraram ineficientes, principalmente, com o surgimento de técnicas de DL. A busca em grade (*grid search*), um dos métodos mais usados com essa finalidade, executa uma busca por força bruta no espaço dos hiperparâmetros. Como esse espaço de busca pode incluir valores reais, uma etapa de pré-processamento para definir manualmente limites e discretizar esse espaço pode ser necessária, antes de aplicar um *grid search*. A natureza dessa busca torna este processo mais suscetível à maldição da dimensionalidade, fazendo com que o espaço de soluções analisado cresça exponencialmente com o número de hiperparâmetros utilizados, sem nenhuma garantia de que a melhor solução será encontrada.

Meta-heurísticas, por outro lado, fornecem mecanismos para realizar essa busca de maneira inteligente. Ao utilizar uma função a ser otimizada como uma espécie de guia, uma meta-heurística percorre sistematicamente o espaço de busca, evitando regiões com um grande número de soluções homogêneas ou próximo a muitas soluções ótimas locais (OSMAN; LAPORTE, 1996). Essa redução no espaço de busca pode tornar o processo de ajuste de hiperparâmetros viável mesmo em situações onde um número considerável de parâmetros precise ser otimizado.

Estabelecido o SA como meta-heurística responsável por otimizar os parâmetros do método de fusão proposto, uma série de questões referentes à modelagem do problema de ajuste de hiperparâmetros precisam ser definidas. Determinar uma representação para soluções manipuladas pelo SA, um conjunto de operadores para modificá-las e uma função a ser otimizada que tenha relação com o problema, são os principais pontos a serem decididos. No âmbito do presente trabalho, duas representações distintas para as soluções foram usadas: uma matriz e um vetor de pesos. A representação em vetor é a representação usada nos trabalhos de Concha et al. (2018) e Tacon et al. (2019), onde cada elemento do vetor refere-se ao peso atribuído à saída de uma *stream* da arquitetura. Na matriz, estende-se esse conceito e pesa-se a contribuição que cada classe de uma determinada *stream* terá na ponderação final. Tratar esse problema a nível de classe tem a vantagem de que os ajustes podem ser mais finos, uma vez que, é natural supor que o desempenho das *streams* pode variar de acordo com a classe analisada, um comportamento que não seria possível de ser capturado pesando igualmente todas as classes. Entretanto, vale salientar que esses mesmos ajustes podem agravar um possível *overfitting* sem um método de regularização adequado (WU et al., 2015). A Figura 15 ilustra estas duas representações.

Cada elemento a_i de uma dessas representações deve estar restrito ao intervalo

Figura 15 – Conjunto de pesos, uma solução do SA, representado como um vetor e como uma matriz, respectivamente.



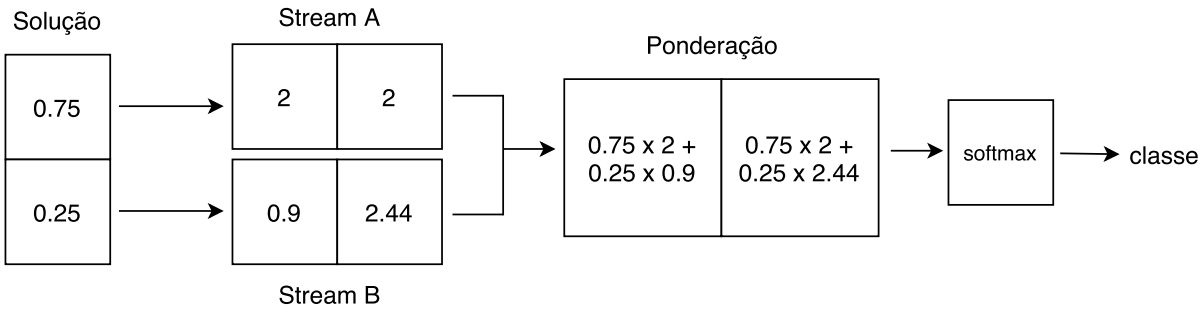
Fonte: elaborado pelo autor.

$0 \leq a_i \leq 1$, de modo que as soluções obtidas, usando uma arquitetura de n *streams*, estejam condicionadas à restrição $\sum_{i=0}^{n} a_i = 1$ em cada linha da representação em matriz, ou no vetor inteiro, no caso da ponderação onde esse tipo de representação é utilizado. Essa restrição é garantida pelos operadores implementados, conforme ilustrado no Algoritmo 1 (linhas 16 e 17). Isso reduz ainda mais o espaço de busca do problema, evitando a avaliação de soluções que sejam múltiplos umas das outras, isto é, soluções que fornecerão o mesmo resultado se aplicadas ao método de fusão proposto.

O método desenvolvido para calibrar os pesos da fusão funciona da seguinte forma: inicialmente, é atribuído um mesmo peso a todas as *streams*. Essa configuração corresponde à fusão por média simples descrita na Subseção 3.3.4. Partindo dessa solução inicial, o SA, a cada iteração, seleciona duas *streams* e promove uma transferência de pesos entre elas. Na abordagem que usa a matriz de pesos, esse processo é realizado em uma das linhas da matriz, correspondente a uma das C classes na base de dados, também selecionada de forma aleatória. Usando essa nova configuração de pesos, calcula-se o valor da função objetivo ponderando as saídas de cada *stream* com os seus respectivos pesos, assim como representado na Figura 16. Esse cálculo é realizado usando como base um certo conjunto de amostras de vídeo. Se houver uma melhoria, a solução corrente é atualizada. Caso contrário, uma diminuição é aceita condicionalmente de acordo com uma probabilidade $p = e^{\Delta_{acc}/T}$. O Algoritmo 1 descreve o método proposto.

Como função objetivo do SA, define-se a seguinte expressão $f = acc - \beta * reg$, onde acc corresponde ao valor de acurácia obtido, usando o conjunto de pesos da solução corrente e reg é um termo de regularização derivado de outra medida obtida neste mesmo conjunto. Nos testes foram feitas tentativas com a acurácia da pior classe, o desvio padrão do valor de acurácia por classe, e medidas de clusterização como o coeficiente de silhueta,

Figura 16 – Um exemplo da aplicação de uma solução provida pelo SA na predição de uma amostra em uma arquitetura *two-stream*. Note os pesos associados à cada uma das *streams* dessa arquitetura. 0.75 para *stream* A e 0.25 para os resultados fornecidos pelo *stream* B. A função softmax é aplicada ao resultado dessa ponderação para prever a classe da amostra em questão.



Fonte: elaborado pelo autor.

como mecanismos de regularização. A constante β determina o quão forte é esse fator regularizador. Para evitar qualquer viés nos experimentos, propõe-se que essas medidas sejam extraídas de um 3-*fold* realizado no conjunto de treinamento. Assumindo que o conjunto de treinamento forneça condições suficientes para criar um modelo com um alto poder de generalização, espera-se que os pesos das *streams*, otimizados neste conjunto, possam ser estendidos com consideráveis chances de sucesso para o conjunto de teste.

Algoritmo 1: *Simulated annealing* para fusão *multi-stream*

Entrada: temperatura inicial T_0;
taxa de decréscimo da temperatura α;
número de *streams* n;
número de classes c;
tipo de representação de solução utilizada t;
Saída: melhor solução s^*;

início
  $s_0 \leftarrow a_{c,n} = \frac{1}{n}$ ou $s_0 \leftarrow a_n = \frac{1}{n}$; `// todas as streams com o mesmo peso, respectivamente, na abordagem com matriz e vetor.`
  $T \leftarrow T_0$;
  $s \leftarrow s_0$;
  $s^* \leftarrow s$;
  enquanto $T \geq T_{min}$ **faça**
    repita
      $s^{'} \leftarrow s$;
      se $t =$ *matriz* **então**
        selecione aleatoriamente uma linha j da matriz em $s^{'}$;
        selecione duas posições aleatórias (p_1, p_2) em $s^{'}_j$;
      senão
        selecione duas posições aleatórias (p_1, p_2) em $s^{'}$;
      fim se
      selecione um valor v randomicamente no intervalo $(0 - s[p_2])$;
      $s^{'}[p_1] \leftarrow \min(1, s^{'}[p_1] + v)$;
      $s^{'}[p_2] \leftarrow \max(0, s^{'}[p_2] - v)$;
      $\Delta_f \leftarrow f(s') - f(s)$;
      se $\Delta_f \geq 0$ **então**
        $s \leftarrow s^{'}$;
      senão
        $s \leftarrow s^{'}$ com uma probabilidade $e^{\ \Delta_{acc}/T}$;
      fim se
      se $f(s') > f(s^*)$ **então**
        $s^* \leftarrow s^{'}$;
      fim se
    até *o equilíbrio térmico em* T *não for alcançado*;
    $T \leftarrow \alpha\, T$;
  fim enquanto
fim

5 EXPERIMENTOS E RESULTADOS

Neste capítulo são apresentados os experimentos conduzidos para avaliar o método proposto. Inicialmente é apresentada uma descrição geral das bases de dados utilizadas e dos detalhes de implementação. No restante do capítulo, estão os experimentos realizados para avaliar os vários aspectos da arquitetura proposta nos mais diversos cenários.

5.1 BASES DE DADOS

Para avaliar o método proposto foram utilizadas duas bases de dados (*datasets*) de ações consideradas desafiadoras: a UCF101 (SOOMRO; ZAMIR; SHAH, 2012) e a HMDB51 (KUEHNE et al., 2013).

O *dataset* UCF101 contém 13.320 vídeos, coletados do YouTube, distribuídos em 101 classes. Os vídeos da UCF101 tem uma duração média de 7 segundos e foram amostrados a uma taxa de *frames* fixa de 25 FPS e resolução de 320×240 *pixels*. As ações retratadas nas amostras de vídeo da UCF101 podem ser divididas em 5 grandes grupos, representando possíveis interações realizadas por humanos ao executar uma ação: interações humano-objeto, humano-humano, movimentos corporais, tocar instrumentos musicais e praticar esportes. É uma base de dados que abrange os principais desafios no reconhecimento de ações, como variações no movimento da câmera, na escala do objeto, de ponto de vista, desordem de fundo, condições de iluminação adversas, entre outros. Essa diversidade é crucial para fornecer um panorama mais confiável do desempenho do modelo em um cenário mais próximo do mundo real.

A HMDB51 possui 51 categorias de ações, cada uma contendo pelo menos 101 clipes, totalizando 6.766 vídeos tirados de várias fontes, principalmente de filmes, e uma pequena parte de bancos de dados públicos. Como os clipes extraídos originalmente da HMDB51 podem variar em tamanho e *frame rate*, a altura de todos os quadros foi redimensionada para 240 *pixels*. A largura foi ajustada de modo a manter a proporção de tela original. Todos os clipes tiveram seu *frame rate* convertido para 30 FPS. Como na UCF101, as ações da HMDB51 podem ser agrupadas em cinco tipos: ações faciais gerais, ações faciais com manipulação de objetos, movimentos corporais gerais, movimentos corporais com manipulação de objetos e movimentos corporais para interação humana. Dentre essas amostras incluem-se vídeos borrados ou de qualidade inferior e ações filmadas em diferentes pontos de vista.

O protocolo de avaliação usado nos dois conjuntos de dados é o mesmo. Os autores fornecem uma divisão recomendada das amostras (*splits*), 3 no total. Cada *split* mantém uma proporção de 70/30% das amostras sendo usadas, respectivamente, para treinamento e teste por classe de ação. A média simples da acurácia dos três *splits* de treinamento/teste disponíveis em cada *dataset* é reportada como o resultado final.

5.2 DETALHES DE IMPLEMENTAÇÃO

Todos os códigos foram desenvolvidos na linguagem de programação Python. Os códigos de extração das *features*, assim como o algoritmo SA, foram executados em CPU por uma máquina com um processador Intel® Xeon® E5-4607 com 48 *threads* e 32GB de memória RAM. Os treinamentos e inferências foram todos realizados em GPU para tirar vantagem do poder de processamento desses equipamentos. Nesta etapa, todos os cálculos foram processados por uma máquina com processador Intel® Core® i7-6700 com 8 *threads*, 16GB de memória RAM e uma placa de vídeo NVIDIA® Titan Xp.

No processo de geração dos vários OFRs e VRs usados como entrada das *streams* MVR e MOFR, respectivamente, foram utilizados os *frames* e OFs calculados a partir do vídeo. Essas informações são obtidas por meio do *framework* ffmpeg, encarregado de extrair os *frames* e da biblioteca OpenCV, ou mais precisamente da sua implementação do algoritmo TVL1 (ZACH; POCK; BISCHOF, 2007), responsável por estimar os fluxos ópticos. Os parâmetros usados nesse processo de extração estão descritos no trabalho de Concha et al. (2018).

As linhas de referência retiradas de cada *frame* ou imagem de OF são concatenadas verticalmente para formar o arranjo 3D que representa, respectivamente, os VRs e OFRs dos vídeos. O número de VRs e OFRs coletados por vídeo, durante o protocolo descrito na Seção 4.4, foi estimado empiricamente como resultado de um dos experimentos realizados. Esse experimento indicou que 10 planos amostrados durante o treinamento e 5 na etapa de teste são os valores que melhor balanceiam custo computacional e desempenho. Os ritmos, VRs e OFRs, assim como os OFs, são calculados em um estágio prévio ao treinamento.

Para as *streams* espacial e temporal, foi usada a implementação em Pytorch fornecida no *framework two-stream* para reconhecimento de ações em vídeos de Zhu et al. (2020). Foram mantidos os mesmos parâmetros descritos por Wang et al. (2015) no processo de treinamento dessas *streams*.

Nas *streams* MVR e MOFR, optou-se por uma versão da InceptionV3 (SZEGEDY et al., 2016), inicializada com os pesos da Imagenet (DENG et al., 2009) e implementada no *framework* Keras (CHOLLET et al., 2015). Algumas técnicas de aumento de dados do Keras foram aplicadas aos dados como *flips* horizontais e verticais, e *zooms* no intervalo de 0,8 a 1,2. Os pesos da rede foram treinados com os seguintes parâmetros: tamanho do *batch* 16 e taxa de aprendizado inicial $1e^{-3}$. Foi definida uma taxa de aprendizado ajustável, reduzida por um fator de 10 depois de 3 épocas sem qualquer melhora na função de custo, respeitando um limite inferior de $1e^{-6}$. O processo de treinamento é conduzido por um otimizador de gradiente descendente estocástico simples (*stochastic gradient descent* – SGD), com *momentum* de $0,9$ e função de entropia cruzada categórica (*categorical cross entropy loss*) como função de custo. Dois critérios de parada foram

definidos para o treinamento: um número fixo de 100 épocas ou uma estratégia de parada antecipada (*early stopping*) de 6 épocas. Ambos os conjuntos de dados possuem esses mesmos parâmetros de treinamento.

O algoritmo padrão do SA apresentado por Kirkpatrick, Gelatt e Vecchi (1983) foi escolhido para compor o método de fusão proposto. Nessa versão clássica, o equilíbrio térmico é dado por um número de iterações do algoritmo sem melhora no valor da função objetivo, e uma temperatura mínima T_{min} determina quando o sistema "cristaliza". Esse ponto de cristalização do sistema define o fim da execução do algoritmo. Os parâmetros usados no SA, conforme apresentado no Algoritmo 1 do Capítulo 4, foram obtidos a partir de exaustivos testes empíricos: $T_0 = 0{,}0001$, $T_{min} = 0{,}0000008$, $iter_{max} = 100$, $\alpha = 0{,}95$.

Conforme informado na Seção 4.8, a função objetivo usada no SA é calculada com base em um 3-*fold* estratificado no conjunto de treinamento. Para realizar esse processo, foi mantida a mesma distribuição de dados encontrada na divisão em *splits* da UCF101 e HMDB51 reportada na Seção 5.1. Assim, cada *fold* tem aproximadamente 2/3 e 1/3 das pessoas (amostras), respectivamente, em seus conjuntos de treinamento e validação.

5.3 RESULTADOS

Esta seção compreende o resultados de vários experimentos realizados para avaliar o método proposto. Experimentos realizados para definir parâmetros sensíveis na construção do método, para mensurar a contribuição das *streams* introduzidas e, por fim, uma comparação com as principais arquiteturas *multi-stream* da literatura.

5.3.1 **Definição do número de planos**

O protocolo descrito na Seção 4.4 leva em consideração que um certo número de planos (ritmos) é extraído de cada vídeo para formar o MVR e o MOFR propostos. Esse é um dos parâmetros críticos da abordagem, uma vez que muitos planos podem tornar o processo inviável computacionalmente, enquanto poucos planos podem fazer com que a arquitetura falhe em capturar corretamente os diversos aspectos do vídeo. Então, o objetivo desse experimento é tentar estimar um número que considere esses dois pontos, ou seja, um valor que equilibre eficácia e eficiência (custo computacional).

O número de planos utilizados durante o processo de treinamento foi cuidadosamente escolhido a partir de experimentos prévios. O número estimado como resultado desses experimentos foi de 10 planos. Com isso, a base de dados é incrementada por um fator de 10, uma quantidade de dados viável de ser processada usando o ambiente de treinamento descrito na Seção 5.2 e que proporciona um incremento no desempenho da arquitetura. Nesse experimento prévio, variando o número de planos usado no treinamento de 1 para 10, o valor da acurácia de teste obtida a partir de um plano central de cada vídeo, subiu

de 64,85% para 69,07% na UCF101 e de 34,34% para 39,52% na HMDB51. Um número superior a 10 planos, como 20, acrescentaria muita complexidade ao processo sem um incremento considerável no resultado final. Com 20 planos, o valor da acurácia de teste foi de 69,02% na UCF101 e de 39,99% na HMDB51. Além disso, quanto mais ritmos forem usados, maiores são as chances de sobreposição entre os planos extraídos o que, em geral, não é interessante já que um possível viés pode aparecer nessas regiões onde vários planos estão sobrepostos.

Fixado o número de planos extraídos durante a etapa de treinamento, uma segunda etapa que envolve o cálculo do número ótimo de ritmos utilizados no protocolo de teste é realizada. A Tabela 1 sintetiza o resultado desse experimento.

Tabela 1 – Acurácia obtida usando um número variado de planos extraídos durante a etapa de treinamento na UCF101 e na HMDB51.

Nº de planos	UCF101		HMDB51	
	MVR	MOFR	MVR	MOFR
1 (Central)	69,07	67,04	39,52	42,00
2	72,28	69,38	40,52	42,70
3	73,98	70,81	41,94	44,38
5	74,54	71,05	42,20	44,75
10	74,77	71,30	43,03	44,90
20	74,87	71,44	42,94	45,23
40	74,91	71,41	43,09	44,98

Fonte: elaborado pelo autor.

Analisando os resultados da Tabela 1, observa-se que os valores reportados de número de planos e acurácia estão correlacionados positivamente. Um comportamento já esperado, uma vez que, presumia-se que mais planos podiam retratar melhor as complexas interações associadas às ações humanas. Entretanto, um número maior de planos pode impactar num aumento considerável do tempo de inferência destes modelos, provocando um gargalo na arquitetura durante a etapa de teste. Foi definido então que 5 planos é o número que melhor equilibra custo computacional e eficácia. O desempenho do modelo com 5 planos é bem próximo ao de 10 planos, com a diferença de que o processo de extração com 10 planos leva quase o dobro do tempo do que usa menos planos para construir a representação utilizada para inferir as ações. O tempo médio de extração de todas as amostras do conjunto de teste da UCF101, que inclui quase 4000 vídeos, diminui de 47 minutos para 25 minutos usando 5 planos ao invés dos 10 propostos inicialmente.

5.3.2 **Desempenho individual das *streams***

Buscando mensurar os limites e impactos que cada *stream* tem para o resultado final da arquitetura gerada, é feita uma análise comparativa do desempenho individual

das várias *streams* que a compõem. Com esse experimento, almeja-se uma forma de esclarecer a importância de cada fluxo no processo de reconhecimento de ações realizado por uma arquitetura dessa natureza. As Tabelas 2 e 3 resumem os resultados obtidos nesse experimento.

Tabela 2 – Resultados da acurácia de cada *stream* no *dataset* UCF101.

Streams individuais	*Split* 1	*Split* 2	*Split* 3	Média
Imagens RGB* (CONCHA et al., 2018)	86,70	86,48	86,58	86,59
Fluxo óptico (CONCHA et al., 2018)	86,02	87,47	87,36	**86,95**
MOFR	68,13	73,33	71,70	71,05
Horizontal - média (CONCHA et al., 2018)	61,93	62,02	63,15	62,37
Vertical - média (CONCHA et al., 2018)	55,18	57,17	53,12	55,16
AVR (CONCHA et al., 2018)	65,24	63,85	65,12	64,74
WVR^+ (TACON et al., 2019)	–	–	–	68,01
MVR^1	69,50	68,37	69,35	**69,07**
MVR	74,02	74,08	75,51	**74,54**

Fonte: elaborado pelo autor.

Tabela 3 – Resultados da acurácia de cada *stream* no *dataset* HMDB51.

Streams individuais	*Split* 1	*Split* 2	*Split* 3	Média
Imagens RGB* (CONCHA et al., 2018)	54,58	51,37	49,35	51,77
Fluxo óptico (CONCHA et al., 2018)	59,67	60,52	59,54	**59,91**
MOFR	45,03	45,29	43,92	44,75
Horizontal - média (CONCHA et al., 2018)	36,15	34,92	35,65	35,57
Vertical - média (CONCHA et al., 2018)	29,38	31,24	30,18	30,27
AVR (CONCHA et al., 2018)	39,35	39,08	40,46	39,63
WVR^+ (TACON et al., 2019)	–	–	–	35,29
MVR^1	39,54	40,26	38,76	39,52
MVR	43,53	42,29	40,78	**42,20**

Fonte: elaborado pelo autor.

As abordagens RGB*, OF e as três primeiras *streams* espaço-temporais apresentadas nas tabelas (horizontal-média, vertical-média e AVR) são contribuições de Concha et al. (2018). Como o trabalho de Tacon et al. (2019) evolui a ideia original de *stream* espaço-temporal baseada em VRs proposta por Concha et al. (2018), decidiu-se que esse trabalho também deveria compor a seleção de métodos destacados como literatura comparativa nos experimentos. É importante ainda destacar que, embora o uso do OF como *stream* temporal não seja uma contribuição de Concha et al. (2018), o resultado de tal trabalho é mencionado porque ele introduz a InceptionV3 como CNN temporal em arquiteturas *multi-streams*.

As outras duas abordagens, MVR e MOFR, respectivamente, ritmo visual e ritmo do fluxo óptico extraídos de múltiplos planos são as contribuições deste trabalho. É possível notar que essas duas abordagens apresentaram valores de acurácia próximos, com uma ligeira vantagem para cada uma, de acordo com o conjunto de dados utilizado. Visando uma maior completude ao experimento, uma segunda *stream* espaço-temporal, MVR^1, foi adicionada, como uma espécie de *baseline*. O MVR^1 possui a mesma estratégia de treinamento do MVR com a diferença de que somente um único plano central é utilizado durante a etapa de teste.

Conforme relatado nos trabalhos de Simonyan e Zisserman (2014a) e Wang et al. (2016), o fluxo óptico apresenta os melhores resultados em ambos os conjuntos de dados, seguido do RGB*, MVR, MOFR, WVR$^+$ e das *features* espaço-temporais de Concha et al. (2018) (médias horizontal e vertical e o AVR).

Vale a pena ressaltar que a abordagem MVR proposta supera todas as outras *streams* espaço-temporais analisadas, destacadas na subdivisão inferior das tabelas. Assume-se que esse comportamento esteja relacionado ao fato de que múltiplos ritmos visuais podem capturar melhor a trajetória de um objeto em movimento em uma cena de vídeo. Isso fica evidente na comparação direta entre MVR e MVR^1. Mesmo que a estratégia de treinamento empregada em ambos os métodos seja idêntica, o fato do MVR usar mais planos para representar uma única amostra durante a etapa de teste dá a ele uma vantagem de quase $5,50\%$ na UCF101 e perto dos $2,70\%$ na HMDB51. Um fator que deixa os resultados do MVR^1 bem similares aos reportados, por exemplo, no AVR e no WVR*, e que corrobora com a suposição de que vários planos são cruciais para o entendimento de certas ações nas bases.

5.3.3 Comparação entre as representações de solução usadas no SA

Dois tipos de estruturas foram utilizadas para representar o conjunto de pesos manipulados como soluções da meta-heurística SA. A primeira representação, um vetor de pesos, corresponde a forma mais comum de ponderação dentro do ambiente HAR. Nessa estratégia é atribuído um único peso para todas as predições de uma certa *stream* da arquitetura. A outra representação, uma matriz de pesos, estende essa ideia de ponderar as contribuições das *streams* à nível de classes. Com base nesses conceitos, cada classe de uma determinada *stream* tem um peso associado. Embora ponderar por meio da matriz pareça ser o mais indicado, uma série de problemas decorrem do uso da matriz de pesos no domínio HAR, como apresentado por Wu et al. (2015). Os ajustes mais finos possíveis de serem realizados nessa segunda representação, por exemplo, podem agravar um possível *overfitting* nas *streams*, sem um método de regularização adequado.

Visando estes pontos, é proposto um experimento comparativo entre estas duas representações de solução. Para tornar essa análise ainda mais precisa, define-se que ambas

as representações serão comparadas usando os vários métodos de regularização descritos na Seção 4.8, em uma tentativa de reduzir o impacto de um possível *overfitting*, que prejudique os resultados da matriz de pesos. É importante mencionar que, o coeficiente de silhueta teve que ser normalizado para o mesmo intervalo dos valores da acurácia (0-1), para garantir que nenhum peso adicional, não desejado, fosse atribuído a uma das variáveis da função objetivo do SA. Usando o coeficiente de silhueta para medir a coesão e separação dos resultados, pretende-se mensurar o poder de generalização da arquitetura, usando os conjunto de pesos estimado em cada um dos cenários abordados na Tabela 4. Para que isso fosse possível de ser realizado, foi necessário que os resultados preditos pela arquitetura fossem convertidos em grupos, simulando o resultado de um procedimento de clusterização. Esses "grupos" então seriam utilizados para calcular o coeficiente de silhueta da clusterização assim como proposto por Rousseeuw (1987).

O mesmo protocolo apresentado na Seção 5.1 foi aplicado nesse experimento. Nesses testes, o SA foi executado 10 vezes em cada um dos cenários apresentados. Os valores finais reportados para a arquitetura *four-stream* proposta, descritos na Tabela 4, são as médias dessas execuções.

Tabela 4 – Comparação entre as duas representações (Rep.) de soluções no *dataset* UCF101. Métodos de regularização (Reg.) pela classe com a menor acurácia (Min), pelo valor do desvio padrão das classes (Std) e usando o coeficiente de silhueta (Sil). O valor da constante de regularização β também foi variado durante esse teste. Foi reportado também o resultado sem nenhum tipo de regularização.

Rep.	Reg.	β	Acc. Treinamento	Acc. Teste
Vetor	–	–	**93,37**	**94,84**
	Min	1	93,02	94,44
		2	92,89	94,25
	Std	1	92,99	94,53
		2	92,85	94,26
	Sil	1	92,29	93,84
		2	92,54	93,72
Matriz	–	–	95,81	93,50
	Min	1	95,77	**93,89**
		2	95,05	93,72
	Std	1	**95,92**	93,07
		2	95,05	92,90
	Sil	1	94,00	91,98
		2	93,93	91,94

Fonte: elaborado pelo autor.

A superioridade da abordagem que utiliza o vetor de pesos ficou evidente nesse

experimento. Mesmo que o valores de acurácia de validação tenham sido superiores nas abordagens que usam a matriz de pesos, o provável *overfitting* nessa abordagem, fez com que as altas taxas de acurácia alcançadas neste conjunto não fossem extrapoladas aos vídeos de teste utilizados, deixando um gap de valores considerável entre estes dois conjuntos. Além disso, uma análise dos resultados descritos na Tabela 4 também demonstrou que nenhum dos métodos de regularização propostos foi efetivo em conter o enviesamento da matriz de pesos ao conjunto utilizado no processo de otimização. Embora os resultados desse tipo de abordagem não tenham sido positivos, acredita-se que essa é uma das formas mais promissoras de realizar essa ponderação. Estudos mais aprofundados terão que ser realizados em trabalhos futuros para confirmar essas suposições e propor novos mecanismos, mais eficientes, para controlar o sobreajuste decorrente do uso da matriz no ambiente do SA.

Com base nos resultados desse experimento e para simplificar ainda mais o método proposto, no restante dos experimentos, ao se referir ao método de fusão SA, assume-se que o vetor de pesos está sendo utilizado sem nenhum tipo de mecanismo de regularização atrelado. Com isso, a função objetivo do SA se reduz ao valor da acurácia obtido a partir do conjunto de treinamento, como especificado na Seção 4.8.

5.3.4 **Comparação dos métodos de fusão**

Uma série de experimentos são propostos com o objetivo de investigar alguns aspectos do método de fusão usado, que demonstrem a sua real contribuição à arquitetura desenvolvida.

No primeiro experimento, é feita uma avaliação da meta-heurística SA como método de otimização de hiperparâmetros. Para isso, compara-se o método de ajuste de pesos baseado no SA com aquele adotado por Concha et al. (2018), que é baseado em um algoritmo de busca por força bruta. Uma estratégia de busca em grade em que cada peso tem seu valor variado em um intervalo de 0 a 10 com saltos de 0,5. Usando essa abordagem de *grid search*, o processo de busca pela melhor combinação de pesos para a arquitetura *four-stream* desenvolvida envolveria a exploração de 160.000 conjuntos possíveis de pesos. Este fato dá uma dimensão de quão proibitiva esta estratégia pode ser dependendo do conjunto de parâmetros a serem analisados.

O mesmo protocolo usado no SA, descrito na Seção 4.8, foi aplicado à abordagem baseada no *grid search*. Nesse protocolo, os pesos da fusão são estimados a partir do conjunto de treinamento. Os valores reportados na Tabela 5 são os valores de acurácia média obtidos usando um 3-*fold* como método de validação cruzada.

A superioridade do SA na maior parte das situações mostra que a discretização do espaço de busca aplicada ao *grid search*, para viabilizar o processo de hiperparâmetrização, tem um impacto negativo na qualidade das soluções obtidas. Essa constatação, aliada à

explosão combinatória associada à natureza desse método, justificam o uso de técnicas de busca mais direcionadas, como a realizada pelo SA, em detrimento às abordagens exaustivas. Em ambos os *datasets*, fica evidente que a complexidade da abordagem *Grid* domina o SA assintoticamente a partir do terceiro fluxo adicionado, mostrando que uma estratégia exaustiva rapidamente se torna inviável com o aumento do número de *streams* adotadas na arquitetura.

Tabela 5 – Comparação da acurácia e do número de iterações (It) entre o *grid search* e o SA no conjunto de treinamento.

Streams	UCF101				HMDB51			
	Grid		SA		Grid		SA	
	Acc	It	Acc	It	Acc	It	Acc	It
RGB*+OF	91.71	400	**91.72**	5112	63.71	400	**63.72**	4515
RGB*+OF+MVR	**92.43**	8000	**92.43**	5509	65.65	8000	**65.67**	4744
RGB*+OF+MOFR	**93.23**	8000	**93.23**	5281	67.93	8000	**67.96**	4670
Todas as *streams*	93.36	160000	**93.37**	5443	68.39	160000	**68.42**	4992

Fonte: elaborado pelo autor.

O próximo experimento fornece uma estimativa da melhoria de desempenho na arquitetura, originada da adoção do método de fusão SA (*SA fusion*). Para este fim, ele é comparado a outros métodos de fusão comumente usados na literatura. A média simples (SIMONYAN; ZISSERMAN, 2014a; CARREIRA; ZISSERMAN, 2017) (Avg), a média ponderada e a fusão baseada em outros modelos de ML. Duas variantes da média ponderada foram usadas nos experimentos. A primeira, $W_{3,2,1,1}$, usa na ponderação a melhor combinação de pesos reportada por Concha et al. (2018). Nesta configuração, fluxo óptico, RGB* e ritmo visual devem ter peso 3, 2 e 1, respectivamente, no processo de fusão. Complementamos este conjunto de pesos atribuindo o mesmo peso usado para o ritmo visual para o MOFR, assim, adequando-o à arquitetura 4-stream proposta. O outro método de fusão por média ponderada (W_{acc}) usa pesos proporcionais aos valores individuais de acurácia das *streams*, obtidos no experimento anterior. A combinação de pesos usada no W_{acc} para UCF101 foi de $0,8536$, $0,8388$, $0,7036$ e $0,6702$, respectivamente para fluxo óptico, RGB*, MVR e MOFR, e a combinação usada para HMDB51 foi $0,5593$, $0,4927$, $0,4249$ e $0,4247$, respectivamente para fluxo óptico, RGB*, MVR e MOFR.

Nos dois esquemas de fusão baseados em modelos de ML, fusão por SVM linear (Um-contra-um) e por regressão logistica (RL), foi adotada uma abordagem semelhante à empregada por Simonyan e Zisserman (2014a): um modelo de ML é treinado usando uma pilha normalizada em L_2 das *features* descritas na Seção 4.8. A fim de garantir que os modelos generalizem os dados corretamente, foi aplicado um 10-fold à regressão logística e um 10-10-fold ao SVM. Os hiperparâmetros de ambos os modelos foram estimados a

partir de uma busca exaustiva em um conjunto de parâmetros frequentemente reportado na literatura (SIMONYAN; ZISSERMAN, 2014a; WU et al., 2015; YE et al., 2015). As Tabelas 6 e 7 mostram os valores de acurácia obtidos no conjunto de teste usando os vários métodos de fusão testados.

Tabela 6 – Resultados de acurácia de diferentes métodos de fusão no *dataset* UCF101.

Streams	UCF101					
	Avg	W_{acc}	$W_{3,2,1,1}$	SVM	RL	SA
RGB*+OF	93,32	93,36	93,52	93,41	93,29	**93,57**
RGB*+OF+MVR	93,06	93,44	93,60	93,42	93,08	**93,99**
RGB*+OF+MOFR	94,19	94,60	94.57	94,28	94,21	**94,79**
Todas as *streams*	93,42	93,91	94,05	93,75	93,70	**94,84**

Fonte: elaborado pelo autor.

Tabela 7 – Resultados de acurácia de diferentes métodos de fusão no *dataset* HMDB51.

Streams	HMDB51					
	Avg	W_{acc}	$W_{3,2,1,1}$	SVM	RL	SA
RGB*+OF	66,84	67,06	67,12	**67,14**	67,12	67,10
RGB*+OF+MVR	66,78	67,63	67,99	67,55	67,50	**68,34**
RGB*+OF+MOFR	69,69	70,19	70,25	69,82	69,62	**70,38**
Todas as *streams*	68,91	69,93	70,02	69,88	69,76	**70,37**

Fonte: elaborado pelo autor.

Analisando o desempenho inferior do método de fusão por média simples reportado nas Tabelas 6 e 7, quando comparado às outras técnicas de fusão, conclui-se que atribuir o "mesmo peso" a todos os fluxos pode fazer com que a abordagem proposta não extraia o potencial máximo dos elementos que constituem a arquitetura. É interessante notar que a adição do MVR à arquitetura *two-stream* (RGB*+OF), provocou uma diminuição na acurácia do modelo, o que deixa ainda mais evidente que essa *stream* deve ter um peso menor que as outras duas *streams* clássicas. A comparação entre os três métodos de fusão por ponderação ($W_{3,2,1,1}$, W_{acc} e SA) mostra que uma simples atribuição de peso pode não ser a melhor opção, motivando o uso de técnicas para buscar o melhor conjunto de pesos. Em certas situações, por exemplo, o SA supera o W_{acc} e o $W_{3,2,1,1}$ por quase 1%, uma quantia considerável se for levado em conta que toda essa análise está sendo realizada no limites de desempenho individual das *streams*. O método proposto supera ainda métodos mais elaborados de fusão como aqueles que aprendem a fundir as *features* usando outros modelos de ML. Isso fica evidente comparando os resultados do SA e do SVM e verificando que somente em um dos cenários, o método fundamentado no SVM apresenta um resultado melhor que a fusão introduzida neste trabalho.

5.3.5 **Desempenho da combinação de *streams***

Como mencionado anteriormente, vídeos consistem de informações de movimento estáticas e dinâmicas. Por esse motivo, é essencial que a capacidade das *streams* de trabalhar nesses diferentes aspectos de um vídeo seja avaliada. Uma maneira interessante de fazer isso é analisar a complementaridade entre esses fluxos de informação. Neste experimento, são fornecidos mecanismos para mensurar esse aspecto complementar entre as diversas *streams* que compõem a arquitetura deste trabalho, com base no valor de acurácia obtido em várias combinações dessas *streams*. Cada possível combinação de *streams* dentre as quatro disponíveis é analisada. O método de fusão desenvolvido é executado dez vezes usando o procedimento descrito na Subseção 4.8 e uma semente de aleatoriedade diferente é associada a cada execução. Esse procedimento é uma tentativa de assegurar que os resultados serão reprodutíveis e que um possível viés decorrente de processos aleatórios no SA e durante o treinamento das *streams* seja mitigado. Os valores reportados nas Tabelas 8, 9, 10 e 11 são a média dessas execuções.

Tabela 8 – Resultados da fusão *two-stream* no dataset UCF101.

Duas *streams*	*Split* 1	*Split* 2	*Split* 3	Média
RGB*+OF	93,63	92,96	94,13	**93,57**
RGB*+MVR	90,85	90,33	90,53	90,57
RGB*+MOFR	91,05	90,61	90,68	90,78
OF+MVR	89,27	89,31	90,88	89,82
OF+MOFR	88,05	90,55	89,91	89,49
MVR+MOFR	80,95	80,45	82,49	81,30

Fonte: elaborado pelo autor.

Tabela 9 – Resultados da fusão *multi-stream* no *dataset* UCF101. Os resultados com † são os valores reportados por Concha et al. (2018). Os valores demarcados com ‡ referem-se aos resultados reportados por Tacon et al. (2019).

Multi stream	*Split* 1	*Split* 2	*Split* 3	Média
RGB*+OF+AVR†	–	–	–	93,74
RGB*+OF+WVR‡	–	–	–	93,80
RGB*+OF+MVR	93,88	93,41	94,69	93,99
RGB*+OF+MOFR	94,85	94,78	94,75	94,79
RGB*+MVR+MOFR	93,09	92,81	92,96	92,95
OF+MVR+MOFR	89,85	91,14	91,29	90,76
Todas as *streams*	94,85	94,65	95,01	**94,84**

Fonte: elaborado pelo autor.

Examinando os melhores resultados presentes na Tabelas 8 e 10, a contribuição crucial da combinação RGB*+Fluxo óptico se torna evidente. Esses elementos constituem

Tabela 10 – Resultados da fusão *two-stream* no dataset HMDB51.

Duas *stream*	*Split* 1	*Split* 2	*Split* 3	Média
RGB*+OF	68,24	66,34	66,73	**67,10**
RGB*+MVR	60,33	57,52	56,99	58,28
RGB*+MOFR	66,59	64,50	62,55	64,55
OF+MVR	65,58	63,41	62,72	63,90
OF+MOFR	64,90	65,20	64,54	64,88
MVR+MOFR	54,88	53,71	52,17	53,59

Fonte: elaborado pelo autor.

Tabela 11 – Resultados da fusão *multi-stream* no *dataset* HMDB51. Os resultados com † são os valores reportados por Concha et al. (2018). Os valores demarcados com ‡ referem-se aos resultados reportados por Tacon et al. (2019).

Multi stream	*Split* 1	*Split* 2	*Split* 3	Média
RGB*+OF+AVR†	–	–	–	69,98
RGB*+OF+WVR‡	–	–	–	67,10
RGB*+OF+MVR	70,05	68,05	66,93	68,34
RGB*+OF+MOFR	71,88	70,54	68,72	**70,38**
RGB*+MVR+MOFR	67,65	65,50	64,21	65,79
OF+MVR+MOFR	67,44	67,22	65,40	66,69
Todas as *streams*	71,90	70,37	68,85	70,37

Fonte: elaborado pelo autor.

a base da maior parte dos trabalhos da literatura. É perceptível que, embora as duas *streams*, MVR e MOFR, apresentem resultados individuais bem similares, a melhoria proporcionada pela incorporação do MOFR foi notavelmente superior. Este fato, retratado pelos resultados nas Tabelas 9 e 11, indica que o MOFR é mais complementar às outras *streams* da arquitetura se comparado ao MVR, AVR e WVR$^+$. Os resultados do AVR, marcados com uma adaga (†) nas Tabelas 9 e 11, são os resultados reportados por Concha et al. (2018). Os valores demarcados com adagas duplas (‡) referem-se aos resultados reportados na arquitetura 3-*stream* de Tacon et al. (2019).

Para complementar os resultados obtidos em experimentos anteriores, foi adotado um procedimento semelhante ao empregado por Choutas et al. (2018) para mostrar o impacto da sua *feature* PoTion na arquitetura I3D (CARREIRA; ZISSERMAN, 2017). Os autores realizaram um experimento para analisar as diferenças de acurácia entre a I3D e seu método I3D+PoTion para cada classe do conjunto de dados JHMDB (JHUANG et al., 2013) e para as classes que apresentaram as maiores variações, positivas e negativas, do *dataset* Kinetics (KAY et al., 2017).

Esse experimento serviu como base para que fosse decidido que seriam avaliadas

as diferenças de acurácia nas 10 melhores e 10 piores classes, quando o MVR e o MOFR fossem usados em conjunto com a arquitetura *two-stream*, comparado ao mesmo resultado obtido se somente a arquitetura *two-stream* fosse utilizada. As médias das diferenças das classes em que houve algum tipo de melhora (*aumento médio acc*) e de piora (*decréscimo médio acc*) na acurácia também foram relatadas. Para entender o comportamento dessas variações por classe em diferentes situações, dois cenários comparativos foram criados. No primeiro, todas as análises de ambos os métodos foram realizadas usando a fusão por média simples (Avg). Dessa forma, é estabelecido uma espécie de *baseline* e as conclusões obtidas podem ser extrapoladas para além do método de fusão proposto. No último cenário, esse mesmo experimento é refeito usando o SA como método de fusão. Para isso, o esquema descrito na Seção 4.8 é executado para os dois métodos (*two-stream+features* propostas e *two-stream* isoladamente). De forma sucinta, com este experimento, pretende-se deixar claro os benefícios de utilizar as *streams* descritas neste trabalho, MOFR e MVR, em conjunto com a arquitetura original *two-stream* de Simonyan e Zisserman (2014a). As Figuras 17 e 18 ilustram os resultados desse experimento.

Os gráficos das Figuras 17(b), 17(d), 18(b) e 18(d) mostram que o MOFR complementa e ajuda a arquitetura desenvolvida a superar o método *two-stream*. Adicionando o MOFR por meio da fusão por média simples, por exemplo, nota-se uma melhora média de 3,68% e 6,39%, respectivamente, nas classes da UCF101 e HMDB51. Esse mesmo comportamento foi observado quando a *SA fusion* foi utilizada para adicionar o MOFR à abordagem desenvolvida. Nessa situação, a melhoria média foi de 3,11% nas classes da UCF101 e 5,84% nas classes da HMDB51. Os resultados obtidos estão descritos nas Tabelas 12 e 13.

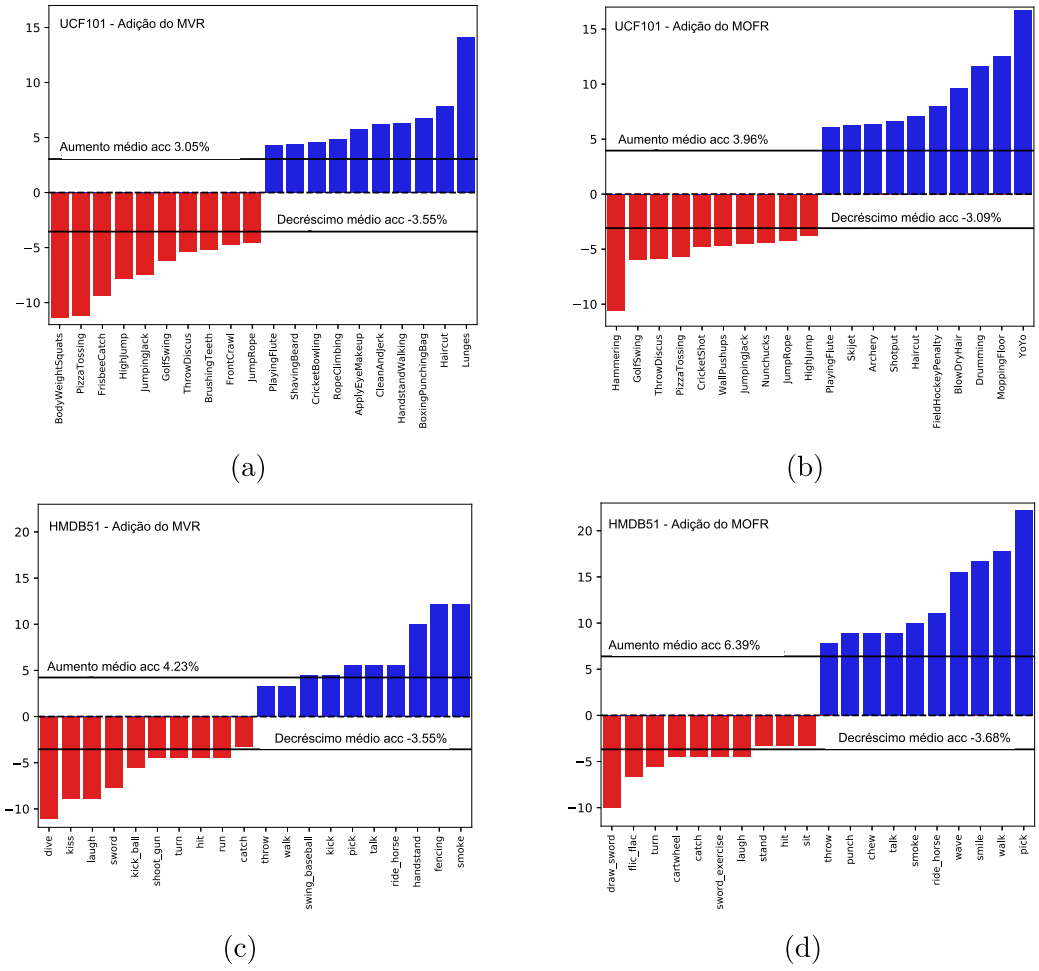
Tabela 12 – Resultados do cálculo de melhora por classe no *dataset* UCF101.

Stream adicionada	Fusão	Melhora	Média	Piora	Média
MVR	Avg	37	3,05 ± 2,75	38	3,55 ± 2,72
	SA	48	2,08 ± 1,52	27	2,03 ± 1,65
MOFR	Avg	46	3,96 ± 3,37	29	3,09 ± 2,15
	SA	54	3,11 ± 2,55	17	2,30 ± 1,59

Fonte: elaborado pelo autor.

Uma análise posterior mais cuidadosa dos resultados mostrou que os resultados mais significativos usualmente estavam relacionados a ações com padrões de movimento bem definidos, como "*yoyo*" (Figura 19) e "*haircut*" na UCF101, e nas classes "*wave*" e "*shake hands*" da HMDB51. Por outro lado, os valores mais baixos geralmente eram associados às classes distinguíveis apenas pela aparência, ou a ações com uma quantidade considerável de movimento em ambas as direções (horizontal e vertical), como a ação

Figura 17 – Variações de acurácia por classe obtidas na UCF101 e na HMDB51 quando o MVR ou o MOFR são agregados à arquitetura *two-stream* usando a fusão por média simples (10 melhores e 10 piores classes).



Fonte: elaborado pelo autor.

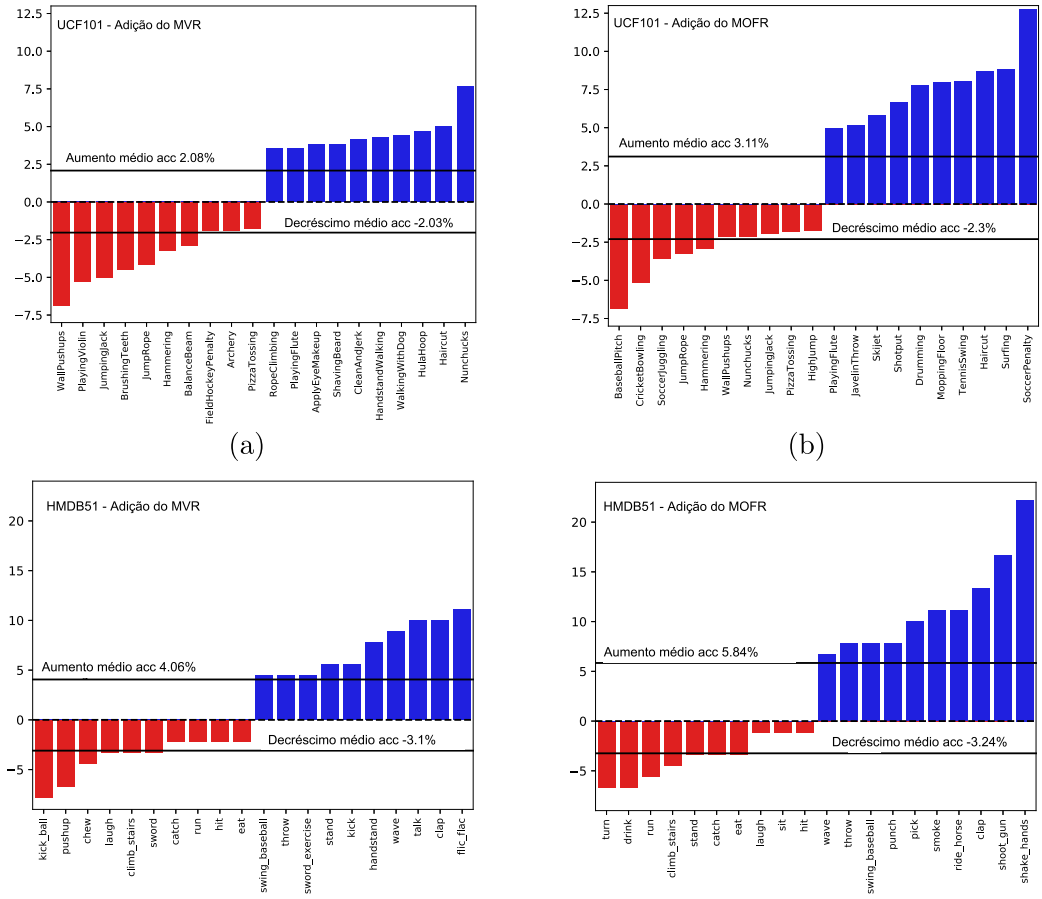
Tabela 13 – Resultados do cálculo de melhora por classe no *dataset* HMDB51.

Stream adicionada	Fusão	Melhora	Média	Piora	Média
MVR	Avg	21	4,23 ± 3,30	26	3,55 ± 2,76
	SA	26	4,06 ± 3,03	14	3,10 ± 1,93
MOFR	Avg	32	6,39 ± 5,21	16	3,68 ± 2,31
	SA	35	5,84 ± 4,53	12	3,24 ± 2,10

Fonte: elaborado pelo autor.

"*wall pushups*" na UCF101, conforme retratado na Figura 20. Neste caso particular, assim como levantado por Concha et al. (2018) e Tacon et al. (2019), planos extraídos em outras direções (e.g. verticais) poderiam capturar com mais precisão a assinatura do movimento realizado. Apesar dessas observações, vale a pena destacar que, em ambos os cenários

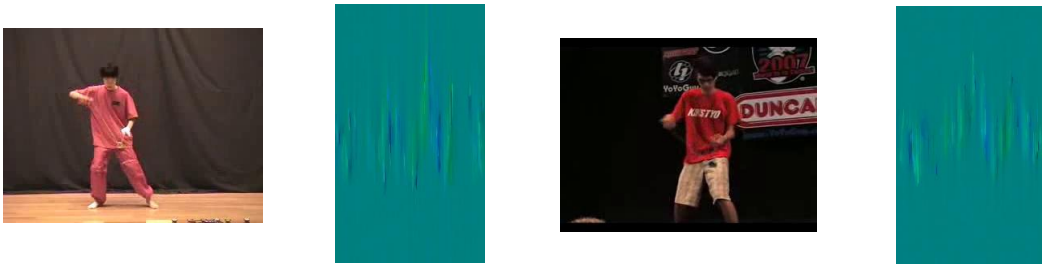
Figura 18 – Variações de acurácia por classe obtidas na UCF101 e na HMDB51 quando o MVR ou o MOFR são agregados à arquitetura *two-stream* usando o SA como método de fusão (10 melhores e 10 piores classes).



Fonte: elaborado pelo autor.

apresentados, o MOFR aprimora o desempenho da arquitetura original, suportando o argumento de que o MOFR é uma *stream* útil em métodos de múltiplas *streams*.

Figura 19 – Dois vídeos distintos da classe “yoyo” e os seus respectivos OFRs retirados de um plano central para fins de simplicidade. Note que os padrões característicos formados no OFR de ambos os vídeos se assemelham. Um indicativo de que o MOFR possivelmente é uma *feature* discriminante dessa classe.



Fonte: elaborado pelo autor.

Figura 20 – Situações onde o MOFR proposto pode não conseguir distinguir corretamente as ações realizadas.

Karatê

Taekwondo

(Duas classes com um assinatura de movimento similar)

(Movimentação em ambas direções em um vídeo da ação “wall pushup”)

Fonte: elaborado pelo autor.

De acordo com as Tabelas 12 e 13, e as Figuras 17(a), 17(c), 18(a) e 18(c), a outra *stream* proposta, o MVR, só adicionou algum novo conhecimento à arquitetura quando combinada através da fusão por SA proposta. Esse tipo de comportamento já foi observado em outros trabalhos (CONCHA et al., 2018; TACON et al., 2019). Em (TACON et al., 2019), por exemplo, os melhores resultados obtidos pelos autores foram encontrados quando um peso menor foi atribuído às predições fornecidas por sua *stream* baseada no ritmo visual. Todas as conclusões acima reforçam que a fusão por média simples pode não extrair todo o potencial das *streams* da arquitetura. Esses resultados também permitem inferir que certas *streams* influenciam mais ativamente o resultado final da abordagem, conforme mencionado.

De fato, é natural supor que as *streams* contribuem de forma diferente para compor a arquitetura final. Algumas *features* têm uma capacidade maior de sintetizar e criar representações eficientes para ações que facilitem seu processo de identificação (KONG; FU, 2018). Em um dos primeiros estudos nessa direção no âmbito de DL, Simonyan e Zisserman (2014a) mostra que uma rede temporal baseada apenas no fluxo óptico pode superar uma rede que trata o aspecto temporal diretamente dos *frames* RGB. Partindo deste trabalho revolucionário, várias outras representações foram introduzidas como *features* de entrada de modelos de DL. Algumas representações com maior poder discriminativo que visam melhorias mais globais aos métodos (SIMONYAN; ZISSERMAN, 2014a; BILEN et al., 2016) e outras com contribuições mais específicas, almejando certos aspectos específicos das ações humanas (CHOUTAS et al., 2018; CONCHA et al., 2018). O ritmo visual se encaixa no segundo grupo, apresentando-se como uma maneira de modelar em 2D as relações temporais de longo alcance. Em uma perspectiva mais ampla, seu desempenho isolado está abaixo das outras *features* geralmente usadas como o RGB e o fluxo óptico. No entanto, esse fato não invalida o uso do MVR nesse domínio. Pelo contrário, os resultados obtidos no segundo cenário, representados pelas Figuras 18(a) e 18(c), demonstram que

ações específicas se beneficiam consideravelmente da agregação desta *stream*. Por exemplo, classes reconhecíveis somente a partir do aprendizado de relações espaço-temporais nos dados como "*rope climbing*" e "*walking with dog*".

5.3.6 **Hierarquia de arquiteturas**

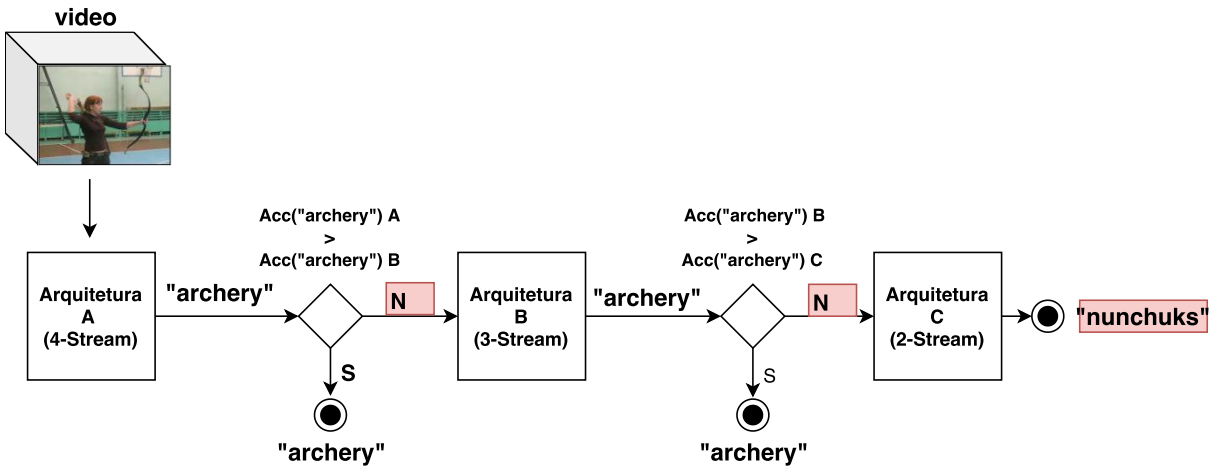
O resultado do experimento anterior, levantou algumas questões interessantes a serem respondidas. Um desses pontos é o fato de que, pelo menos atualmente, ainda não existe um método de fusão que consiga capturar integralmente o conhecimento das *streams* que constituem a arquitetura. Sempre existe algum tipo de perda durante o processo de agregação, e essas perdas tendem a crescer com o aumento no número de *streams*, já que mais variáveis são adicionadas aos cálculos. As Figuras 17 e 18 retratam uma dessas situações, em que parte da informação é perdida pela adição de uma terceira *stream* à arquitetura, que fez com que ela "desaprendesse" algumas classes. Em condições ideais esse tipo de comportamento nunca deveria ser observado, uma vez que, os métodos de fusão deveriam ser capazes de reter todo o conhecimento adquirido previamente, fazendo com que o conhecimento só pudesse ser adicionado e nunca perdido durante a combinação das *streams*. Embora desejável, essa característica não é tão simples de ser alcançada (KONG; FU, 2018).

Com base nos resultados do experimento anterior e nestas observações iniciais, é proposta uma espécie de hierarquia de arquiteturas. A premissa básica desse arranjo de modelos é a de que as amostras deveriam ser classificadas somente pelos modelos que têm o conhecimento necessário para identificá-las. Dessa forma, espera-se que este efeito de "esquecimento" proporcionado pela adição de uma *stream* seja mitigado, e que o conhecimento só seja adicionado à arquitetura. O primeiro nível da hierarquia proposta compreende a arquitetura que, em tese, é a mais robusta, o modelo com 4 *streams*. No próximo nível estão as arquiteturas 3-*stream* (RGB*+OF+MVR e RGB*+OF+MOFR) e por fim, no último nível hierárquico, está a arquitetura *two-stream* original (RGB*+OF) (SIMONYAN; ZISSERMAN, 2014a). Essa estrutura padrão pode passar por alterações para inclusão e remoção de níveis hierárquicos.

Nesse esquema, ao processar uma amostra, é feita uma verificação se a arquitetura no nível analisado classifica melhor a ação predita, se comparado à arquitetura do próximo nível. Se o resultado for sim, a ação predita pela arquitetura se torna o palpite final do método. Caso contrário, ela repassa a amostra para o próximo nível hierárquico que repete esse mesmo passo de verificação, até que um dos níveis forneça a predição final do vídeo em questão. Para verificar se uma determinada arquitetura classifica melhor uma amostra, é refeito o mesmo procedimento adotado no experimento da Seção 5.3.5, onde são analisadas as diferenças de acurácia por classe decorrentes da adição de uma nova *stream* à arquitetura. A Figura 21 ilustra o processo de funcionamento desse método.

Algumas combinações possíveis das *streams* utilizadas neste trabalho foram testadas e esses resultados estão descritos na Tabela 14.

Figura 21 – Uma visão geral da hierarquia de arquiteturas classificando uma amostra da classe "*archery*". O fluxo de execução realizado está destacado em vermelho. É interessante notar que o último modelo "C" que teoricamente seria o mais indicado para classificar a amostra de vídeo analisada pode classificá-la erroneamente, porém espera-se que essa situação não ocorra muito.



Fonte: elaborado pelo autor.

Tabela 14 – Resultados obtidos a partir da utilização da arquitetura hierárquica nos dois datasets.

Hierarquia	UCF101	HMDB51
4-*stream*	**94,84**	**70,37**
4-*stream* → 3-*stream* (MOFR)	**94,84**	69,20
4-*stream* → 3-*stream* (MVR)	94,64	70,12
4-*stream* → 3-*stream* (MOFR) → 2-*stream*	94,71	70,07
4-*stream* → 3-*stream* (MVR) → 2-*stream*	94,52	69,12
4-*stream* → 3-*stream* (MVR) → 3-*stream* (MVR)	94,62	70,27
4-*stream* → 3-*stream* (MVR) → 3-*stream* (MOFR) → 2-*stream*	94,61	69,88

Fonte: elaborado pelo autor.

Observando os resultados da Tabela 14, verificou-se que a estrutura hierárquica projetada não obteve o desempenho esperado, ficando inclusive pior que a arquitetura 4-*stream* isoladamente. Alguns fatores podem estar influenciando negativamente os resultados do método proposto. Por exemplo, o 3-*fold* usado para estimar as variações da acurácia interclasse entre as arquiteturas pode não estar conseguindo generalizar a ponto de refletir esses mesmos conhecimentos no conjunto de teste. Além disso, como somente o valor de acurácia é levado em consideração durante este procedimento, não existe nenhuma garantia de que o conjunto de amostras de teste fornecidos de uma determinada classe seja realmente identificável pela arquitetura com maior acurácia nessa classe. Dessa forma,

uma quantidade considerável de erros pode estar sendo propagada através das arquiteturas, o que impacta significativamente o resultado final da abordagem, conforme observado nos resultados da Tabela 14. Um estudo mais aprofundado tem que ser realizado para responder essas questões e tentar tornar essa forma de abordar o problema viável.

5.3.7 **Comparação com as abordagens estado da arte**

Por fim, é feita uma comparação da arquitetura *four-stream* introduzida no presente trabalho com as abordagens consideradas estado da arte no HAR. O protocolo de teste descrito na Seção 5.1 foi utilizado para avaliar a arquitetura proposta.

Os melhores resultados obtidos foram 94,8% para UCF101 e 70,4% para HMDB51. A Tabela 15 mostra que o método *multi-stream* proposto alcança taxas de acurácia muito competitivas se comparado às abordagens estado-da-arte.

Tabela 15 – Comparação das taxas de acurácia (%) para os *datasets* HMDB51 e UCF101.

Método	Pré-treinamento	UCF101	HMDB51
I3D (CARREIRA; ZISSERMAN, 2017)	ImageNet+Kinetics	98,0	80,7
I3D+PoTion (CHOUTAS et al., 2018)	ImageNet+Kinetics	**98,2**	80,9
SVMP+I3D (WANG et al., 2018)	ImageNet+Kinetics	–	81,3
DTPP (ZHU; ZHU; ZOU, 2018)	ImageNet+Kinetics	98,0	**82,1**
TDD^{iDT} (WANG; QIAO; TANG, 2015)	ImageNet	91,5	65,9
TVNets^{iDT} (FAN et al., 2018b)	ImageNet	**95,4**	**72,6**
STResnet^{iDT} (FEICHTENHOFER; PINZ; WILDES, 2017)	ImageNet	94,9	72,2
OFF (SUN et al., 2018)	—	**96,0**	**74,2**
STResnet (FEICHTENHOFER; PINZ; WILDES, 2017)	ImageNet	94,2	68,9
Two-Stream (SIMONYAN; ZISSERMAN, 2014a)	ImageNet	88,0	59,4
Two-Stream TSN (WANG et al., 2016)	ImageNet	94,0	68,5
Three-Stream TSN (WANG et al., 2016)	ImageNet	94,2	69,4
Three-Stream (WANG et al., 2017)	ImageNet	94,1	**70,4**
KVMDF (ZHU et al., 2016)	ImageNet	93,1	63,3
STP (WANG et al., 2017b)	ImageNet	94,6	68,9
Multi-Stream+ResNet152 (CONCHA et al., 2018)	ImageNet	94,3	68,3
Multi-Stream+InceptionV3 (CONCHA et al., 2018)	ImageNet	93,7	69,9
Multi-Stream (TACON et al., 2019)	ImageNet	93,8	67,1
Four-Stream proposta	ImageNet	**94,8**	**70,4**

Fonte: elaborado pelo autor.

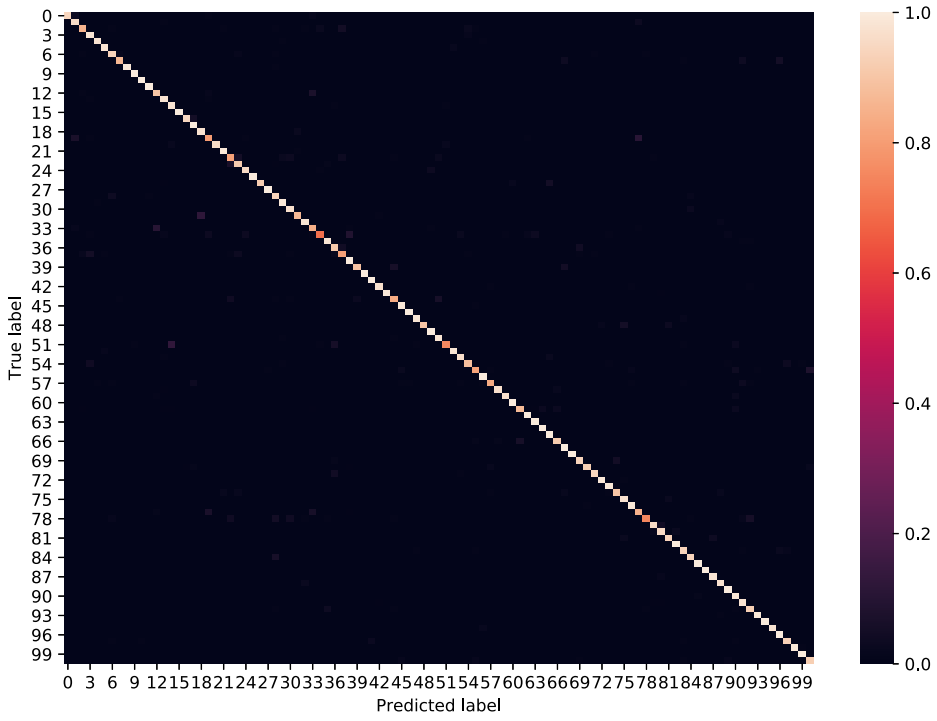
Para ambos os conjuntos de dados, o método desenvolvido é superado apenas pelos trabalhos pré-treinados com o conjunto de dados Kinetics (KAY et al., 2017) e pelo método OFF introduzido por Sun et al. (2017). No entanto, não consideramos uma comparação direta com essas abordagens baseadas na Kinetics. A quantidade expressiva de recursos computacionais necessários para treinamento com esse *dataset* torna seu uso impraticável sem uma infraestrutura de alto desempenho. As abordagens que usam o

IDT também não foram levadas em consideração durante os testes comparativos. Embora essa *feature* seja uma forma eficiente de adicionar novos conhecimentos a métodos DL, ela pode mascarar o desempenho isolado da arquitetura no HAR, dificultando uma análise comparativa mais precisa destes métodos. Os resultados competitivos da STResnet, por exemplo, têm um decréscimo de 0,7% na UCF101 e 3,3% na HMDB51, sem a utilização desta *feature*, deixando este método com resultados inferiores ao de vários outros métodos que também não utilizam o IDT.

Dentre os trabalhos estado da arte, destaca-se o OFF. Esse método não utiliza o pré-treinamento da Kinetics e nem explora a agregação com o IDT e ainda assim consegue resultados bem próximos das abordagens que se utilizam desses artifícios. O resultado apresentado é proveniente de uma arquitetura formada pela combinação das streams RGB e OF e da nova *stream* OFF aplicada à diferença RGB, do OF e do RGB. No entanto, vale a pena salientar que, assim como destacado pelos próprios autores, o processo de treinamento das diversas sub-redes responsáveis pela geração do OFF demanda muito esforço computacional, o que inviabiliza o uso dessa arquitetura em ambientes que não sejam de alto desempenho.

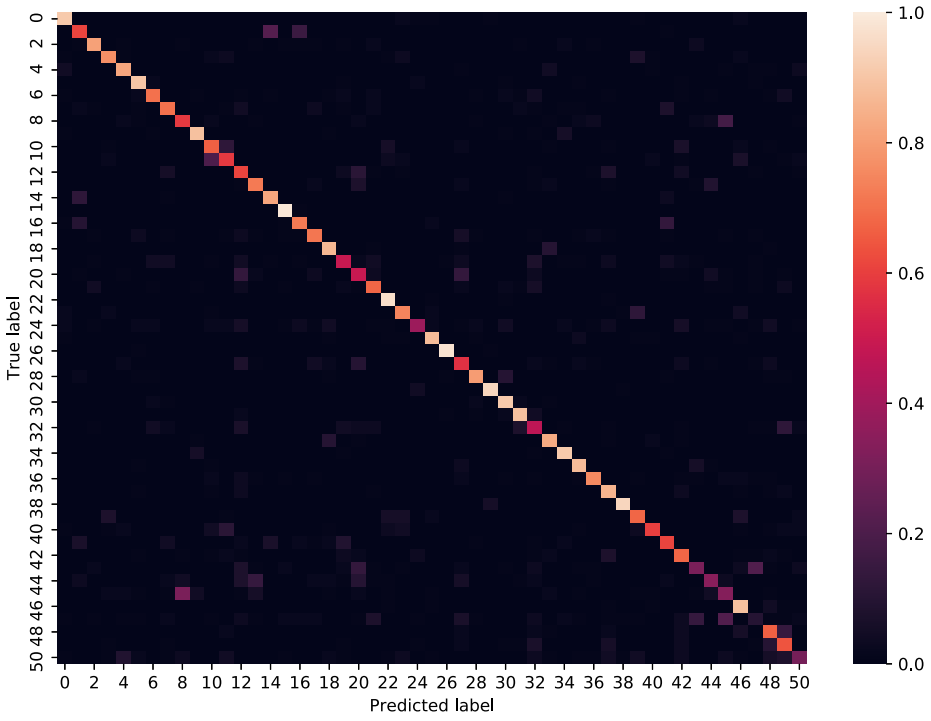
As matrizes de confusão da arquitetura *multi-stream* proposta para as bases de dados UCF101 e HMDB51, respectivamente, estão representadas nas Figuras 22 e 23. As classes estão ordenadas por nome. Analisando essas matrizes, é possível observar uma razoável confusão entre as classes "*blow dry hair*" e "*haircut*" do *dataset* UCF101 (índices 12 e 33, respectivamente). Conjectura-se que isso esteja relacionado ao fato de que ambas as classes têm uma assinatura de movimento bem similar. Na HMDB51, a confusão entre classes ocorre entre as ações "*draw sword*" e "*sword exercise*" (8 e 45, respectivamente), o que provavelmente se deve ao fato de que essas duas classes possuem amostras extraídas dos mesmos vídeos. Amostras quase idênticas em aparência e, que em alguns casos, também podem possuir assinaturas de movimento bem semelhantes, o que dificulta o processo de distinção entre estas ações.

Figura 22 – Matriz de confusão da arquitetura *multi-stream* final sobre os três *splits* da UCF101.



Fonte: elaborado pelo autor.

Figura 23 – Matriz de confusão da arquitetura *multi-stream* final sobre os três *splits* da HMDB51.



Fonte: elaborado pelo autor.

6 CONSIDERAÇÕES FINAIS

Este trabalho apresentou uma série de melhorias a uma arquitetura *multi-stream* aplicada ao problema de reconhecimento de ações humanas baseada em vídeos. A arquitetura construída sobre essas modificações apresentou resultados competitivos, comparáveis às abordagens estado da arte. Um indicativo de que as modificações propostas foram efetivas na solução do problema.

Para aperfeiçoar a *stream* espaço-temporal, foi introduzido o uso de múltiplos ritmos visuais (MVR). Com a extração de vários ritmos visuais, vários aspectos do objeto em movimento são explorados, o que aprimora o processo de reconhecimento da ação correspondente. Os experimentos reforçaram que ganhos expressivos podem ser obtidos usando essa estratégia de utilizar mais planos para representar os vídeos. A *stream* espaço-temporal aprimorada no presente trabalho supera a *stream* espaço-temporal original de Concha et al. (2018) nos dois conjuntos de dados avaliados.

A contribuição mais significativa deste trabalho foi a incorporação de uma nova *stream* temporal ao modelo de arquiteturas *multi-stream*. A informação criada para esse fim, o ritmo do fluxo óptico, é uma fatia extraída do volume de fluxo óptico de um vídeo. Assim como o ritmo visual, o ritmo do fluxo óptico é uma *feature* criada com o intuito de codificar relações de longo alcance nos vídeos. Comparado ao ritmo visual, o ritmo do fluxo óptico contribui mais para a arquitetura desenvolvida. Isso significa que o ritmo do fluxo óptico é mais complementar que o ritmo visual, às *streams* clássicas RGB* e fluxo óptico. Os resultados dos experimentos mostram uma melhora significativa no desempenho final da abordagem, proporcionada pela incorporação da nova *stream* temporal baseada no uso de múltiplos ritmos do fluxo óptico.

O protocolo de treinamento/teste especificado exclusivamente para as duas *streams* propostas nesse trabalho, MVR e MOFR, se mostrou um ótimo artifício para aumentar a eficácia dessas *streams*. Ao ponderar os diversos ritmos retirados de cada vídeo, consegue-se explorar satisfatoriamente as outras dimensões não endereçadas pelo ritmo usado. Os resultados expressivos obtidos a partir da adoção desse protocolo fizeram com que a *stream* baseada no ritmo visual apresentada superasse com folga as *streams* fundamentadas no VR de Concha et al. (2018) e de Tacon et al. (2019) em ambos os *datasets*.

Além disso, é apresentado um novo método de fusão por média ponderada baseado na utilização da metaheurística *simulated annealing*. A fusão SA proposta é uma forma inteligente de otimizar os pesos da fusão, evitando estratégias exaustivas como um *grid search*. Contrariando os algoritmos de força bruta geralmente empregados nesses métodos exaustivos, o SA orienta a busca por regiões mais promissoras do espaço dos hiperparâmetros, tornando todo o processo mais eficiente. Assim, a arquitetura final tende a sofrer menos com a maldição da dimensionalidade, tornando-se mais escalável.

Os resultados competitivos relatados na Tabela 15 suportam essas premissas e mostram a eficácia da fusão proposta. Vale a pena reforçar que, embora a fusão a nível de classe não tenha apresentado os resultados esperados, existem fortes evidências na literatura e nos experimentos de que essa seja uma forma mais robusta, que o método amplamente difundido de ponderação baseado em vetores de peso, de combinar as várias *streams* de uma arquitetura. A constatação de que certas classes estão altamente ligadas a certas características dos vídeos, por exemplo, é um dos fatores que justifica o uso de um método de fusão que trabalhe a nível de classe.

Ainda que a hierarquia de arquiteturas não tenha se mostrado num primeiro momento uma abordagem viável, foi dado o passo inicial no estudo de métodos que viabilizem esse tipo de arquitetura no reconhecimento de ações em vídeos. Foram levantados dois pontos principais a serem superados por esses métodos: controlar a propagação dos erros entre os níveis da hierarquia e o estudo de formas mais eficientes de mensurar as diferenças de desempenho entre os modelos dos diferentes níveis hierárquicos. Apesar dos desafios a serem superados, acredita-se que esse também seja um possível caminho a ser seguido na construção de métodos cada vez mais robustos dentro do ambiente HAR.

É importante mencionar ainda que as melhorias discutidas neste trabalho não estão restritas à arquitetura proposta e podem ser estendidas a outras abordagens *multi-stream*, como aquelas que trabalham sobre conjuntos maiores de dados, a exemplo da Kinetics. Os ganhos observados provavelmente podem ser reproduzidos sem maiores problemas nessas arquiteturas, fazendo com que esses métodos possam melhorar as suas respectivas eficácias.

Como trabalhos futuros, pretende-se explorar a adição de outras *features* valiosas ao problema de reconhecimento de ações como o iDT (WANG et al., 2013) e o OFF (SUN et al., 2018) à arquitetura do presente trabalho. Uma outra vertente a ser seguida está na utilização de outros modelos de CNN, como base das *streams* como explorado por Carreira e Zisserman (2017). Outro aspecto a ser explorado é a possibilidade de usar várias direções dos ritmos visuais e dos ritmos do fluxo óptico para cada vídeo. Os experimentos realizados por Concha et al. (2018), por exemplo, já mostram sinais de que os ritmos visuais verticais podem melhorar a taxa de reconhecimento desses modelos. Além disso, é necessária uma análise mais profunda do ritmo do fluxo óptico. Entender as limitações dessa abordagem na representação de padrões de movimento de longo prazo presentes em vídeos do mundo real são algumas das questões em aberto a serem respondidas. Outras propostas para trabalhos futuros incluem:

- Usar bases de dados de ações maiores, como a Kinetics e Sports1M (KARPATHY et al., 2014), para pré-treinar o modelo proposto. O maior número de vídeos e classes nesses *datasets* permite uma melhor compreensão das interações relacionadas à execução de uma ação, gerando modelos potencialmente mais robustos;

- Explorar diferentes estratégias de fusão, como fusões por meio de outros modelos, a nível de amostras ou até mesmo baseadas em outras metaheurísticas, como o PSO. Um estudo mais aprofundado da fusão a nível de classe também pode ser um caminho a ser explorado;
- Elaborar uma CNN específica para extração ou treinamento dos ritmos visuais e do ritmo dos fluxos ópticos, assim como realizado por Sun et al. (2017) na criação do OFF.

REFERÊNCIAS

ACEVEDO, Javier; MALDONADO, Saturnino; SIEGMANN, Philip; LAFUENTE, Sergio; GIL, Pedro. Tuning L1-SVM Hyperparameters with Modified Radius Margin Bounds and Simulated Annealing. In: **Proceedings of the 9th International Work Conference on Artificial Neural Networks**. Berlin, Heidelberg: Springer-Verlag, 2007. (IWANN'07), p. 284–291.

ARIF, Sheeraz; JING, Wang; HASSAN, Tehseen; FEI, Zesong. 3d-cnn-based fused feature maps with lstm applied to action recognition. **Future Internet**, v. 11, p. 42, 02 2019.

ASLANI, Sepehr; MAHDAVI-NASAB, Homayoun. Optical flow based moving object detection and tracking for traffic surveillance. **International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering**, v. 7, n. 9, p. 1252–1256, 2013.

BACCOUCHE, Moez; MAMALET, Franck; WOLF, Christian; GARCIA, Christophe; BASKURT, Atilla. Sequential deep learning for human action recognition. In: SPRINGER. **International Workshop on Human Behavior Understanding**. [S.l.], 2011. p. 29–39.

BILEN, Hakan; FERNANDO, Basura; GAVVES, Efstratios; VEDALDI, Andrea; GOULD, Stephen. Dynamic image networks for action recognition. In: **IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2016. p. 3034–3042.

BISHOP, Christopher M. **Pattern Recognition and Machine Learning**. [S.l.]: Springer, 2006.

BLUM, Christian; ROLI, Andrea. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. **ACM Comput. Surv.**, v. 35, p. 268–308, 01 2001.

CARREIRA, Joao; ZISSERMAN, Andrew. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In: IEEE. **IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.], 2017. p. 4724–4733.

CHAQUET, Jose M; CARMONA, Enrique J; FERNÁNDEZ-CABALLERO, Antonio. A survey of video datasets for human action and activity recognition. **Computer Vision and Image Understanding**, Elsevier, v. 117, n. 6, p. 633–659, 2013.

CHERNBUMROONG, Saisakul; CANG, Shuang; YU, Hongnian. Genetic algorithm-based classifiers fusion for multisensor activity recognition of elderly people. **IEEE journal of biomedical and health informatics**, IEEE, v. 19, n. 1, p. 282–289, 2014.

CHOLLET, François et al. **Keras**. 2015. <https://keras.io>.

CHOUTAS, Vasileios; WEINZAEPFEL, Philippe; REVAUD, Jérôme; SCHMID, Cordelia. PoTion: Pose MoTion Representation for Action Recognition. In: **CVPR 2018 - IEEE Conference on Computer Vision and Pattern Recognition**. Salt Lake City, United States: IEEE, 2018. p. 7024–7033.

CONCHA, Darwin Ttito; MAIA, Helena De Almeida; PEDRINI, Helio; TACON, Hemerson; BRITO, Andre De Souza; CHAVES, Hugo De Lima; VIEIRA, Marcelo Bernardes. Multi-stream convolutional neural networks for action recognition in video sequences based on adaptive visual rhythms. In: **IEEE International Conference on Machine Learning and Applications (ICMLA)**. [S.l.]: IEEE, 2018. p. 473 – 480.

DALAL, Navneet; TRIGGS, Bill. Histograms of oriented gradients for human detection. In: **International Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2005. v. 1, p. 886–893.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. ImageNet: A Large-Scale Hierarchical Image Database. In: **IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2009. p. 248 – 255.

DIBA, Ali; SHARMA, Vivek; GOOL, Luc Van. Deep temporal linear encoding networks. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2017. p. 2329–2338.

FAN, Lijie; HUANG, Wen-bing; GAN, Chuang; ERMON, Stefano; GONG, Boqing; HUANG, Junzhou. End-to-End Learning of Motion Representation for Video Understanding. **arXiv preprint arXiv:1804.00413**, 2018.

FAN, Lijie; HUANG, Wenbing; GAN, Chuang; ERMON, Stefano; GONG, Boqing; HUANG, Junzhou. End-to-end learning of motion representation for video understanding. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2018. p. 6016–6025.

FATHI, Alireza; MORI, Greg. Action recognition by learning mid-level motion features. In: IEEE. **2008 IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.], 2008. p. 1–8.

FEICHTENHOFER, Christoph; PINZ, Axel; WILDES, Richard P. Spatiotemporal multiplier networks for video action recognition. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2017. p. 4768–4777.

GOLDBERG, David E. **Genetic Algorithms in Search, Optimization and Machine Learning**. 1st. ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989. ISBN 0201157675.

GUO, Kai; ISHWAR, Prakash; KONRAD, Janusz. Action recognition using sparse representation on covariance manifolds of optical flow. In: IEEE. **2010 7th IEEE international conference on advanced video and signal based surveillance**. [S.l.], 2010. p. 188–195.

HAYKIN, Simon. **Neural networks: a comprehensive foundation**. [S.l.]: Prentice Hall PTR, 1994.

HORN, Berthold; SCHUNCK, Brian G. Determining optical flow. **Artificial Intelligence**, v. 17, p. 185–203, 08 1981.

HUBEL, David H; WIESEL, Torsten N. Receptive fields and functional architecture of monkey striate cortex. **The Journal of physiology**, Wiley Online Library, v. 195, n. 1, p. 215–243, 1968.

HWANG, Inhwan; CHA, Geonho; OH, Songhwai. Multi-modal human action recognition using deep neural networks fusing image and inertial sensor data. In: IEEE. **2017 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)**. [S.l.], 2017. p. 278–283.

JHUANG, H.; GALL, J.; ZUFFI, S.; SCHMID, C.; BLACK, M. J. Towards understanding action recognition. In: **International Conf. on Computer Vision (ICCV)**. [S.l.: s.n.], 2013. p. 3192–3199.

JI, Shuiwang; XU, Wei; YANG, Ming; YU, Kai. 3D Convolutional Neural Networks for Human Action Recognition. **Transactions on Pattern Analysis and Machine Intelligence**, IEEE Computer Society, Washington, DC, USA, v. 35, n. 1, p. 221–231, 2013.

JOHANSSON, Gunnar. Visual perception of biological motion and a model for its analysis. **Perception & Psychophysics**, Springer, v. 14, n. 2, p. 201–211, 1973.

KAHANI, Reza; TALEBPOUR, Alireza; MAHMOUDI-AZNAVEH, Ahmad. A Correlation Based Feature Representation for First-Person Activity Recognition. **arXiv preprint arXiv:1711.05523**, 2017.

KARPATHY, Andrej; TODERICI, George; SHETTY, Sanketh; LEUNG, Thomas; SUKTHANKAR, Rahul; FEI-FEI, Li. Large-scale video classification with convolutional neural networks. In: **CVPR**. [S.l.: s.n.], 2014. p. 1725 – 1732.

KAY, Will; CARREIRA, Joao; SIMONYAN, Karen; ZHANG, Brian; HILLIER, Chloe; VIJAYANARASIMHAN, Sudheendra; VIOLA, Fabio; GREEN, Tim; BACK, Trevor; NATSEV, Paul; SULEYMAN, Mustafa; ZISSERMAN, Andrew. The Kinetics Human Action Video Dataset. **arXiv preprint arXiv:1705.06950**, 2017.

KENNEDY, James. Particle swarm optimization. **Encyclopedia of machine learning**, Springer, p. 760–766, 2010.

KHALIFA, M. H.; AMMAR, M.; OUARDA, W.; ALIMI, A. M. Particle swarm optimization for deep learning of convolution neural network. In: **2017 Sudan Conference on Computer Science and Information Technology (SCCSIT)**. [S.l.: s.n.], 2017. p. 1–5.

KIM, Hyeokman; LEE, Jinho; YANG, Jae-Heon; SULL, Sanghoon; KIM, Woonkyung M; SONG, S Moon-Ho. Visual rhythm and shot verification. **Multimedia Tools and Applications**, Springer, v. 15, n. 3, p. 227–245, 2001.

KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by simulated annealing. **Science**, v. 220, n. 4598, p. 671–680, 1983.

KLASER, Alexander; MARSZAŁEK, Marcin; SCHMID, Cordelia. A Spatio-Temporal Descriptor based on 3D-Gradients. In: BRITISH MACHINE VISION ASSOCIATION. **19th British Machine Vision Conference**. [S.l.], 2008. p. 275–1.

KONG, Yu; FU, Yun. Human action recognition and prediction: A survey. **arXiv preprint arXiv:1806.11230**, 2018.

KOULAMAS, Christos; ANTONY, SR; JAEN, R. A survey of simulated annealing applications to operations research problems. **Omega**, Elsevier, v. 22, n. 1, p. 41–56, 1994.

KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. Imagenet Classification with Deep Convolutional Neural Networks. In: **Advances in Neural Information Processing Systems**. [S.l.: s.n.], 2012. p. 1097–1105.

KUEHNE, Hilde; JHUANG, Hueihan; STIEFELHAGEN, Rainer; SERRE, Thomas. HMDB51: A Large Video Database for Human Motion Recognition. In: **High Performance Computing in Science and Engineering**. [S.l.]: Springer, 2013. p. 571–582.

LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. Deep Learning. **Nature**, Nature Publishing Group, v. 521, n. 7553, p. 436, 2015.

LECUN, Yann; BOTTOU, Léon; BENGIO, Yoshua; HAFFNER, Patrick et al. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, Taipei, Taiwan, v. 86, n. 11, p. 2278–2324, 1998.

LIU, Shuaicheng; YUAN, Lu; TAN, Ping; SUN, Jian. Steadyflow: Spatially smooth optical flow for video stabilization. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2014. p. 4209–4216.

MAHAJAN, Dhruv; GIRSHICK, Ross B.; RAMANATHAN, Vignesh; HE, Kaiming; PALURI, Manohar; LI, Yixuan; BHARAMBE, Ashwin; MAATEN, Laurens van der. Exploring the limits of weakly supervised pretraining. **arXiv preprint arXiv:1805.00932**, 2018.

MAIA, Helena A.; FIGUEIREDO, Ana Mara De Oliveira; OLIVEIRA, Fábio Luiz Marinho De; MOTA, Virgínia Fernandes; VIEIRA, Marcelo Bernardes. A Video Tensor Self-Descriptor based on Variable Size Block Matching. **Journal of Mobile Multimedia**, Rinton Press, Incorporated, v. 11, n. 1&2, p. 090–102, 2015.

MARAVALL, Darío; LOPE, Javier de; H., José Antonio Martín. Hybridizing Evolutionary Computation and Reinforcement Learning for the Design of Almost Universal Controllers for Autonomous Robots. **Neurocomputing**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 72, n. 4-6, p. 887–894, 2009.

MARR, David; HILDRETH, Ellen. Theory of edge detection. **Proceedings of the Royal Society of London. Series B. Biological Sciences**, The Royal Society London, v. 207, n. 1167, p. 187–217, 1980.

MATTIOLI, Fernando; CAETANO, Daniel; CARDOSO, Alexandre; NAVES, Eduardo; LAMOUNIER, Edgard. An Experiment on the Use of Genetic Algorithms for Topology Selection in Deep Learning. **Journal of Electrical and Computer Engineering**, Hindawi, v. 2019, 2019.

MCCULLOCH, Warren S; PITTS, Walter. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, Springer, v. 5, n. 4, p. 115–133, 1943.

MINSKY, Marvin; PAPERT, Seymour. **Perceptrons: An Introduction to Computational Geometry**. Cambridge, MA, USA: MIT Press, 1969.

MOESLUND, Thomas B; HILTON, Adrian; KRÜGER, Volker. A survey of advances in vision-based human motion capture and analysis. **Computer vision and image understanding**, Elsevier, v. 104, n. 2-3, p. 90–126, 2006.

NG, Joe Yue-Hei; HAUSKNECHT, Mattew; VIJAYANARASIMHAN, Sudheendra; VINYALS, Oriol; MONGA, Rajat; TODERICI, George. Beyond Short Snippets: Deep Networks for Video Classification. In: **IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2015. p. 4694–4702.

NIELSEN, Michael A. **Neural networks and deep learning**. [S.l.]: Determination press San Francisco, CA, USA:, 2015. v. 25.

OSMAN, Ibrahim H.; LAPORTE, Gilbert. Metaheuristics: A bibliography. **Annals of Operations Research**, v. 63, n. 5, p. 511–623, 1996.

POUYANFAR, Samira; SADIQ, Saad; YAN, Yilin; TIAN, Haiman; TAO, Yudong; REYES, Maria Presa; SHYU, Mei-Ling; CHEN, Shu-Ching; IYENGAR, S. S. A survey on deep learning: Algorithms, techniques, and applications. **ACM Comput. Surv.**, ACM, New York, NY, USA, v. 51, n. 5, p. 92:1–92:36, set. 2018. ISSN 0360-0300.

RAVANBAKHSH, Mahdyar; MOUSAVI, Hossein; RASTEGARI, Mohammad; MURINO, Vittorio; DAVIS, Larry S. Action Recognition with Image based CNN Features. **arXiv preprint arXiv:1512.03980**, 2015.

RODRIGUEZ, Roberto Iglesias; RODRíGUEZ, Miguel; REGUEIRO, Carlos; CORREA, José; BARRO, S. Combining reinforcement learning and genetic algorithms to learn behaviours in mobile robotics. In: . [S.l.: s.n.], 2006. p. 188–195.

ROKACH, Lior. Ensemble-based classifiers. **Artif. Intell. Rev.**, v. 33, p. 1–39, 02 2010.

ROSENBLATT, Frank. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological review**, American Psychological Association, v. 65, n. 6, p. 386, 1958.

ROUSSEEUW, Peter J. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. **Journal of computational and applied mathematics**, Elsevier, v. 20, p. 53–65, 1987.

SCOVANNER, Paul; ALI, Saad; SHAH, Mubarak. A 3-dimensional sift descriptor and its application to action recognition. In: ACM. **Proceedings of the 15th ACM international conference on Multimedia**. [S.l.], 2007. p. 357–360.

SEVILLA-LARA, Laura; LIAO, Yiyi; GÜNEY, Fatma; JAMPANI, Varun; GEIGER, Andreas; BLACK, Michael J. On the Integration of Optical Flow and Action Recognition. **arXiv preprint arXiv:1712.08416**, 2017.

SEVILLA-LARA, Laura; SUN, Deqing; JAMPANI, Varun; BLACK, Michael J. Optical Flow with Semantic Segmentation and Localized Layers. **arXiv preprint arXiv:1603.03911**, 2016.

SHARMA, Himani; KUMAR, Sunil. A Survey on Decision Tree Algorithms of Classification in Data Mining. **International Journal of Science and Research (IJSR)**, v. 5, 04 2016.

SIMONYAN, Karen; ZISSERMAN, Andrew. Two-Stream Convolutional Networks for Action Recognition in Videos. In: **Advances in Neural Information Processing Systems**. [S.l.: s.n.], 2014. p. 568–576.

SIMONYAN, Karen; ZISSERMAN, Andrew. Very Deep Convolutional Networks for Large-Scale Image Recognition. **arXiv preprint arXiv:1409.1556**, 2014.

SOOMRO, Khurram; ZAMIR, Amir Roshan; SHAH, Mubarak. UCF101: A Dataset of 101 Human Actions Classes from Videos in the Wild. **arXiv preprint arXiv:1212.0402**, 2012.

SOUZA, Marcos Roberto. **Digital Video Stabilization: Algorithms and Evaluation**. Dissertação (Mestrado) — Instituto de Computação, Universidade Estadual de Campinas, Campinas, Brazil, 2018.

SUN, Shuyang; KUANG, Zhanghui; OUYANG, Wanli; SHENG, Lu; ZHANG, Wei. Optical flow guided feature: A fast and robust motion representation for video action recognition. **arXiv preprint arXiv:1711.11152**, 2017.

SUN, Shuyang; KUANG, Zhanghui; SHENG, Lu; OUYANG, Wanli; ZHANG, Wei. Optical flow guided feature: a fast and robust motion representation for video action recognition. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2018. p. 1390–1399.

SZEGEDY, Christian; VANHOUCKE, Vincent; IOFFE, Sergey; SHLENS, Jon; WOJNA, Zbigniew. Rethinking the Inception Architecture for Computer Vision. In: **IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2016. p. 2818–2826.

TACON, Hemerson; BRITO, André S; CHAVES, Hugo L; VIEIRA, Marcelo Bernardes; VILLELA, Saulo Moraes; MAIA, Helena de Almeida; CONCHA, Darwin Ttito; PEDRINI, Helio. Human action recognition using convolutional neural networks with symmetric time extension of visual rhythms. In: SPRINGER. **International Conference on Computational Science and Its Applications**. [S.l.], 2019. p. 351–366.

TORRES, Berthin S; PEDRINI, Helio. Detection of Complex Video Events through Visual Rhythm. **The Visual Computer**, Springer, p. 1–21, 2016.

TRAN, Du; BOURDEV, Lubomir; FERGUS, Rob; TORRESANI, Lorenzo; PALURI, Manohar. Learning spatiotemporal features with 3d convolutional networks. In: **Proceedings of the IEEE International Conference on Computer Vision**. [S.l.: s.n.], 2015. p. 4489–4497.

TURAGA, P.; CHELLAPPA, R.; SUBRAHMANIAN, V. S.; UDREA, O. Machine Recognition of Human Activities: A Survey. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 18, n. 11, p. 1473–1488, 2008.

ULLAH, Amin; AHMAD, Jamil; MUHAMMAD, Khan; SAJJAD, Muhammad; BAIK, Sung Wook. Action recognition in video sequences using deep bi-directional LSTM with CNN features. **IEEE Access**, IEEE, v. 6, p. 1155–1166, 2018.

VALIO, Felipe Braunger; PEDRINI, Helio; LEITE, Neucimar Jeronimo. Fast Rotation-Invariant Video Caption Detection based on Visual Rhythm. In: SPRINGER. **Iberoamerican Congress on Pattern Recognition**. [S.l.], 2011. p. 157–164.

VAROL, Gül; LAPTEV, Ivan; SCHMID, Cordelia. Long-term temporal convolutions for action recognition. **Transactions on Pattern Analysis and Machine Intelligence**, IEEE, v. 40, n. 6, p. 1510–1517, 2017.

WANG, B.; SUN, Y.; XUE, B.; ZHANG, M. Evolving Deep Convolutional Neural Networks by Variable-Length Particle Swarm Optimization for Image Classification. In: **2018 IEEE Congress on Evolutionary Computation (CEC)**. [S.l.: s.n.], 2018. p. 1–8.

WANG, Heng; KLÄSER, Alexander; SCHMID, Cordelia; CHENG-LIN, Liu. Action Recognition by Dense Trajectories. In: **CVPR 2011 - IEEE Conference on Computer Vision & Pattern Recognition**. Colorado Springs, United States: IEEE, 2011. p. 3169–3176.

WANG, Heng; KLÄSER, Alexander; SCHMID, Cordelia; LIU, Cheng-Lin. Dense trajectories and motion boundary descriptors for action recognition. **International Journal of Computer Vision**, Springer, v. 103, n. 1, p. 60–79, 2013.

WANG, Heng; SCHMID, Cordelia. Action Recognition with Improved Trajectories. In: **The IEEE International Conference on Computer Vision (ICCV)**. [S.l.: s.n.], 2013.

WANG, Hao; YANG, Yanhua; YANG, Erkun; DENG, Cheng. Exploring Hybrid Spatio-Temporal Convolutional Networks for Human Action Recognition. **Multimedia Tools and Applications**, Springer, v. 76, n. 13, p. 15065–15081, 2017.

WANG, Jue; CHERIAN, Anoop; PORIKLI, Fatih; GOULD, Stephen. Video representation learning using discriminative pooling. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2018. p. 1149–1158.

WANG, Liangliang; GE, Lianzheng; LI, Ruifeng; FANG, Yajun. Three-stream CNNs for Action Recognition. **Pattern Recognition Letters**, Elsevier, v. 92, p. 33–40, 2017.

WANG, Limin; QIAO, Yu; TANG, Xiaoou. Action recognition with trajectory-pooled deep-convolutional descriptors. In: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2015. p. 4305–4314.

WANG, Limin; XIONG, Yuanjun; WANG, Zhe; QIAO, Yu. Towards Good Practices for very Deep Two-Stream Convnets. **arXiv preprint arXiv:1507.02159**, 2015.

WANG, Limin; XIONG, Yuanjun; WANG, Zhe; QIAO, Yu; LIN, Dahua; TANG, Xiaoou; GOOL, Luc Van. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. In: SPRINGER. **European Conference on Computer Vision**. [S.l.], 2016. p. 20–36.

WANG, Mei; DENG, Weihong. Deep face recognition: A survey. **arXiv preprint arXiv:1804.06655**, 2018.

WANG, X.; GAO, L.; SONG, J.; SHEN, H. Beyond Frame-level CNN: Saliency-Aware 3-D CNN with LSTM for Video Action Recognition. **IEEE Signal Processing Letters**, v. 24, n. 4, p. 510–514, April 2017. ISSN 1070-9908.

WANG, Yunbo; LONG, Mingsheng; WANG, Jianmin; YU, Philip S. Spatiotemporal Pyramid Network for Video Action Recognition. In: IEEE. **IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.], 2017. p. 2097–2106.

WANG, Yang; TRAN, Vinh; HOAI, Minh. Eigen evolution pooling for human action recognition. **arXiv preprint arXiv:1708.05465**, 2017.

WEINLAND, Daniel; RONFARD, Remi; BOYER, Edmond. A survey of vision-based methods for action representation, segmentation and recognition. **Computer Vision and Image Understanding**, Elsevier, v. 115, n. 2, p. 224–241, 2011.

WU, Zuxuan; JIANG, Yu-Gang; WANG, Xi; YE, Hao; XUE, Xiangyang; WANG, Jun. Fusing multi-stream deep networks for video classification. **arXiv preprint arXiv:1509.06086**, 2015.

YE, Hao; WU, Zuxuan; ZHAO, Rui-Wei; WANG, Xi; JIANG, Yu-Gang; XUE, Xiangyang. Evaluating Two-Stream CNN for Video Classification. **arXiv preprint arXiv:1504.01920**, 2015.

YU, Dingjun; WANG, Hanli; CHEN, Peiqiu; WEI, Zhihua. Mixed pooling for convolutional neural networks. In: MIAO, Duoqian; PEDRYCZ, Witold; SLEZAK, Dominik; PETERS, Georg; HU, Qinghua; WANG, Ruizhi (Ed.). **Rough Sets and Knowledge Technology**. Cham: Springer International Publishing, 2014. p. 364–375.

ZACH, Christopher; POCK, Thomas; BISCHOF, Horst. A duality based approach for realtime TV-L 1 optical flow. In: SPRINGER. **Joint Pattern Recognition Symposium**. [S.l.], 2007. p. 214–223.

ZHANG, Richard; ISOLA, Phillip; EFROS, Alexei A. Colorful image colorization. **arXiv preprint arXiv:1603.08511**, 2016.

ZHU, Jiagang; ZHU, Zheng; ZOU, Wei. End-to-end video-level representation learning for action recognition. In: **2018 24th International Conference on Pattern Recognition (ICPR)**. [S.l.: s.n.], 2018. p. 645–650.

ZHU, Wangjiang; HU, Jie; SUN, Gang; CAO, Xudong; QIAO, Yu. A Key Volume Mining Deep Framework for Action Recognition. In: **IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2016. p. 1991–1999.

ZHU, Yi; LI, Xinyu; LIU, Chunhui; ZOLFAGHARI, Mohammadreza; XIONG, Yuanjun; WU, Chongruo; ZHANG, Zhi; TIGHE, Joseph; MANMATHA, R; LI, Mu. A comprehensive study of deep video action recognition. **arXiv preprint arXiv:2012.06567**, 2020.