UFES

Universidade Federal do Espírito Santo
Centro Tecnológico - Departamento de Engenharia Elétrica
Programa de Pós-Graduação em Engenharia Elétrica

Sistema de Reconhecimento de Gestos e Ações em Tempo Real Baseado em Visão Computacional

Clebeson Canuto dos Santos

Vitória-ES, Dezembro de 2020

Clebeson Canuto dos Santos

Sistema de Reconhecimento de Gestos e Ações em Tempo Real Baseado em Visão Computacional

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para a obtenção do título de Doutor em Engenharia Elétrica na área de concentração Robótica, Controle e Automação.

Universidade Federal do Espírito Santo – UFES

Centro Tecnológico

Programa de Pós-Graduação em Engenharia Elétrica

Orientadora: Prof[a]. Dr[a]. Raquel Frizera Vassallo

Co-orientador: Prof. Dr. José Alberto Rosado dos Santos-Victor

Vitória-ES

Dezembro de 2020

Clebeson Canuto dos Santos

Sistema de Reconhecimento de Gestos e Ações em Tempo Real Baseado em Visão Computacional

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para a obtenção do título de Doutor em Engenharia Elétrica na área de concentração Robótica, Controle e Automação.

Aprovada em 16 de Dezembro de 2020:

COMISSÃO EXAMINADORA

Profª. Drª. Raquel Frizera Vassallo
Universidade Federal do Espírito Santo
Orientadora

Prof. Dr. José Alberto Rosado dos Santos-Victor
Instituto Superior Técnico da Universidade de Lisboa
Co-orientador

Prof. Dr. Alexandre José Malheiro Bernardino
Instituto Superior Técnico da Universidade de Lisboa
Membro Externo

Prof. Dr. Jugurta Rosa Montalvão Filho
Universidade Federal de Sergipe
Membro Externo

Prof. Dr. Douglas Almonfrey
Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo
Membro Externo

Prof. Dr. Patrick Marques Ciarelli
Universidade Federal do Espírito Santo
Membro Interno

À minha família por sempre me apoiar em todas as decisões. Com uma menção especial ao meu avô, Senhor Zito, que não está mais entre nós, e ao meu sobrinho, Heitor, que acabou de chegar.

AGRADECIMENTOS

Se eu tivesse que resumir essa caminhada em apenas uma palavra, seria "resiliência". Na carreira acadêmica, as derrotas são inúmeras, porém, as conquistas valem a pena. As centenas de noites viradas lendo artigos e implementando códigos, adicionadas aos milhares de experimentos que não deram certo, provocam um grande desgaste psicológico. A autopressão para alcançar resultados melhores faz com que esse caminho seja ainda mais árduo, sendo que as crises de ansiedade passam a fazer parte do cotidiano. Apesar de tudo, essa carreira oferece inúmeros momentos de felicidade, sendo o prazer da busca contínua pelo conhecimento e o reconhecimento pelo trabalho alguns deles. O estresse da pesquisa e da escrita de artigos científicos são compensados com uma frase simples: Parabéns, seu trabalho foi aceito para publicação! As milhares de horas estudando, preparando experimentos e analisando resultados são compensadas com um convite para participar de uma banca, para ser orientador de aluno ou mesmo ministrar um curso. Por isso, eu escolhi a carreira acadêmica, e hoje estou aqui, tentando expressar meus agradecimentos a todos que, de forma direta ou indireta, participaram dessa minha caminhada. Foram muitos os tropeços e muitas as quedas, porém, sempre contei com pessoas que me levantaram ou mesmo que me seguraram em todo o caminho.

Sendo assim, agradeço primeiramente aos meus pais, Nailda Farias e José Canuto, por nunca me deixarem desistir. A infância não foi fácil, e mesmo sem muita instrução formal, uma frase que fazia parte das conversas em família era: estude para ser gente. Antes eu não sabia o que isso queria dizer, apenas sabia que deveria seguir o que eles me diziam. Hoje sei que o conhecimento abre vários caminhos e oportunidades, e fornece o pensamento crítico necessário à busca pela dignidade que nos faz sentir como "gente"; que nos faz ser verdadeiramente livres. Como disse o professor Mário Sérgio Cortella: "o que importa é saber o que importa". Gratidão aos meus pais por isso. Estendo esses agradecimentos ao meu irmão, Cledisson Canuto, às minhas irmãs, Tays e Cláudia Canuto, à minha sobrinha, Hevylen e ao meu sobrinho, Heitor.

Agradeço à minha esposa, Raílly Vanessa, por me apoiar incondicionalmente, tendo que desistir dos seus próprios sonhos para viver os meus. Sou grato por você me dar o valor que muitas das vezes eu penso não ter. Inclusive, agradeço por não ter ficado chateada quando, em plena lua de mel, tive que ficar um dia inteiro na frente do computador para terminar e enviar o meu projeto de doutorado.

À minha família, os Farias, em especial à minha avó, Dona Maria, e ao meu avô, Senhor Zito (que descanse em paz). Um exemplo de união e amor. Meu caráter foi moldado pelo exemplo de vocês. Sou muito orgulhoso por tê-los em minha vida. Às minhas tias,

tios e primos por todo apoio.

Agradeço à minha orientadora, Raquel Vassallo, por todo o empenho, discussões e ensinamentos, tanto acadêmicos quanto de vida. Tenho muito orgulho em tê-la como orientadora, você que é uma mulher, professora e orientadora como poucas pessoas. Não tenho dúvidas quanto à sua importância para que eu conseguisse chegar aqui. Você foi mais que uma orientadora, na maioria das vezes, foi uma grande amiga e conselheira. Estendo esses agradecimentos ao seu marido, Alexandre do Carmo, pelas discussões, carinho e confiança durante todos esses anos.

Agradeço ao meu co-orientador, José Santos-Victor, por me receber em seu laboratório e por me lembrar da importância do método científico em um trabalho acadêmico. Cada conversa, cada reunião era uma tonelada de conhecimento e experiência que eu recebia e tentava digerir durante vários dias.

Agradeço também aos meus co-orientadores informais, Jorge Samatelo e Plínio Moreno. Vocês, sem dúvida, tiveram que me aturar, aturar a minha ignorância em determinados assuntos e me mostraram o caminho das pedras, principalmente, no que tange à área de *machine learning*. Hoje ainda não sei muita coisa dessa área, mas sabia muito menos, e não me achava capaz de saber tanto. Foi Jorge, com a sua empolgação, o seu conhecimento e sua vontade de ensinar, que me fez pensar ser capaz e resolver mergulhar nesse imenso oceano. Agradeço pelas inúmeras reuniões e discussões, e pela disponibilidade durante a semana, sábados, domingos e feriados.

Agradeço agora aos meus amigos (autarquias), em especial, aos meus *partners*, Ricardo Salvador e Flávio Machado. Conheço vocês desde o início do doutorado e pretendo cultivar essa amizade para sempre. A Ricardo eu quero agradecer por todos os conselhos e discussões sobre a vida pessoal e profissional, além de organizar os melhores *fires*. Agradecer inclusive, por ter sido meu "piloto de fuga" um dia que precisei ser levado às pressas ao hospital; te devo essa. A Flávio, eu agradeço por ter dado a mim a oportunidade de experimentar o que é ser orientador, e por me fazer ser mais crítico em meus pensamentos e em minhas escolhas. Cada conversa nossa, que costumam ser bem longas, levanta novas reflexões que me fazem evoluir como pessoa. Agradeço a Felippe Mendonça, meu primeiro amigo no Espírito Santo. A sua solicitude foi de grande importância para a minha adaptação. Agradeço por você ser o amigo engenheiro, mecânico, marceneiro, carregador de mudanças e por ter auxiliado Ricardo na minha ida ao hospital. Sou grato também a Rodolfo Picoreti por todos os ensinamentos, discussões e momentos de descontração.

Em geral, agradeço aos meus amigos de laboratório: Bananal, Wítalo, Vinícius, Pedro, Bruna, Thaís Pedruzzi, Demuth, Matheus, Ricardo MLT, Bento, Wagner, Messi, Mariana e muitos outros que por lá passaram nos últimos quatro anos. Vocês fizeram meus dias parecerem bem mais curtos com todos os momentos de discussões e descontração. Vocês também são minhas autarquias. Sou grato também aos companheiros de laboratório

em Portugal: Heitor, João Avelino, Paul Schydlo, Hugo, Min, Dimitris, Ana Luísa, Atabak, Pedro Vicente e Nuno Duarte. Vocês me receberam com muito carinho no Vislab e fizeram com que eu me sentisse em casa.

A todos os professores do PPGEE pelos ensinamentos. Em especial à coordenação por ser tão solicita em meus pedidos, e à secretaria, na pessoa de Aline Amaral, pelo enorme profissionalismo e pelo empenho na resolução de todas as questões levantadas.

À equipe de limpeza, tando da Ufes quanto do ISR, por proporcionar-me um ambiente de trabalho sempre limpo e agradável.

Agradeço também ao apoio financeiro dado pela Fapes e pela Capes por meio das bolsas concedidas. Muitos não sabem a importância de uma bolsa. Eu sei! Sei que sem elas eu provavelmente não estaria aqui concluindo esses agradecimentos.

Por fim, a todos que, de forma direta ou indireta, contribuíram para a minha formação pessoal e acadêmica.

"Sonhar é verbo, é seguir,
é pensar, é inspirar,
é fazer força, insistir,
é lutar, é transpirar.
São mil verbos que vêm antes
do verbo realizar."
(Bráulio Bessa)

RESUMO

Esta tese tem como objetivo investigar e propor mecanismos de reconhecimento e antecipação de gestos dinâmicos e ações baseando-se apenas em visão computacional. Três propostas objetivaram o reconhecimento de gestos: *Star RGB* - uma representação de movimento que condensa os *frames* de um vídeo em uma imagem RGB; *Star iRGB* - uma versão iterativa da *Star RGB* que pode ser usada por modelos de aprendizagem de natureza sequencial; e *Star* $iRGB_{hand}$ - um modelo iterativo para o reconhecimento de gestos que utiliza a forma das mãos como contexto. Para a antecipação de ações, foram apresentados modelos bayesianos, baseados em redes neurais recorrentes, que usam informações de contexto para diminuir a ambiguidade entre movimentos semelhantes, além de um limiar sobre a incerteza epistêmica estimada como mecanismo de tomada de decisão quanto ao momento da antecipação. Nesse contexto, foram propostos dois modelos para reconhecer e antecipar gestos de forma *online*. Todas as propostas foram validadas por meio de diversos experimentos cujos resultados foram comparados a vários *baselines*. Nesse sentido, foram utilizados três conjuntos de dados principais: o Montalbano, para os gestos capturados por apenas uma câmera; o IS-Gesture, para gestos capturados em um ambiente multicâmeras; e o Acticipate, para a antecipação de ações. Os resultados alcançados com os modelos para reconhecimento de gestos foram os melhores para o conjunto Montalbano quando se consideram os trabalhos que utilizam apenas imagens RGB. Mesmo quando comparados aos modelos multimodais, baseados em CNN 3D, os resultados estão entre os melhores, ficando levemente atrás (menos de 1%) de apenas duas propostas multimodais. Na tarefa de antecipação de ações, as acurácias de reconhecimento e antecipação obtidas sobre o Acticipate são as melhores alcançadas nesse conjunto de dados até o presente momento. Finalmente, considerando os modelos que objetivam reconhecer e antecipar os gestos de modo *online*, para o Montalbano, o modelo proposto também conseguiu resultados entre os melhores da literatura. Já em relação ao IS-Gesture, o qual representa o desafio de maior complexidade devido ao ambiente multicâmeras, as acurácias médias de reconhecimento e antecipação dos gestos foram consideradas satisfatórias, havendo ainda indícios claros de onde devem ser realizadas melhorias para se atingir melhores resultados. Quanto ao tempo de execução, os modelos propostos mostraram-se viáveis para fornecer informações para uma aplicação que demanda uma taxa de atualização de até 10 FPS. Assim, é possível a utilização de tais modelos em uma aplicação interacional em tempo real, em um ambiente com uma ou várias câmeras. Em resumo, todas as propostas mostraram-se bem promissoras, além de obterem resultados que ultrapassam os principais trabalhos da literatura que abordam os conjuntos de dados anteriormente mencionados.

Palavras-chave: reconhecimento de gestos dinâmicos, antecipação de ações, interação humano-máquina, redes neurais profundas, visão computacional.

ABSTRACT

This thesis aims to investigate and propose mechanisms for recognizing and anticipating dynamic gestures and actions based only on computer vision. Three proposals are focused on gesture recognition: Star RGB - a representation that condenses the montion contained in the frames of a video into only one RGB image; Star iRGB - an iterative version of Star RGB that can be used by learning models of sequential nature; and Star iRGB$_{hand}$ - an iterative model for recognizing gestures that uses the shape of the hands as context. For action anticipation, bayesian models based on recurrent neural networks were presented, which uses context information to reduce the ambiguity between similar movements in addition to a threshold on the estimated epistemic uncertainty to decide when an action should be anticipaded. In this context, two models have been proposed to recognize and anticipate gestures online. All proposals were validated through several experiments whose results were compared to several baselines. In this sense, three main datasets were used: Montalbano, for gestures captured by only one camera; IS-Gesture, for gestures captured in a multi-camera environment; and Acticipate, for action anticipation. The results achieved with the gesture recognition models were the best for the Montalbano set when considering works that use only RGB images. Even when compared to multimodal models, based on CNN 3D, the results are among the best, just slightly behind (less than 1%) two multimodal proposals. In the task of anticipating actions, the accuracy of recognition and anticipation obtained when using the dataset Acticipate were the best ones achieved so far. Finally, considering the models that aim to recognize and anticipate gestures online, the proposed model that works with only one camera has also achieved results among the best in literature for the Montalbano dataset. In relation to IS-Gesture, which represents the most complex challenge due to the multi-camera environment, the average accuracy of recognition and anticipation of gestures was considered satisfactory, with clear indications of where improvements should be made to achieve better results. Regarding the execution time, the proposed models were all able to provide information for an application that requires a frame rate of up to 10 FPS. Thus, it is possible to use such models in an interactive application in real time, in an environment with one or several cameras. In summary, all the proposals have shown to be very promising, obtaining results that go beyond the main related works that address the previously mentioned datasets.

Keywords: dynamic gesture recognition, action anticipation, human-robot interaction, deep learning, computer vision.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

BBB	*Bayes By BackProp*
BNN	*Bayesian Neural Networks* (Redes neurais bayesianas)
CGM	*Convolutional Graphical Models* (Gráficos convolucionais)
CNN	*Convolutional Neural Networks* (Redes neurais convolucionais)
CONV-1D	Convolução unidimensional
DFNN	*Deep Feedforward Neural Networks* (Rede neural profunda sem realimentação)
DL	*Deep Learning* (Aprendizado profundo)
FC	*Fully Connecteds* (Redes totalmente conectadas)
FCN	*Fully Convolutional Networks* (Redes completamente convolucionais)
FPS	Taxa de de exibição de um vídeo em Frames Por Segundo
FPR	*False-Positive Rate* (Taxa de falsos-positivos)
GAN	*Generative Adversarial Networks* (Redes adversativas geradoras)
GAP	*Global Average Pooling* (Operador que calcula a média espacial de um mapa de características)
GRIT	*Gesture Commands for Robot InTeracton*) (Conjunto de dados gestuais para a interação com robôs)
GRU	*Gated Recurrent Unit*
HMI	*Human-Machine Interaction* (Interação humano-máquina)
HMM	*Hidden Markov Model* (Modelos ocultos de Markov)
Ifes	Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo
IST	Instituto Superior Técnico da Universidade de Lisboa
LSTM	*Long-Short Term Memory* (Memória de longo e curto prazo)
MC	*Monte Carlo Simulation* (Simulação de Monte Carlo)
MHI	*Moviment History Image* (Imagem do histórico de movimento)

ML *Machine Learning* (Aprendizado de máquinas)

MLP *Multilayer Perceptron* (Perceptron multicamadas)

MoCap *Motion Capture* (Captura de movimento)

MRF *Markov Random Field* (Campos aleatórios de Markov)

PARAFAC *Parallel FactorAnalysis* (Decomposição de posto tensorial)

RNN *Recurrent Neural Networks* (Redes neurais recorrentes)

RPC *Remote Procedure Call* (Chamada de procedimento remoto)

SGD *Stochastic Gradient Descent* (Gradiente descendente estocástico)

SVM *Support Vector Machine* (Máquinas de vetores de suporte)

TL *Transfer Learning* (Transferência de aprendizado)

TPR *True-Positive Rate* (Taxa de verdadeiros-positivos)

TMO Taxa Média de Observação

Ufes Universidade Federal do Espírito Santo

VD *Variational Dropout*

VI *Variational Inference* (Inferência variacional)

VisLab Laboratório de Visão Computacional e Robótica do Instituto Superior Técnico

SUMÁRIO

1 INTRODUÇÃO

> A ciência vai muito além da sua mera prática. Por trás das fórmulas complicadas, das tabelas de dados experimentais e da linguagem técnica, encontra-se uma pessoa tentando transcender as barreiras imediatas da vida diária, guiada por um insaciável desejo de adquirir um nível mais profundo de conhecimento e de realização própria.
>
> Marcelo Gleiser

1.1 Motivação e objeto de pesquisa

Os seres humanos têm a capacidade natural de interagir uns com os outros e realizar tarefas conjuntas. Parte dessa capacidade se deve à habilidade de perceber o ambiente e reconhecer padrões que os ajudam a interagir, reconhecendo ou antecipando ações e intenções, e assim, tomar as decisões necessárias. Com o aumento da capacidade de processamento dos dispositivos eletrônicos e a oferta de aplicações cada vez mais adaptadas a diferentes públicos, as máquinas estão cada dia mais presentes na vida cotidiana das pessoas. No entanto, para que essa participação tenha maior efetividade, é preciso que elas também possuam essa capacidade de interação, reconhecimento e antecipação inerente ao ser humano.

Nesse sentido, pela diferença na maneira como as pessoas e as máquinas interpretam as informações, é preciso, primeiramente, determinar uma interface que facilite a comunicação entre elas. Assim, como a intenção é inserir as máquinas no cotidiano das pessoas de forma a não causá-las estranheza, é importante que tal interface seja a mais intuitiva possível. Dessa forma, definida a interface, deve-se então dotar a máquina da inteligência necessária para interpretá-la, ou, no caso dos robôs humanoides, também de executá-las.

Na Interação Humano-Máquina (HMI - do inglês *Human-Machine Interaction*), várias interfaces de comunicação podem ser usadas, como controles manuais, dispositivos de leitura cerebral, fala, gestos, dentre outras. Particularmente, gestos e fala são as alternativas menos intrusivas e mais naturais para os seres humanos. Entre a fala e os gestos, há muitos casos em que os gestos são preferidos ao invés da fala, uma vez que, com exceção às linguagens de sinais, eles não exigem um componente gramatical obrigatório para sua elaboração (MCNEILL, 1992).

Os gestos podem ser classificados como estáticos, ou seja, não há mudança em sua forma (pose) ao longo do tempo; ou dinâmicos, caso em que existe um conjunto de poses que variam dentro de um intervalo de tempo. De acordo com McNeill (1992), os gestos naturais, os mais intuitivos, são principalmente dinâmicos e realizados na maioria das vezes pelas mãos. Portanto, ao tentar usar efetivamente os gestos para a HMI, os estáticos desempenham um papel menos significativo. Dessa forma, por considerar a intuitividade na interação um aspecto importante, um dos focos de estudo do presente trabalho, além das ações, serão os gestos dinâmicos. Sendo assim, neste manuscrito, o substantivo gesto, quando não adjetivado, estar-se-á referindo-se à sua forma dinâmica.

Após a escolha dos gestos como a interface de comunicação, deve-se eleger os mecanismos artificiais que, não só os interpretem e/ou os executem, mas também permitam que as máquinas percebam padrões humanos, mesmo quando da ausência de comunicação, como é o caso das ações realizadas de maneira isolada ou conjunta. Tais mecanismos irão depender de dois principais fatores: ter uma definição clara do que difere uma ação de um gesto, e quais os tipos de dados podem ser utilizados para representá-los.

No contexto de interação e colaboração, uma ação pode ser definida como uma sequência de movimentos realizados a fim de alcançar um objetivo. Já um gesto pode ser definido como um conjunto de pequenos movimentos que objetivam a comunicação de algo. Ou seja, o gesto é uma ação cujo objetivo a ser alcançado é a comunicação. Para ambos os casos, o termo movimento representa o deslocamento de um objeto no espaço através do tempo. Sendo que, para as ações no geral, os objetos podem ser quaisquer entidades físicas, e, para os gestos, são comumente restritos a partes do corpo, como mãos, braços, cabeça, olhos, dentre outros.

Uma vez definidos os conceitos de gestos e ações, parte-se para os dados que podem representá-los. Desde o advento do Microsoft Kinect[1], em meados de 2011, e de outros sucessores, como *Asus Xtion Pro*[2] ou *Intel Realsense*[3], existe uma tendência na utilização de sensores multimodais para o reconhecimento de gestos e ações. Isto se dá devido à possibilidade de captura não só de imagens RGB, mas também do esqueleto e dados de profundidade de vários indivíduos presentes no ambiente. Oferecendo, assim, informações resumidas e precisas sobre os seus movimentos. Com isso, diferentes trabalhos têm utilizado tais sensores a fim de melhorar suas taxas de reconhecimento. A esse variado conjunto de informações é dado o nome de multimodal, e ele pode servir para alimentar tanto modelos distintos, onde cada tipo de informação alimenta um modelo específico, quanto um único modelo, onde todas os tipos de informações são tratadas e fundidas por apenas um modelo, como o apresentado em (NEVEROVA et al., 2016).

[1] <https://msdn.microsoft.com/en-us/library/hh438998.aspx>
[2] <https://www.asus.com/3D-Sensor/Xtion_PRO/>
[3] <https://click.intel.com/realsense.html>

Mesmo oferecendo vantagens significativas, essa dependência de sensores específicos causa uma restrição aos ambientes interacionais onde tais soluções podem ser implantadas. Assim, ambientes que possuem apenas câmeras RGB, como as câmeras de vigilância, facilmente encontradas em muitos espaços públicos ou privados, não podem ser considerados por tais modelos de reconhecimento. Esse é um dos motivos pelos quais alguns estudos, como os descritos neste trabalho, estão focados no desenvolvimento de métodos de reconhecimento baseados apenas em informações visuais. Sendo assim, as imagens RGB são escolhidas aqui como a fonte de informação a ser utilizada pelas máquinas, o que situa este trabalho na área de visão computacional.

Ao adaptar as definições anteriores para o problema de reconhecimento e antecipação de ações ou gestos por visão computacional, tem-se que: o movimento é representado pela variação de intensidades em determinadas áreas de imagens consecutivas. Assim, gestos e ações podem ser representados por variações de intensidades em partes específicas de uma sequência de imagens. Logo, para tentar resolver o problema por meio de visão computacional, o dado de entrada de um modelo de aprendizagem de máquina pode ser: (*i*) um vídeo, seja ele representado por uma sequência de imagens em um arquivo único e compactado, ou por uma sequência de imagens em arquivos individuais; ou (*ii*) um *stream* (fluxo) de vídeo, onde as imagens são fornecidas uma a uma por uma fonte de dados específica. A diferença entre os dois tipos de dados é a disponibilidade das imagens. Enquanto no vídeo é possível acessar todas as imagens num mesmo instante de tempo, no *stream*, o acesso a cada imagem é dado seguindo uma determinada taxa de fornecimento.

1.2 Problemática

Devido à sua importância para a HMI, têm-se cada vez mais estudos focados no desenvolvimento de sistemas e modelos eficientes de reconhecimento de gestos. Uma lista dos reconhecedores comumente usados pode ser vista em Mitra e Acharya (2007), Rautaray e Agrawal (2015) e Saikia e Saharia (2016). Alguns desses trabalhos usam mais de um tipo de informação para obter melhores resultados, os quais são chamados de métodos multimodais (NEVEROVA et al., 2016). Eles geralmente utilizam informações de cores (imagens no formato RGB), medições de profundidade e articulações do esqueleto para detectar e reconhecer gestos. Essas informações oriundas de várias fontes, como juntas de esqueletos e profundidade, complementam positivamente as informações de cores, o que facilita a tarefa de separação de classes realizada pelo reconhecedor (ESCALERA; ATHITSOS; GUYON, 2017).

No entanto, como dito antes, o uso de dados multimodais pode restringir a aplicação das soluções propostas a ambientes com sensores específicos que não as câmeras RGB, cada vez mais ubiquamente distribuídas. Porém, é importante salientar que vários desses

trabalhos não desprezam a importância das câmeras a cores. O que se tem é um *trade-off* entre a restrição de uso dos modelos que utilizam informações multimodais e a dificuldade de treiná-los utilizando apenas informações de cor. Esse problema advém da alta dimensionalidade das imagens. Por exemplo, considerando uma imagem em baixa resolução de 240×320 píxeis, e uma sequência média de 60 *frames* (quadros) por gesto ou ação, têm-se dados de entrada com uma dimensionalidade média de 13.824.000 valores de intensidade. Por outro lado, usando uma imagem de profundidade é possível extrair características dinâmicas, como o movimento do esqueleto virtual de uma pessoa, e reduzir a dimensão dos dados de entrada em centenas de ordem de grandeza. Com isso, facilita-se o treinamento do modelo, ao mesmo tempo que se reduz o custo computacional por ele demandado. Dessa forma, uma possibilidade para usar visão computacional de maneira mais eficiente neste tipo de problema, é reduzir significativamente a dimensionalidade dos dados, seja por meio de extração de características individuais de cada pessoa, como o seu esqueleto virtual, seja por outra representação compacta da ação ou gesto presente na sequência de imagens.

Objetivando a representação de gestos em vídeos, alguns trabalhos têm obtido resultados significativos, como em Barros et al. (2014b) e Tsironi et al. (2017). No entanto, os bons resultados por eles mencionados são alcançados usando vocabulários de gestos específicos, que contêm gestos significativamente diferentes entre si, o que, geralmente, simplifica a tarefa de reconhecimento. Assim, o primeiro problema aqui encontrado é o de obter uma representação capaz de melhorar o reconhecimento de gestos baseando-se apenas em informações de cor, que possa lidar com gestos de diferentes classes, não facilmente distinguíveis entre si e contidos em sequências de imagens de tamanho variável.

Outro aspecto importante a ser considerado é que, não necessariamente, duas sequências de imagens que descrevem dois movimentos idênticos representam um mesmo gesto ou ação. Para exemplificar, considere-se uma sequência de imagens que define um gesto "U", no qual a mão de uma pessoa vai da direita para a esquerda de maneira repetida. Considere-se agora uma outra sequência de imagens onde um relógio de pêndulo descreve o mesmo movimento realizado pela pessoa na primeira sequência. Se o modelo considerasse apenas o movimento extraído da sequência de imagens, a pessoa e o relógio, ambos estariam realizando o gesto "U". No entanto, é claro que isso não pode ser verdade, já que um relógio não gesticula. O problema seria solucionado por meio da análise do contexto no qual o movimento foi realizado. Neste caso, por meio da determinação de quem o estava realizando, ou a qual objeto ele se referia (mão ou pêndulo), seria possível distinguir o que era e o que não era gesto. Outro problema é encontrado quando se tenta reconhecer/antecipar ações e gestos em vídeos por meio apenas da informação de movimento, uma vez que esse tipo de informação pode ser ambígua.

Por fim, apesar da possibilidade de reconhecer gestos ou ações em vídeos, quando

o objetivo é utilizar a solução em um ambiente real, onde as imagens são fornecidas na forma de *stream*, deve-se optar por uma das seguintes abordagens:

1. Para reconhecimento, tarefa que utiliza toda a sequência de imagens contendo uma ação ou um gesto, como o *stream* não fornece um vídeo já segmentado, deve-se utilizar alguma técnica de segmentação (também conhecida como *spotting*), capaz de detectar quando um gesto ou ação começa e termina, armazenar a sequência de imagens fornecida nesse intervalo, e assim, reconhecê-lo(a). O problema é ter de esperar toda a sequência de imagens para poder tomar uma decisão. Já que, como dito em Godøy e Leman (2010), muitas das vezes não é simples precisar quando uma ação ou gesto começa e termina. Isso pode fazer com que a decisão seja postergada de maneira indefinida, ou seja tomada quando o gesto não esteja sendo executado.

2. Para a antecipação, a cada imagem, ou a cada janela de imagens recebida, prediz-se uma classe para as possíveis ações. Sendo assim, é possível realizar o reconhecimento antes mesmo que a sequência de imagens esteja completa. Dessa forma, pode-se tomar uma decisão de maneira mais rápida, o que é de extrema importância quando se deseja uma resposta rápida do ambiente ou do agente ao qual a interação se destina. Mesmo sendo a maneira natural com a qual os seres humanos interagem e se comunicam, para poder antecipar uma ação é necessário aumentar a confiança de que as próximas imagens da sequência representarão a ação predita. Para isso, é necessário diminuir a incerteza sobre a predição da ação em execução, dado o conhecimento acumulado por meio das imagens já observadas. No entanto, como os principais modelos aplicados aos problemas de visão computacional têm uma natureza determinística, a probabilidade estimada não corresponde à medida de incerteza aproximada de que as imagens observadas até um determinado instante representam uma classe específica de ação. O mesmo vale para a antecipação de gestos.

1.3 Hipóteses

Baseados nos problemas anteriormente citados, algumas hipóteses são levantadas:

1. Por meio de uma representação compacta de movimento de um vídeo, um método baseado apenas em imagens RGB pode ser tão efetivo para o reconhecimento de gestos dinâmicos quanto os métodos multimodais são.

2. A escolha empírica da informação de contexto pode ser uma alternativa efetiva para distinguir diferentes ações ou gestos representados por movimentos ambíguos.

3. Para o problema de antecipação, a incerteza sobre a predição é um limiar eficaz e mais confiável do que o valor da probabilidade estimada pelo modelo.

1.4 Soluções propostas

Nos últimos anos, o aprendizado profundo alcançou o estado da arte em diferentes tarefas, como a classificação de imagens (HE et al., 2016a; KRIZHEVSKY; SUTSKEVER; HINTON, 2012a; SIMONYAN; ZISSERMAN, 2014c), processamento de linguagem natural (DEVLIN et al., 2019; VASWANI et al., 2017), reconhecimento de ações/atividades (KARPATHY et al., 2014; SIMONYAN; ZISSERMAN, 2014a; CARREIRA; ZISSERMAN, 2017a) e reconhecimento de gestos (NEVEROVA et al., 2016; NEVEROVA et al., 2014; TSIRONI et al., 2017; BARROS et al., 2014b). Alguns trabalhos, como o de Bilen et al. (2017), Rodriguez, Fernando e Li (2018) e Choutas et al. (2018), representam uma ação estimando o movimento dos atores (usuários) envolvidos. No caso de ações simples e de alta variação interclasse, o movimento pode fornecer informação suficiente para que a tarefa de reconhecimento/antecipação seja bem sucedida. Porém, no caso de ações mais complexas e ambíguas, o movimento pode não ser suficiente. Nesses casos, alguns detalhes durante a antecipação, como a posição dos objetos, a relação entre mãos e objeto/pessoa e o tipo de objeto manipulado, podem oferecer informações que ajudam a caracterizar cada tipo de ação. Dessa forma, usando apenas movimento, o modelo exclui o contexto, uma informação crítica para a tarefa de antecipação/reconhecimento.

Com relação à tarefa de reconhecimento de ação, as abordagens de fluxo duplo (*two streams*) (SIMONYAN; ZISSERMAN, 2014a; CARREIRA; ZISSERMAN, 2017a; KWON et al., 2018) são as mais bem-sucedidas, pois usam o movimento como a principal fonte de informação para descrever cada ação e usam o contexto como uma informação adicional, que ajuda a caracterizar cada classe individualmente. Em tais soluções, o movimento é representado pelo fluxo óptico calculado entre imagens sequenciais, enquanto as imagens representam a informação contextual (BARADEL et al., 2018), que são extraídas de maneira implícita pelas Redes Neurais Convolucionais (CNN - do inglês *Convolutional Neural Network*) (LECUN; BENGIO et al., 1995a). No entanto, para extrair as informações contextuais contidas nas imagens de maneira auto-supervisionada, o procedimento de treinamento das CNNs exige grandes conjuntos de dados para obter resultados satisfatórios. Como consequência, as abordagens de fluxo duplo não são eficazes na resolução de problemas fornecidos por pequenos conjuntos de dados, como os comumente disponibilizados para o treinamento de métodos automáticos aplicados à colaboração humano-humano ou humano-robô.

Analisando as abordagens de *two streams* de um ponto de vista mais estatístico, ao utilizar a imagem por inteiro como fonte de contexto, pelo princípio da Razão Insuficiente

de Bayes-Laplace[4], é coerente assumir que todos os píxeis da imagem têm as mesmas chances de representarem uma informação de contexto relevante para o problema. Isso pode ser visto como a adição de uma priori com distribuição uniforme sobre todos os seus píxeis, o que, possivelmente, dificultará o processo de aprendizagem em pequenos conjuntos de dados. Uma abordagem mais efetiva seria assumir que, de fato, apenas algumas áreas da imagem possuem informações contextuais relevantes para o problema. Ou seja, seria o mesmo que adicionar uma priori infinitamente forte sobre os píxeis da imagem, uma vez que vários deles possuem chances nulas de representarem uma informação de contexto para a solução da tarefa. Em outras palavras, isso é o mesmo que fazer uso do conhecimento humano para determinar a informação de contexto que melhore a tarefa de antecipação/reconhecimento sem prejudicar significativamente a convergência do modelo.

Dessa forma, devido às vantagens no uso do aprendizado profundo na resolução dos problemas de visão computacional e a possibilidade de uma definição mais acertada sobre informações contextuais, as soluções propostas neste trabalho, as quais objetivam adquirir dados que comprovem as hipóteses levantadas, são as seguintes:

- Uma técnica de representação de movimento capaz de melhorar o reconhecimento de gestos dinâmicos, mesmo em situações de baixa variância interclasse.
- Uma variante iterativa da técnica proposta que possa ser utilizada por modelos de natureza sequencial.
- Um modelo de antecipação de ações que usa a informação contextual em adição à de movimento, e a incerteza como limiar de tomada de decisão.
- Um modelo de antecipação e reconhecimento de gestos.
- A combinação das propostas anteriores a fim de alcançar resultados satisfatórios no reconhecimento e antecipação de gestos ou ações de forma *online*.
- Um sistema capaz de antecipar e reconhecer gestos de forma *online* e em tempo real em um ambiente interacional multicâmeras.

As principais condições delimitadoras das propostas apresentadas são:

1. Deve-se considerar que os gestos ou as ações são executados por apenas uma pessoa;
2. As câmeras devem ser estáticas, sendo que, no caso da abordagem multicâmeras, as mesmas devem ainda ser calibradas;
3. Com relação ao usuário, para as abordagens monocâmera, elas devem estar de frente, não acima da cabeça e a uma distância entre 2 e 3 metro de distância;

[4] Esse princípio afirma que, na ausência de evidências suficientemente relevantes, deve-se distribuir a crença igualmente entre todos os possíveis resultados. Pode ser conhecido também como princípio da indiferença: *"If there is noknown reason for predicating of our subject one rather than another of several alternatives, then relatively to suchknowledge the assertions of each of these alternatives have an equal probability"* (PIGOU, 1921).

4. Deve existir pouco deslocamento do corpo da pessoa em relação à câmera quando gestos ou ações estiverem sendo executados. Apenas os membros superiores podem mover-se livremente.

Quaisquer situações que não cumpram todas essas condições fugirão ao escopo desta tese.

1.5 Objetivos

O objetivo principal desse trabalho é o de propor um sistema de reconhecimento de gestos e ações, baseado em visão computacional, que possa ser executado em tempo real e que seja robusto a baixas variações inter-classes. Sendo assim, para alcançar o objetivo principal, os seguintes objetivos específicos foram traçados:

1. Propor uma técnica capaz de representar o movimento presente em uma sequência de imagens.
2. Utilizar a representação proposta para resolver um problema complexo de reconhecimento de gestos dinâmicos.
3. Propor uma versão iterativa de tal representação a fim de utilizar modelos de natureza sequencial para melhorar o reconhecimento de gestos dinâmicos.
4. Testar e validar essa última técnica em uma aplicação a ser executada em tempo real.
5. Mostrar a importância de analisar o problema a ser resolvido a fim de determinar quais tipos de informações são mais relevantes para serem utilizados como fonte de contexto.
6. Propor um modelo capaz de antecipar ações e/ou gestos utilizando a incerteza estimada na predição como limiar de tomada de decisão, e a informação de contexto como complemento à informação de movimento.
7. Validar o modelo proposto por meio de um problema de antecipação de ações.
8. Utilizar as abordagens anteriores a fim de antecipar gestos de modo *online* em tempo real.
9. Propor um sistema que possa reconhecer e antecipar gestos em um espaço inteligente multicâmera.
10. Elaborar gráficos, tabelas e figuras que gerem o lastro científico necessário à comprovação das hipóteses levantadas.

11. Escrever e publicar os resultados obtidos na forma de artigos científicos.

1.6 Materiais e métodos

O presente trabalho caracteriza-se como uma pesquisa de natureza aplicada, com objetivos exploratório-descritivos, cujas análises são realizadas por meio de uma abordagem quantitativa, e de procedimentos bibliográficos e experimentais.

A primeira etapa deste trabalho se deu a partir de um levantamento bibliográfico sobre os trabalhos desenvolvidos para o reconhecimento de gestos em geral. O principal objetivo desta etapa foi contextualizar a problemática, definir os objetos da pesquisa, bem como os conjuntos de dados e métricas de avaliação a serem utilizados. O mesmo procedimento foi realizado para o problema de antecipação de ações.

A partir da delimitação dos objetos da pesquisa, foram realizados novos levantamentos bibliográficos sobre os principais trabalhos que os abordavam, de maneira a coletar o referencial teórico necessário para a construção da base científica da pesquisa. Esta etapa prosseguiu paralelamente às demais, até a elaboração do presente manuscrito. Assim, foi possível mapear os principais trabalhos que surgiram nesta mesma linha de pesquisa. O que colaborou para o enriquecimento das discussões, propostas e resultados aqui apresentados.

Iniciando pelo reconhecimento de gestos, buscou-se uma representação de movimento que pudesse ser utilizada de maneira a aumentar a velocidade de predição de um gesto, sem que fossem comprometidos os resultados das métricas de avaliação. Após selecionar as principais abordagens, experimentos foram realizados a fim de comprovar a efetividade das mesmas quando aplicadas sobre um grande conjunto de dados de reconhecimento de gestos em vídeos. De posse dos resultados, a primeira hipótese foi levantada e pesquisas bibliográficas foram realizadas a fim de confirmar que a mesma ainda não havia sido testada. Assim, as duas primeiras propostas foram apresentadas.

O problema de antecipação de ações foi delimitado durante o período de estágio realizado entre 2018 e 2019 no Laboratório de Visão Computacional (VisLab) do Instituto Superior Técnico (IST) da Universidade de Lisboa, Portugal. Lá, a partir de pesquisas bibliográficas, de discussões com outros pesquisadores e da análise minuciosa de alguns conjuntos de dados, percebeu-se a existência de algumas lacunas deixadas pelas principais propostas que objetivavam contribuir para a solução do problema de antecipação de ações e gestos em vídeos. A primeira lacuna observada foi a falta de discussões claras sobre o problema causado pela ambiguidade quando se usava apenas o movimento como fonte de informação. Tal problema, além de prejudicar significativamente os modelos de reconhecimento, pode ser ainda mais prejudicial em modelos de antecipação, uma vez que

estes tendem ter uma maior incerteza de predição, devido à incompletude das observações. Outra lacuna observada dizia respeito à falta de discussão sobre quais mecanismos poderiam ser empregados para decidir-se quando uma ação ou um gesto deveria ser antecipado em uma aplicação *online*. Sendo assim, a segunda e a terceira hipótese foram levantadas, pesquisas bibliográficas foram feitas, propostas foram apresentadas e experimentos foram realizados.

Objetivando o reconhecimento e a antecipação de gestos em um ambiente real multicâmeras, um trabalho foi desenvolvido a fim de propor uma técnica capaz de obter as coordenadas tridimensionais do esqueleto de uma pessoa presente na cena. Tal técnica baseou-se em geometria epipolar para reconstruir tridimensionalmente o esqueleto de um usuário por meio dos seus esqueletos bidimensionais extraídos das imagens de cada uma das câmeras. Apesar do trabalho não ter sido realizado majoritariamente pelo autor desta tese, ele fez parte da mesma pesquisa a qual contou com a sua participação. A partir da representação obtida pela técnica proposta, foram realizadas as etapas de pesquisa e de análise dos modelos de reconhecimento e antecipação de gestos para tais ambientes.

Para poder adquirir dados que pudessem comprovar as hipóteses levantadas, milhares de experimentos foram realizados, utilizando várias configurações de diversos modelos. Assim, os resultados dos principais experimentos foram criteriosamente analisados e representados na forma imagens, gráficos e tabelas, de forma a facilitar o entendimento na sua divulgação. Nos experimentos com os modelos de reconhecimento de gestos, foram utilizados quatro conjuntos de dados distintos, sendo três disponibilizadas abertamente na *Internet* e um outro coletado no Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo, Campus de Vitória (Ifes), por meio de um termo de confidencialidade dos dados. Este último foi utilizado apenas para o treinamento. Assim, para coletar o conjunto de teste, os mesmos experimentos realizados no Ifes também foram realizados no Laboratório de Visão Computacional e Robótica da Universidade Federal do Espírito Santo (Ufes). Participando como voluntários, o próprio autor deste trabalho e um aluno de iniciação científica também envolvido nesta mesma pesquisa. Já para a antecipação de ações, foi utilizado um conjunto de dados adquirido e disponibilizado abertamente pelo VisLab-IST.

Todo o tratamento dos dados, visualização de resultados e implementação dos modelos foram realizados utilizando bibliotecas implementadas na linguagem Python. Além disso, os experimentos iniciais foram realizados numa plataforma de colaboração do Google (Google Colab) e os posteriores em computadores e servidores pertencentes aos laboratórios aos quais esta pesquisa pertence, a saber:

- No Laboratório de Visão Computacional e Robótica da Ufes, foram utilizados três servidores contendo uma GPU cada e um quarto servidor contendo quatro GPUs. Além dos servidores, fez-se uso da infraestrutura de câmeras disponível no

laboratório para levantar o conjunto de dados de testes utilizado para avaliar os modelos de reconhecimento e antecipação de gestos em ambientes multicâmeras. Além disso, o laboratório disponibilizou um computador pessoal como estação de trabalho individual para o autor durante todo o período de pesquisa.

- No Vislab-IST, foi utilizado um servidor com duas GPUs, além de um computador pessoal como estação de trabalho individual.

A escrita deste manuscrito e a implementação dos códigos[5] dos experimentos realizados durante o período de pandemia do vírus Sars-Cov-2 se deram exclusivamente no computador portátil do próprio autor.

1.7 Contribuições esperadas

Dentre as principais contribuições esperadas para esta tese, podem-se destacar:

a) Uma técnica de representação de movimento que pode ser utilizada no reconhecimento de ações e/ou gestos.
b) Uma versão iterativa da técnica de representação de movimento proposta, que pode ser utilizada no reconhecimento de ações e/ou gestos utilizando modelos sequenciais.
c) Um modelo não sequencial de reconhecimento de gestos representados na forma de vídeos de tamanhos variados.
d) Um modelo sequencial de reconhecimento de gestos representados na forma de vídeos de tamanhos variados.
e) Um método de fusão de características tão eficaz quanto os tradicionais, mas que facilita a explicação sobre as decisões do modelos.
f) Um modelo capaz de melhorar antecipação de ações utilizando o valor de incerteza e a informação de contexto.
g) A formalização de três métricas para a avaliação de modelos de antecipação.
h) Um modelo de antecipação de gestos representados na forma de vídeos de tamanhos variados.
i) Um modelo de reconhecimento de gestos em tempo real em um ambiente com uma única câmera.
j) Um modelo de reconhecimento e antecipação de gestos em tempo real em um ambiente interacional multicâmeras.

[5] Todos os códigos utilizados para realizar os experimentos apresentados em cada capítulo podem ser acessados nos seguintes repositórios: <https://github.com/clebeson/action_anticipation/> e <https://github.com/clebeson/gesture_recognition/>.

Além deste manuscrito, a maioria das contribuições citadas estão registradas na forma de seis publicações de artigos, todas com revisões arbitradas por pares, sendo quatro em congressos nacionais e duas em um periódico qualificado (Qualis-ENG IV A1). Segue uma lista dos artigos ordenados por ano de publicação:

Ano de 2017:

- SANTOS, C. C.; PICORETI, R.; CARMO, A. P.; VASSALLO, R. F.; GARCIA , A.. Modelagem de um Comportamento Interacional Entre Homen e Robô para um Espaço Inteligente Baseado em Visão Computacional. XIII Simpósio Brasileiro de Automação Inteligente - SBAI 2017, p. 2265-2270.

Ano de 2018:

- SANTOS, C. C.; SAMATELO, J. L. A. ; VASSALLO, R. F. *Improving Dynamic Gesture Recognition by Using CNNs and Color Information Representation*. Workshop de Visão Computacional - WVC, 2018.
- SANTOS, C. C.; SILVA, L. A. ; MENDONCA, F. Q. ; P, Rodolfo ; SAMATELO, J. L. A. ; VASSALLO, R. F. . Uso de Transfer Learning para o Reconhecimento de Gestos Dinâmicos. XXII Congresso Brasileiro de Automática - CBA 2018.

Ano de 2019:

- SANTOS, C. C.; SAMATELO, J. L. A. ; VASSALLO, R. F. Reconhecimento de gestos dinâmicos para a interação humano-robô. XIV Simpósio Brasileiro de Automação Inteligente - SBAI 2019.

Ano de 2020:

- SANTOS, C. C.; SAMATELO, J. L. A.; VASSALLO,R. F. *Dynamic gesture recognition by using CNNs and star RGB: A temporal information condensation*. Neurocomputing, 2020.
- SANTOS, C. C., MORENO, P., SAMATELO, J. L. A., VASSALLO, R. F., SANTOS-VICTOR, J. *Action Anticipation for Collaborative Environments: The Impact of Contextual Information and Uncertainty-Based Prediction*. Neurocomputing . 2020. No prelo.

1.8 Estrutura do trabalho

Para um melhor entendimento, este manuscrito está estruturado da seguinte forma:

Capítulo 1 - Introdução: Responsável por apresentar a motivação necessária à pesquisa e onde foram discutidos os principais problemas encontra-

dos, quais as hipóteses levantadas, soluções propostas e os objetivos da pesquisa.

Capítulo 2 - Referencial teórico: Aqui serão apresentados os principais conceitos e as definições necessárias ao entendimento das propostas apresentadas nesta tese. Esses conhecimentos também facilitarão o entendimento de como os experimentos foram realizados e como os resultados foram alcançados.

Capítulo 3 - Reconhecimento de gestos dinâmicos: Nesse capítulo, serão apresentadas duas propostas de técnicas de representação de movimento a serem utilizadas no reconhecimento de gestos dinâmicos: uma para modelos não sequenciais e outra para modelos sequencias. Além disso, serão mostrados os experimentos realizados em três diferentes conjuntos de dados, os resultados obtidos em cada um deles e os experimentos adicionais realizados a fim de validar a viabilidade das propostas para aplicações em tempo real.

Capítulo 4 - Antecipação de ações: Aqui, será apresentado um modelo estocástico capaz de antecipar uma ação fazendo uso da incerteza de predição estimada. Neste capítulo, também será abordada a melhora alcançada após a utilização da informação de contexto em adição à informação de movimento na antecipação de ações em um conjunto de dados de ações colaborativas. Assim, a fim de avaliar a sua efetividade na resolução de um problema mais complexo, será proposto um modelo para a antecipação dos gestos presentes em um dos conjuntos de dados utilizados na tarefa de reconhecimento do Capítulo 3.

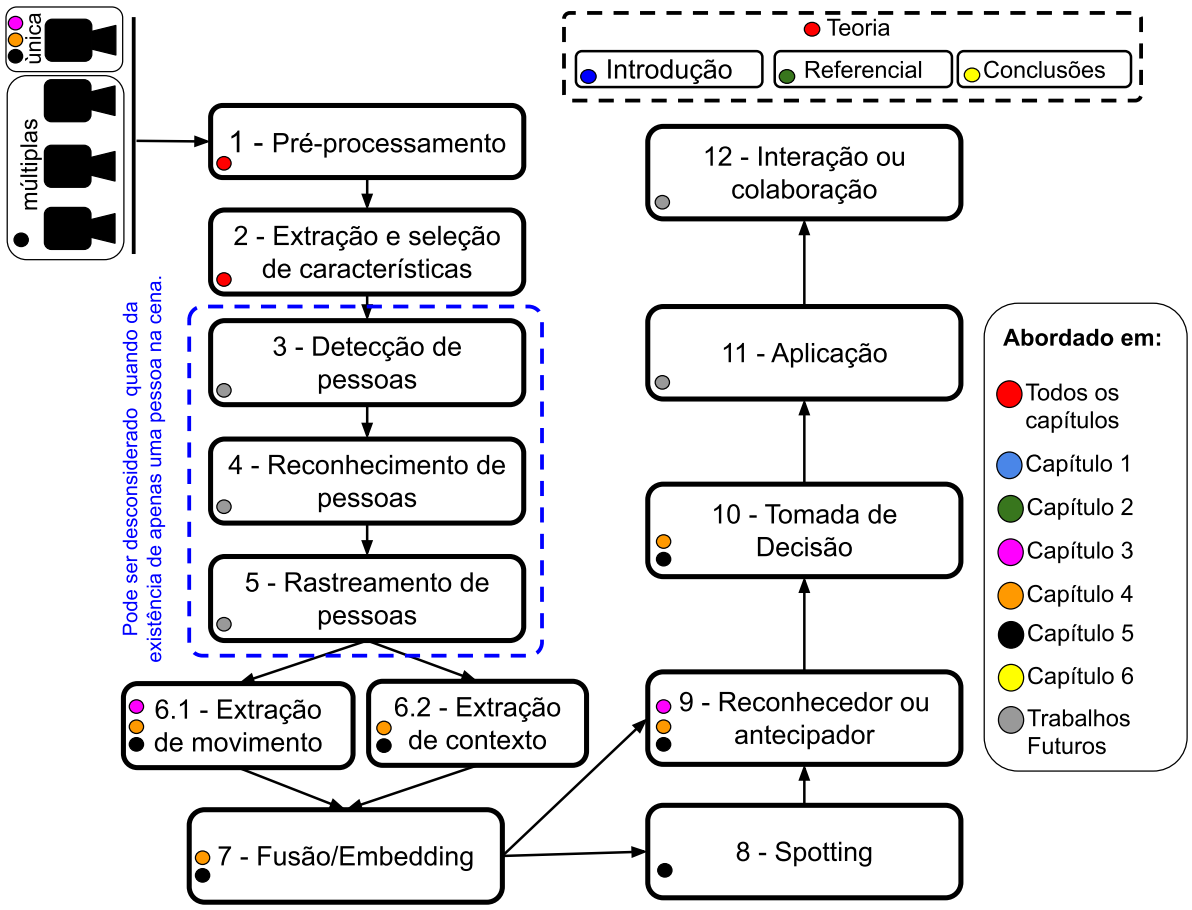
Capítulo 5 - Reconhecimento e antecipação *online* de gestos: Nesse capítulo, será proposto um reconhecedor de gestos em tempo real baseado nos modelos criados anteriormente. Para isso, será proposto um modelo responsável pela realização da operação de *spotting* do gesto e, logo após, um outro modelo que realizará o reconhecimento ou antecipação dos mesmos. Dessa forma, como aplicação, será apresentada uma proposta de um reconhecedor e antecipador de gestos a ser utilizado em um ambiente inteligente multicâmeras. Os experimentos e resultados dos modelos também serão apresentados ao final do capítulo.

Capítulo 6 - Conclusões e trabalhos futuros: Nesse último capítulo serão apresentadas as conclusões sobre as propostas, quais os resultados

alcançados nos experimentos e quais deles corroboram as hipóteses levantadas. Além disso, neste capítulo, serão discutidas as limitações de cada proposta e quais os possíveis trabalhos a serem realizados a fim de darem continuidade a esta pesquisa. Nele também serão abordados os problemas que ainda carecem de soluções e que não foram foco de estudo deste trabalho.

Na Figura 1, pode ser visto um diagrama completo de uma possível solução que objetiva a interação ou colaboração entre agentes baseando-se apenas em visão computacional. Nela, também pode ser visto em que parte do manuscrito cada bloco do diagrama será abordado.

Figura 1 – Diagrama completo de uma possível solução para interação ou colaboração baseando-se apenas em visão computacional. Em cada bloco, um ou mais círculos apresentam em qual capítulo o mesmo será a abordado. A fim de cobrir todo o manuscrito, foram inseridos mais três blocos correspondentes às partes teóricas que nortearam a pesquisa.



Fonte: Próprio autor.

2 REFERENCIAL TEÓRICO

> *Essentially, all expressions of human nature ever produced, from a caveman's paintings to Mozart's symphonies and Einstein's view of the universe, emerge from the same source: the relentless dynamic toil of large populations of interconnected neurons.*
>
> Miguel Nicolelis

Para uma melhor compreensão do que será abordado nesta tese, faz-se necessário a apresentação de alguns conceitos teóricos. Dessa forma, este capítulo resumirá a teoria por trás das abordagens utilizadas nas soluções propostas.

2.1 Aprendizado de máquina

O aprendizado de máquina (ML - do inglês *Machine Learning*) é uma parte da estatística aplicada que usa algoritmos computacionais capazes de aprender por meio de dados (GOODFELLOW; BENGIO; COURVILLE, 2016). Uma definição mais formal sobre algoritmos de aprendizagem é dada por Mitchell et al. (1997): *"Um programa de computador aprende a partir da experiência E com respeito a uma tarefa T e uma medida de desempenho P, se o seu desempenho em uma tarefa T, medida por P, melhora a experiência E"* (tradução livre).

No aprendizado de máquina, a tarefa é aquilo que se deseja resolver, a experiência é o artifício utilizado na resolução da tarefa e a medida de desempenho é a responsável por guiar o mecanismo de aprendizagem em seu processo de aquisição de experiência. Por exemplo, considere-se o problema de reconhecer gestos em vídeos. Para fazê-lo de maneira automática, poder-se-ia escrever um código que representasse todas as possibilidades de realização de um gesto. No entanto, nem os próprios seres humanos sabem especificar com detalhes como eles reconhecem os gestos uns dos outros. Dessa maneira, uma solução mais factível é criar um algoritmo capaz de extrair características específicas de um vídeo, e determinar a qual gesto elas pertencem. No início, o algoritmo funciona de maneira aleatória, pois ainda não adquiriu experiência suficiente (aprendeu) para dizer com certeza qual o gesto que está sendo executado no vídeo. Assim, é necessário indicar quando ele está certo ou errado, ou quão próximo ele está da solução correta. Com isso, após várias iterações recebendo diferentes vídeos com diferentes gestos, o algoritmo será capaz de reconhecer um gesto de maneira mais segura. Esse processo se repete até que se alcance

um limiar de satisfação para a qualidade da experiência adquirida. O algoritmo resultante desse processo também pode ser chamado de modelo computacional, ou estimador teórico.

Um modelo pode ser classificado como paramétrico, quando uma quantidade fixa de parâmetros é responsável por adquirir toda a experiência necessária na resolução do problema, independente da quantidade de dados utilizada durante o processo de aprendizagem; ou não paramétrico, quando o número de parâmetros que representa a experiência varia com a quantidade de dados.

A capacidade de aprendizagem de um modelo paramétrico é dado comumente pela quantidade de parâmetros que o mesmo possui. Assim, um modelo com um grande número de parâmetros é dito ter mais capacidade que outro modelo com menos parâmetros. A capacidade de aprendizagem também está relacionada com a capacidade de generalização do modelo, ou seja, a de resolver tarefas pertencentes ao mesmo domínio, mas que são representadas por dados com distribuições nunca antes observadas. O mais importante de um modelo é a sua capacidade de generalização. Assim, a fim de adquirir métricas que indiquem o grau de tal generalização, os dados utilizados podem ser divididos em três principais conjuntos: o de treino, o de validação e o de teste. Dessa forma, enquanto os dados dos conjuntos de treino e validação são observados durante o processo de treinamento, os do conjunto teste são utilizados apenas para avaliar a qualidade do modelo após o seu treinamento.

Um modelo com baixa capacidade de aprendizagem, ao tentar resolver um problema complexo (alta dimensionalidade), normalmente não alcança uma boa medida de desempenho durante o treinamento. Assim, diz-se que o mesmo está subajustado aos dados de treino (*underfitting*). Já quando um modelo com alta capacidade de aprendizagem obtém uma medida de desempenho nos dados de treino muito maior que nos dados de testes ou validação, diz-se que o mesmo não foi capaz de generalizar o problema (baixa capacidade de generalização), ou que o mesmo está sobreajustado aos dados de treino (*overfitting*). No aprendizado estatístico, os problemas de *under* e *overfitting* também são conhecidos como o *trade-off* viés-variância e podem ser tratados das seguintes maneiras: para modelos com *underfitting*, deve-se aumentar a sua capacidade de aprendizagem e (ou) diminuir a dimensão dos dados de entrada; já para aqueles com *overfitting*, deve-se diminuir a sua capacidade de aprendizagem e (ou) penalizá-los por meio de uma ou mais técnicas de regularização, por exemplo, normalização L1 e L2, *dropout*, *data augmentation*, dentre outras. Uma visão mais ampla sobre tais técnicas é dada em Goodfellow, Bengio e Courville (2016), e uma leitura mais detalhada sobre o *trade-off* viés-variância pode ser feita em Bishop (2006), Murphy (2012) ou James et al. (2013).

Em resumo, o processo de aprendizagem de máquinas emprega quatro principais componentes: um modelo, os dados, um algoritmo de aprendizagem e uma função de custo. O modelo é o componente responsável por resolver o problema; os dados representam o

domínio ao qual o modelo será aplicado; o algoritmo de aprendizagem será o encarregado de extrair dos dados a experiência responsável por tornar o modelo capaz de resolver o problema e a função de custo guiará o algoritmo de aprendizagem em sua tarefa.

A experiência do modelo é extraída dos dados do conjunto de treino. No entanto, como a função de custo também é avaliada sobre tal conjunto de dados, não é possível afirmar que o modelo está correto em suas predições. Ele pode estar sendo tendencioso a encontrar uma maneira fácil de obter um boa avaliação na métrica de desempenho, sem se preocupar com dados ainda não observados. Gerando assim a necessidade de utilização dos conjuntos de validação e de teste.

Como o treinamento é comumente um processo iterativo, é necessário saber quando pará-lo. Assim, pelos resultados da métrica de desempenho aplicada sobre o conjunto de validação, o qual o modelo não usa para adquirir experiência, é possível avaliar a sua capacidade de generalizar a resolução do problema e poder, assim, ou parar o treinamento, ou iniciá-lo novamente. No entanto, após várias tentativas para obter um bom desempenho sobre conjunto de validação, pode-se ter enviesado o modelo a resolver o problema apenas para tal conjunto dados. Assim, a fim de obter uma métrica menos tendenciosa sobre o desempenho do modelo, utiliza-se o conjunto de teste, o qual o modelo nunca havia observado. O resultado obtido nesse último conjunto de dados pode ser então utilizado nas decisões que envolvam o uso do modelo treinado.

Em casos onde o número de amostras de todo o conjunto de dados é muito pequeno, é comum dividi-lo apenas em treino e validação e realizar um processo chamado de validação cruzada. Nesse processo, o modelo é treinado e validado k vezes utilizando configurações diferentes dos dados de treino e validação. Assim, o resultado, a ser considerado como o desempenho do modelo na resolução do problema, é a média e a variância dos resultados da métrica de avaliação obtidos nos k treinamentos realizados. Após saber quais os hiperparâmetros responsáveis pelo modelo com melhores resultados na validação cruzada, os mesmos são utilizados para treinar um modelo incluindo todo o conjunto de dados, sem divisões. Na validação cruzada, quanto maior o valor de k, menos enviesado é o modelo, maior é a sua variância e mais demorado é o processo de avaliação. Sendo assim, costuma-se escolher um valor que equilibre a capacidade de generalização do modelo (*trade-off* viés-variância) e o tempo (recursos computacionais) gasto (demandados) em seu treinamento. Essa técnica é muito utilizada para a escolha de modelos e hiperparâmetros. Assim, para um estudo mais aprofundado sobre o seu funcionamento e características, pode-se consultar James et al. (2013).

Um modelo é escolhido baseando-se na natureza e dimensionalidade dos dados que representam o problema a ser resolvido. Sendo assim, no aprendizado de máquina, a primeira etapa é determinar o tipo de dados a ser utilizado na solução do problema e como o mesmo será tratado a fim de facilitar a aprendizagem. A seguir, define-se o

modelo, o algoritmo de aprendizagem e a métrica de desempenho a ser utilizada. Com isso em mãos, inicia-se o processo de treinamento. O modelo recebe como entrada os dados de treinamento, transforma-os para uma representação que facilite a compreensão do problema e oferece uma solução para o mesmo. Essa solução é avaliada pela função de custo, que fornece os resultados para o algoritmo de aprendizagem. Em seguida, ele usa essas informações para modificar o modelo de maneira que na próxima iteração o resultado da avaliação seja melhorado.

2.2 Redes neurais clássicas

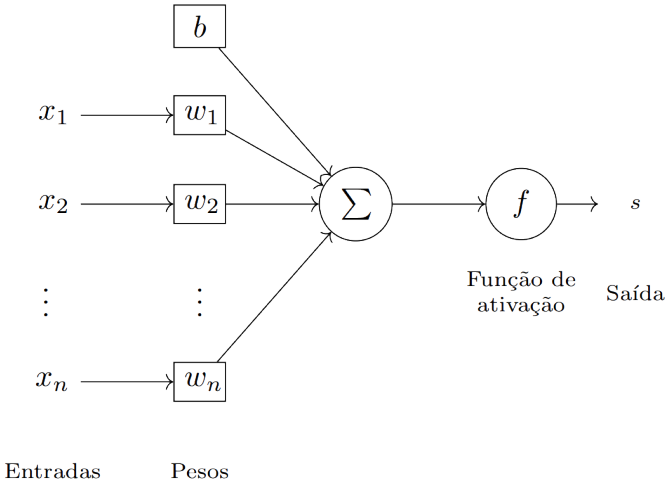
Dentre as várias arquiteturas de modelos existentes para o aprendizado de máquinas, as redes neurais artificiais chamam a atenção por serem um tipo de modelo paramétrico capaz de resolver problemas complexos. Além disso, chamam a atenção ainda pela característica intrínseca de poderem ser processadas de forma paralela e de permitirem o aumento da capacidade de aprendizagem sem muitos prejuízos ao tempo gasto durante a predição.

Uma rede neural artificial é um modelo computacional inspirado no cérebro humano, onde camadas de neurônios são responsáveis por receber e enviar dados uns para os outros (HAYKIN, 2007). Cada neurônio aplica uma transformação aos seus dados de entrada de maneira a oferecer uma melhor representação para os neurônios seguintes. Dessa forma, ao fim do processo, é possível reconhecer padrões, inferir valores ou tomar decisões a partir dos dados de entrada. A Figura 2 apresenta um modelo de neurônio artificial chamado de *perceptron* simples. Note-se que ele recebe um conjunto de valores como entrada, realiza uma soma ponderada e transforma o resultado por meio de uma função de ativação não linear. Em notação matricial, o resultado de um perceptron pode ser obtido por meio da Equação 2.1, onde f é a função de ativação, $\mathbf{w} = [w_1,w_2,...,w_n]^T$ é um vetor de pesos, $\mathbf{x} = [x_1,x_2,x_3,...,x_n]^T$ é um vetor de entradas e b é um peso que representa o *bias* (viés).

$$s = f(\mathbf{w}^T\mathbf{x} + b) \tag{2.1}$$

Uma rede neural normalmente possui múltiplos neurônios em uma estrutura com múltiplas camadas (MLP - do inglês *multilayer perceptron*), como a da Figura 3. Dessa forma, a saída de uma camada pode ser descrita como $\mathbf{s} = f(\mathbf{W}^T\mathbf{x} + \mathbf{b})$, ou seja, como o resultado de uma função não linear f aplicada ponto a ponto sobre o vetor resultante de uma transformação afim de um vetor de entrada $\mathbf{x}$, fornecido pela camada anterior, utilizando uma matriz de pesos $\mathbf{W}$ e um vetor de *bias* $\mathbf{b}$. Sendo assim, o processo de treinamento de uma rede neural acontece mediante o ajuste dos valores de cada elemento de $\mathbf{W}$ e $\mathbf{b}$. O que faz com que toda a experiência do modelo possa ser armazenada ou

Figura 2 – Perceptron simples com N entradas (x_1,x_2,...,x_n), um *bias* b, $N+1$ pesos, um operador de soma, uma função de ativação não linear e uma saída s.

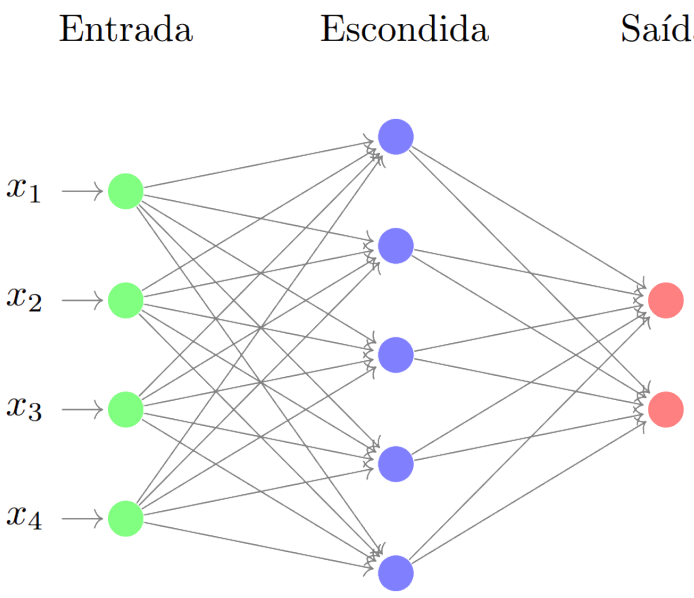


Fonte: Próprio autor.

transferida via seus pesos.

Inspirado no teorema apresentado por Kolmogorov (1957), o teorema da aproximação universal (HORNIK et al., 1989) afirma que uma rede neural multicamadas com apenas um camada escondida com um número infinito de neurônios pode aproximar qualquer função contínua para qualquer grau de precisão desejado. Por esse motivo, as redes neurais são utilizadas para tentar resolver problemas complexos, onde os dados de entrada e as suas saídas correspondentes não possuem uma relação linear.

Figura 3 – Rede neural do tipo MLP com 4 neurônios de entrada, 1 camada escondida com 5 neurônios e 1 camada de saída com 2 neurônios



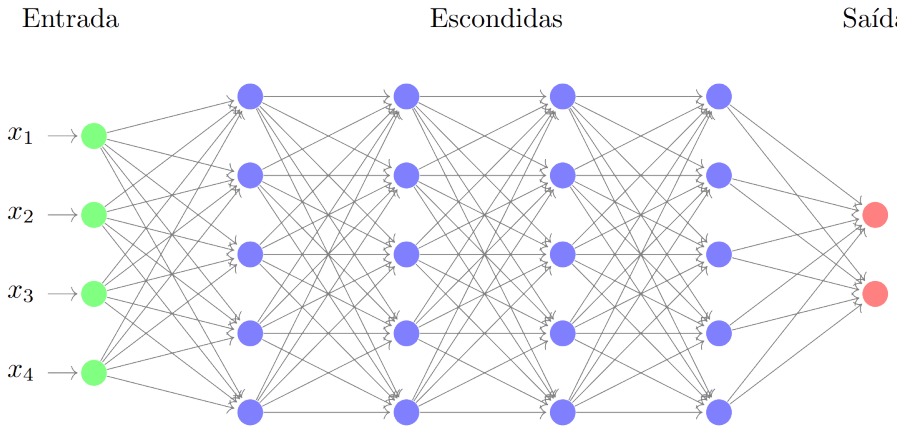
Fonte: Próprio autor.

2.3 Redes Neurais Profundas

A função de uma rede neural é transformar os dados de entrada de maneira a obter um representação capaz de ser facilmente separável por uma função linear. Dessa forma, cada camada aplica uma transformação sobre a saída da camada anterior, de sorte que mais camadas podem melhorar a representação obtida. No entanto, quanto mais camadas a rede tem, mais a sua capacidade de aprendizagem aumenta e as técnicas de treinamento precisam ser ainda mais robustas.

Quando uma rede neural possui mais de uma camada escondida, ela é chamada de rede neural profunda sem realimentação (DFNNs - do inglês *Deep Feedforward Neural Networks*) (HINTON; OSINDERO; TEH, 2006), e seu processo de aprendizagem é então chamado de aprendizado profundo (DL - do inglês *Deep Learning*). Em DL, tanto as MLPs, quanto as DFNNs são chamadas de redes totalmente conectadas (FCs - do inglês *Fully Connected*), devido à sua natureza multiplicativa matriz-vetor, que faz com que todos os neurônios de uma camada tenham conexão com todas as saídas da camada anterior. A Figura 4 apresenta um exemplo de uma DFNN[1].

Figura 4 – Representação de uma rede neural profunda sem realimentação.



Fonte: Próprio autor.

2.4 Redes Neurais Convolucionais

Modelos profundos, como aqueles apresentados na Figura 4, costumam alcançar bons resultados em problemas cujas características podem ser permutadas (de natureza não estruturada) e representadas na forma de vetores unidimensionais (GOODFELLOW; BENGIO; COURVILLE, 2016). Uma peculiaridade desse tipo de modelo é a relação direta do seu número de parâmetros com a dimensão do vetor de entrada. Quanto maior a

[1] É importante destacar que uma DFNN é um tipo de MLP, no entanto, com muitas camadas escondidas.

dimensionalidade da entrada, maior será a sua capacidade de aprendizagem, o que exigirá maior poder processamento para o seu treinamento, um conjunto de dados com um número maior de amostras, além de outros mecanismos de regularização.

Em problemas de visão computacional, onde as imagens possuem milhares, ou até mesmo milhões de píxeis, o uso de DFNNs pode ser inviável, a não ser que se utilize etapas de pré-processamento sobre os dados de entradas, também conhecidas como extração e seleção de características. No entanto, cada problema necessita de uma etapa de pré-processamento específica, o que pode retardar as resoluções de problemas de visão computacional, além de torná-las dependentes de uma tarefa humana altamente especializada conhecida como engenharia de características (*feature engineering*). Detalhes sobre várias técnicas de *feature engineering* podem ser encontrados em Nixon e Aguado (2019), Chandrashekar e Sahin (2014) ou Guyon et al. (2008).

As imagens, em sua maioria, possuem informações estacionárias e organizadas de forma espacial, fazendo com que píxeis vizinhos estejam fortemente relacionados a uma determinada característica. Dessa maneira, baseando-se em estudos sobre o funcionamento do córtex visual dos mamíferos, publicados entre o final da década de 1950 e meados da década de 1970, foram criadas as redes neurais convolucionais (CNNs - do inglês *Convolutional Neural Networks*). A CNN é um tipo de rede neural artificial que possui uma estrutura organizada em camadas, sendo que as primeiras (mais rasas) possuem um campo receptivo menor e são responsáveis por capturar informações relacionadas à geometria dos objetos (bordas e angulações); já as últimas (mais profundas) possuem um campo receptivo maior e capturam informações mais elaboradas dos objetos (informações semânticas), como características de texturas e relações entre as formas primitivas capturadas pelas camadas mais rasas.

Apesar das principais descobertas sobre o funcionamento do córtex visual dos mamíferos terem sido entre os anos de 1959 e 1972, a primeira tentativa de reprodução artificial de tal modelo perceptivo foi apresentada em Fukushima e Miyake (1982), um modelo de reconhecimento de dígitos manuscritos chamado de *neocognitron*. Devido à impossibilidade de ser treinado de maneira supervisionada, o *neocognitron* não conseguiu ser absolvido pela comunidade científica e industrial. Porém, deu fortes inspirações para que LeCun et al. (1989b) propusessem uma abordagem similar que poderia ser treinada de maneira automática e supervisionada, muito similar às CNNs, no entanto ainda não possuía tal denominação. Quando dessa proposta, o maior limitante era o poder processamento demandado para o seu treinamento. Por isso, ela não despertou tanto brilho nos olhos dos principais pesquisadores da área.

Apenas em LeCun, Bengio et al. (1995a) que as CNNs foram assim denominadas. Agora, com um maior desenvolvimento dos dispositivos de processamento e algumas mudanças no seu modo de funcionamento, elas conseguiram ser aplicadas efetivamente na

resolução do problema de reconhecimento de dígitos manuscritos, tornando-as assim um dos mecanismos mais eficazes no reconhecimento de assinaturas em cheques e documentos utilizados nos anos 2000. Porém, ainda demandavam um elevado poder de processamento, o que inviabilizava a sua utilização na resolução de problemas mais complexos, como a classificação de imagens naturais, o reconhecimento de objetos e processamento de linguagem natural.

Apenas a partir de 2012, após a possibilidade de utilizar as unidades de processamento gráfico (GPUs - do inglês *Graphical Process Units*) no treinamento paralelo das CNNs, que surgiram arquiteturas de redes convolucionais como a Alexnet (KRIZHEVSKY; SUTSKEVER; HINTON, 2012b), VGG16 (SIMONYAN; ZISSERMAN, 2014c), GoogleNet (SZEGEDY et al., 2015) e Resnet (HE et al., 2016b). Tais arquiteturas alcançaram resultados significativos na classificação de imagens naturais em um dos mais difíceis desafios da área, o ILSVRC (RUSSAKOVSKY et al., 2015). Isso acabou por chamar a atenção da comunidade científica. Desde então, protagonizam as principais soluções de visão computacional e processamento de linguagem natural desenvolvidas na academia e utilizadas na indústria.

As redes convolucionais são assim chamadas pois aplicam operações de convolução, utilizando máscaras (*kernels*) com valores treináveis, sobre as entradas fornecidas pelas camadas anteriores (*feature maps*). Dessa forma, é possível extrair, selecionar e combinar automaticamente as características mais úteis para a resolução do problema, sem a necessidade de *feature engineering*. Uma propriedade interessante das CNNs, é que a quantidade e o tamanho dos seus filtros não variam de acordo com o tamanho da imagem de entrada. Assim, é possível utilizar a mesma CNN para imagens de tamanhos variados, o que não é possível fazer com as DFNNs sem que se modifique a sua estrutura.

Devido à natureza da aplicação das máscaras de convolução sobre as imagens ou sobre os *features maps*, as CNNs são equivariantes a translações e invariantes a pequenas rotações. Tais propriedades também são encontradas no córtex visual dos mamíferos, onde campos receptivos próximos ativam neurônios adjacentes. Por esses e outros motivos, as CNNs são uma das arquiteturas de redes neurais profundas mais utilizadas no momento para solucionar problemas de visão computacional em imagens e vídeos, ou mesmo em dados de voz. Indo um pouco mais além, as CNNs podem ser consideradas o modelo de aprendizagem de máquina bioinspirado mais eficaz da atualidade (GOODFELLOW; BENGIO; COURVILLE, 2016), mesmo que ainda careça de muitos estudos para explicar detalhadamente o seu funcionamento.

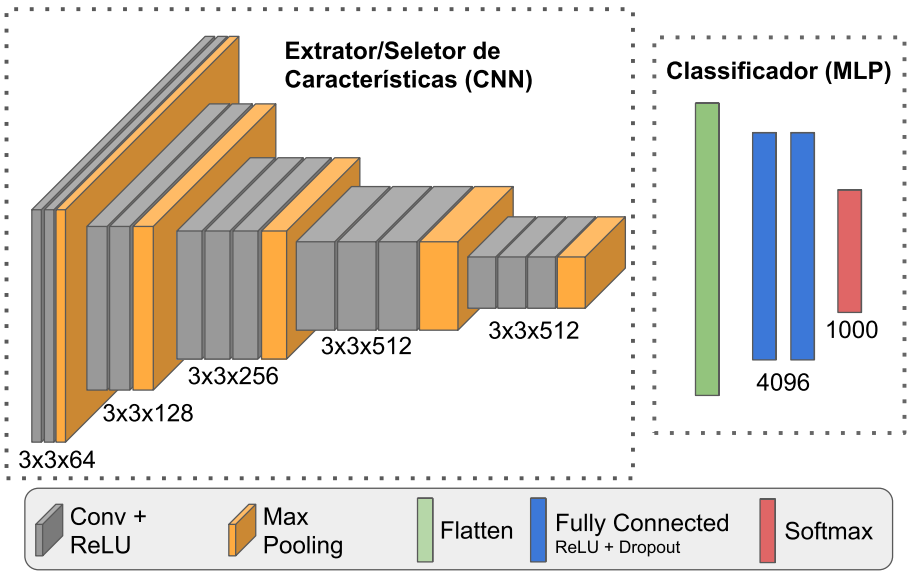
Em suma, uma CNN é similar a uma rede neural multicamadas, contendo camadas de neurônios que realizam operações de convolução seguidas de transformações não lineares[2] sobre os resultados de camadas anteriores. Os parâmetros de tais operações são

[2] Uma das funções não lineares mais utilizadas nas CNNs é função de ativação ReLU (NAIR; HINTON,

conhecidos como pesos ou *kernels* (núcleos) e também são ajustados por meio de uma etapa de treinamento via um algoritmo de otimização, conhecido como *Back Propagation* (RUMELHART; HINTON; WILLIAMS, 1985). Na Figura 5, é possível ver a representação gráfica de uma CNN. A rede representada é uma VGG16 (SIMONYAN; ZISSERMAN, 2014c), muito utilizada como rede base para a comparação de resultados em diversos problemas de visão computacional.

Apesar de atualmente existirem as chamadas redes completamente convolucionais (FCNs - do inglês *Fully Convolutional Networks*), ainda é comum utilizar redes com arquiteturas híbridas, como a VGG16. Nelas, uma CNN realiza a tarefa de extração e seleção de características, enquanto que uma MLP é utilizada na tarefa de classificação. Com isso, aproveita-se tanto o potencial das CNNs de extraírem e selecionarem as características das imagens (vídeos), que são mais relevantes para a solução do problema, quanto o das MLPs de as classificarem. Além do mais, por poderem ser treinadas via *back propagation*, o treinamento do extrator e do classificador é realizado simultaneamente (treinamento fim a fim). Sendo assim, neste trabalho, todas as arquiteturas de CNNs utilizadas serão híbridas.

Figura 5 – Representação de uma rede neural convolucional conhecida como VGG16.



Fonte: Próprio autor.

Um leitura mais densa sobre as redes neurais convolucionais e o aprendizado profundo pode ser visto em Goodfellow, Bengio e Courville (2016), LeCun, Bengio e Hinton (2015). Para acompanhar as intuições que levaram à proposta das CNNs, pode-se ver Fukushima e Miyake (1982), LeCun et al. (1989b), LeCun et al. (1989a), LeCun, Bengio et al. (1995b). Já para para as teorias sobre o funcionamento do córtex visual dos

2010a).

mamíferos, têm-se Hubel e Wiesel (1959), Hubel e Wiesel (1962), Hubel e Wiesel (1965), Hubel e Wiesel (1968).

2.5 *Dropout*

Como dito antes, uma vez que as redes neurais profundas possuem uma maior capacidade de aprendizagem, elas são mais propícias a *overfitting*. Dessa forma, a fim de obter melhores resultados na generalização da aprendizagem, devem-se utilizar mecanismos de regularização.

Além dos clássicos regularizadores L1 e L2, conhecidos na literatura de ML como mecanismos de decaimento de pesos (MURPHY, 2012), o *dropout*, proposto em Srivastava et al. (2014), é um regularizador capaz de simular a técnica de *ensemble*[3], *bagging*, sem ter que treinar vários modelos de maneira separada. Mais detalhes sobre a técnica de *bagging* e outras técnicas de *ensemble* podem ser vistas em Haykin (2007).

O *dropout* funciona da seguinte maneira: o vetor $S = (s_1,s_2,s_3,...,s_N)$ contendo as N saídas de uma camada da rede neural é multiplicado ponto a ponto por um vetor $Z = (z_1, z_2, z_3, ..., z_N)$, que representa uma máscara binária, onde cada elemento é amostrado de uma distribuição de *Bernoulli* com probabilidade $1-p$. Aqui, p representa a probabilidade de um neurônio continuar ativo no cálculo da próxima camada. Assim, a máscara desativa alguns neurônios de cada camada, o que faz com que a rede seja penalizada e forçada a aprender diferentes características para o mesmo dado de entrada. Fazendo-a depender menos das ativações de neurônios específicos, aumentando assim a sua capacidade de generalização. Esse processo é descrito na Equação (2.2), na qual $\odot$ representa o produto *Hadamard* (multiplicação ponto a ponto).

$$\begin{aligned} dropout(S,p) &= S \odot Z \\ &= (s_1 z_1, s_2 z_2, s_3, z_3, ..., s_N z_N) \\ &z_i \sim Bernoulli(1-p), \forall i \in (1,2,3,...,N) \end{aligned} \tag{2.2}$$

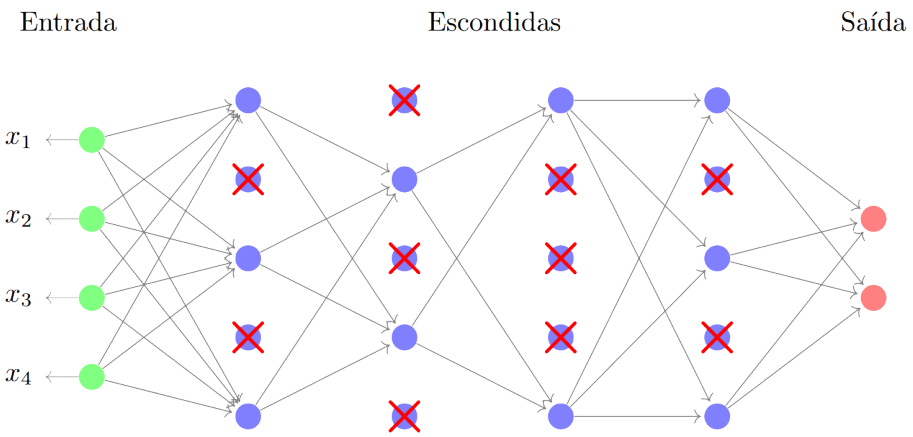
Uma representação da aplicação de *dropout* sobre a rede da Figura 4 com uma probabilidade $p = 0{,}5$ é apresentada na Figura 6.

2.6 *Transfer Learning*

Segundo Kumar (2017), a inicialização dos pesos de uma rede convolucional influencia diretamente na sua capacidade de aprendizagem (convergência). Sendo assim,

[3] o termo ensemble foi traduzido em Haykin (2007) como comitê de máquinas.

Figura 6 – Representação da função *dropout* em uma rede profunda.



Fonte: Próprio autor.

escolher os pesos iniciais é uma etapa importante para obtenção de resultados satisfatórios no problema a ser resolvido. No entanto, essa tarefa ainda é muito subjetiva, e depende do conhecimento e experiência do criador do modelo. Não obstante, para problemas que compartilham similaridades, os pesos iniciais podem ser aqueles obtidos por meio de um treinamento prévio em outro conjunto de dados, o que também é conhecido como transferência de aprendizado (TL - do inglês *Transfer Learning*) (PAN; YANG, 2010). Desse modo, ao invés de inicializar os pesos de um modelo com valores aleatórios, utilizam-se os valores dos pesos de um modelo treinado em outro conjunto de dados. Dessa maneira, a inicialização tem maior efetividade, já que, o novo modelo pode convergir em um menor número de iterações. Ou seja, o *transfer learning* é caracterizado pela mudança de domínio entre modelos.

2.6.1 Principais características

É importante destacar que essa técnica vai além da inicialização dos pesos, permitindo também o reaproveitamento da arquitetura da rede original, o que possibilita mais rapidez na implementação de modelos de aprendizagem que utilizam arquiteturas semelhantes.

Existem três principais casos para a aplicação do TL, a saber:

1. Quando o novo conjunto de dados é grande e semelhante (natureza e características) ao utilizado na rede original já treinada. Aqui, podem-se utilizar os pesos da rede original, ajustando-os minimamente ao problema definido pelo novo conjunto de dados, por meio de um curto treinamento. A essa técnica dá-se o nome de *fine tunning* (ajuste fino).

2. Quando o novo conjunto de dados é pequeno, mas semelhante ao utilizado na rede original já treinada. Neste caso, a primeira parte da rede é inicializada com os pesos da rede original, e os pesos da segunda parte são inicializados aleatoriamente. Assim, durante o treinamento, treina-se apenas a segunda parte, utilizando a primeira como extrator de características. Em alguns casos, após o modelo chegar próximo da sua região de convergência, pode-se aplicar o *fine tunning* em toda a rede.

3. Quando o novo conjunto de dados é diferente do utilizado na rede original previamente treinada. Nesse caso, independente do seu tamanho, deve-se utilizar parte da rede previamente treinada e parte iniciada aleatoriamente. O treinamento deve ser feito utilizando o *fine tunning*.

Como dito antes, uma característica interessante das CNNs aplicadas a imagens é que as primeiras camadas são responsáveis por descrever bordas e texturas, as intermediárias descrevem formas e as finais descrevem padrões de mais alto nível, também conhecidas como camadas semânticas (ZEILER; FERGUS, 2014). Sendo assim, quando o conjunto de dados difere do utilizado na etapa de treinamento da rede (ou seja, existe a necessidade de *fine tunning*), as camadas iniciais e intermediárias costumam trazer melhores resultados para o reconhecimento, pois selecionam características de maior generalidade.

Percebe-se que um outro ponto interessante do *transfer learning* é a possibilidade de reutilização apenas da parte convolucional da rede e optar por uma rede totalmente conectada, com um número diferente de neurônios, ou mesmo utilizar um outro tipo de classificador. Isso faz com que a rede reutilizada sirva como um extrator de características treinado a partir dos dados, o que pode eliminar a custosa etapa de *feature engineering*.

2.6.2 Definição matemática

A fim de melhorar o entendimento sobre o TL, uma definição matemática do método de aprendizado é dada a seguir.

Em geral, no contexto de aprendizagem de máquinas, um domínio $\mathcal{A}$ consiste de dois componentes: um espaço de características $\mathcal{X}$; e uma distribuição de probabilidade marginal $P(\mathbf{x})$, onde $\mathbf{x} = \{x_1,...,x_n\} \in \mathcal{X}$. Agora, considerando um domínio específico $\mathcal{B}$, uma tarefa $\mathcal{T}$ será definida por duas componentes: um espaço de rótulos $\mathcal{Y}$; e uma função preditiva objetivo $f(\bullet)$, que não é observada, mas pode ser aprendida pelos dados do conjunto de treinamento; sendo que o conjunto de treinamento consiste em um conjunto de pares $\{\mathbf{x}_k, \mathbf{y}_k\}$, onde $\mathbf{x}_k \in \mathcal{X}$ e $\mathbf{y}_k \in \mathcal{Y}$. A função $f(\bullet)$ pode ser usada para predizer o rótulo correspondente a $f(\mathbf{x})$ de uma nova instância $\mathbf{x} \in \mathcal{X}$. De um ponto de vista probabilístico, a função $f(\mathbf{x})$ pode ser interpretada como $P(\mathbf{y}|\mathbf{x})$.

Para um melhor entendimento da notação definida acima, suponha-se o problema de classificação de imagens. Então, em relação ao domínio $\mathcal{B}$, $\mathcal{X}$ é o espaço de todas as imagens possíveis; $\mathbf{x}_k$ é uma imagem particular; x_i é o i-ésimo vetor ou tensor de características correspondente à i-ésima imagem; n é o número de píxeis ou vetores de características em $\mathbf{x}_k$. Em relação à tarefa $\mathcal{T}$, $\mathcal{Y}$ é o conjunto dos rótulos pelos quais devem ser classificadas as imagens, sendo que $\mathbf{y}_k$ assume o valor de um desses rótulos.

Pelas definições acima, tem-se: um domínio $\mathcal{B} = \{\mathcal{X}, P(\mathbf{x})\}$ e uma tarefa $\mathcal{T} = \{\mathcal{Y}, f(\bullet)\}$. Ao considerar uma tarefa fonte e destino, tem-se que o conjunto de dados do domínio fonte de tamanho S (*Source*) é definido como o conjunto de pares $\{(\mathbf{x}_{S_1}, \mathbf{y}_{S_1}), ..., (\mathbf{x}_{S_n}, \mathbf{y}_{S_n})\}$, onde $\mathbf{x}_{S_k} \in \mathcal{X}_S$ e $\mathbf{y}_{S_k} \in \mathcal{Y}_S$, e que o conjunto de dados do domínio destino de tamanho T (*Target*) é definido como o conjunto de pares $\{(\mathbf{x}_{T_1}, \mathbf{y}_{T_1}), ..., (\mathbf{x}_{T_n}, \mathbf{y}_{T_n})\}$, onde $\mathbf{x}_{T_k} \in \mathcal{X}_T$ e $\mathbf{y}_{T_k} \in \mathcal{Y}_T$.

Como é indicado em Pan e Yang (2010), na maioria dos casos, $0 \leq n_T \ll n_S$. Assim, seja: um domínio fonte $\mathcal{B}_S$ e a tarefa a aprender $\mathcal{T}_S$; um domínio destino $\mathcal{B}_T$ e tarefa a aprender $\mathcal{T}_T$. Então, o objetivo de TL é ajudar no aprendizado da função preditiva $f_T(\bullet)$ em $\mathcal{B}_T$ usando o conhecimento em $\mathcal{B}_S$ e $\mathcal{T}_S$, sendo $\mathcal{B}_S \neq \mathcal{B}_T$ e (ou) $\mathcal{T}_S \neq \mathcal{T}_T$.

2.7 Problemas de natureza sequencial

Em fontes de dados como as imagens, as características estão distribuídas de forma espacial e representadas como sendo os valores de intensidade de cada píxel. Porém, em algumas fontes de informação, como áudio, texto e vídeo, por exemplo, suas características podem ser também temporais, ou seja, elas podem evoluir ao passar do tempo, de maneira a gerar dependências entre diferentes observações. Assim, características de instantes de tempos distintos (observações distintas) podem ser altamente dependentes. Assim, são caracterizados os problemas de natureza sequencial estudados neste trabalho.

2.7.1 Principais abordagens

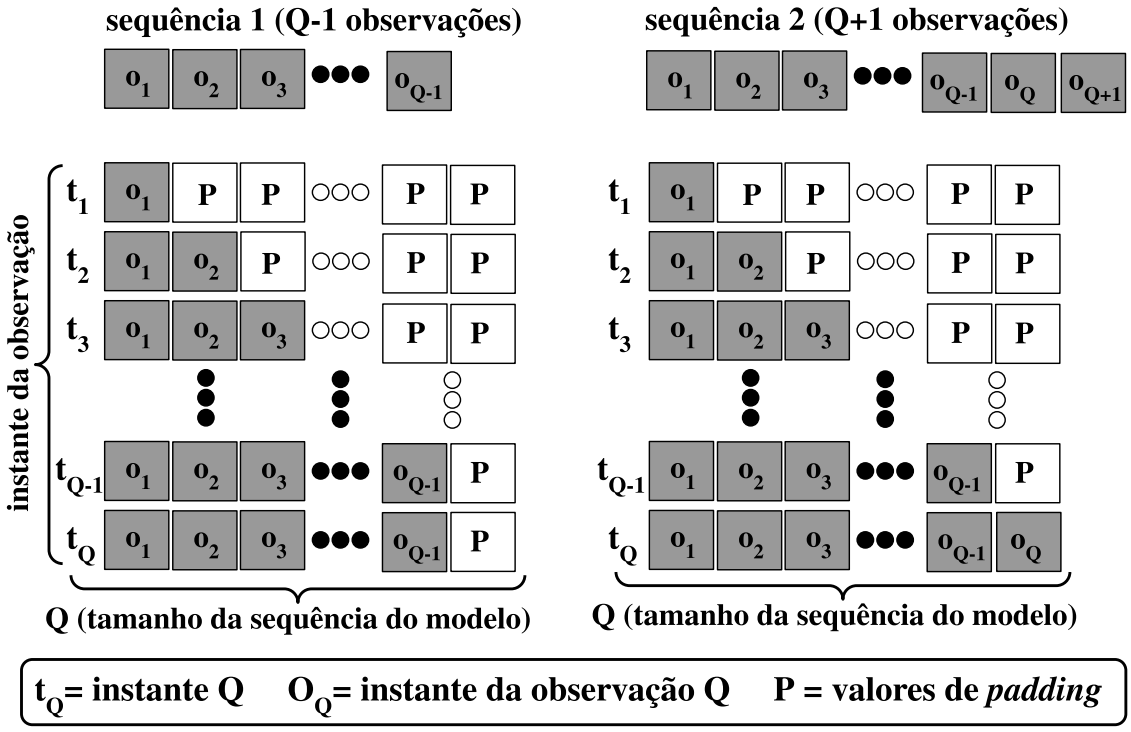
Assumindo-se a existência de dependência entre observações de diferentes instantes de tempo, um problema de natureza sequencial pode ser tratado de duas maneiras principais: considerando que todas as sequências são limitadas a um tamanho fixo, Q, de observações, ou assumindo o tamanho original (variável) das sequências.

A primeira abordagem desconsidera a variação no tamanho de todas as sequências e introduz um novo hiperparâmetro Q que influencia diretamente no desempenho do modelo. Nesse sentido, uma sequência de tamanho $T > Q$ pode ser amostrada para conter apenas Q observações, ou deve ser truncada na observação Q, descartando-se $T - Q$

observações; enquanto uma sequência de tamanho $T < Q$ deve ser preenchida com $Q - T$ valores. Uma ilustração desse processo pode ser vista na Figura 7. Perceba-se que os procedimentos anteriormente descritos não permitem que para cada nova observação tenha-se uma predição sobre a classe a qual ela pertence. Esse comportamento prejudica principalmente aplicações iterativas que funcionam de forma *online*.

Nesse sentido, com o objetivo de adquirir um valor que corresponda às chances de uma sequência até o momento t ($t \leq Q$) pertencer a uma das classes do problema, pode-se alimentar o modelo com uma sequência de tamanho Q, onde as últimas t observações pertencem à sequência real e as primeiras $Q - t$ observações são valores de preenchimento, também conhecidos como valores de *padding*. Uma vantagem dessa abordagem é permitir o uso de modelos não sequenciais, como *Naive Bayes* ou MLP, para classificar a sequência, uma vez que a dependência entre observações pode ser tratada como dependência entre características. Por outro lado, o seu principal problema é o desperdício de capacidade de processamento no início da sequência, já que a maioria dos dados de entrada são preenchidos com valores padrão. Além da possível perda de desempenho nas métricas de avaliação quando a natureza do problema exigir que a dependência entre longas sequências deva ser considerada.

Figura 7 – Representação de duas sequências com tamanhos diferentes para um modelo que recebe uma sequência de entrada fixa de tamanho Q. A cada instante de tempo t, a t-ésima observação da sequência é adicionada à sequência fixa que alimenta o modelo. Assim, o modelo pode predizer a classe representada pelas t primeiras observações da sequência.



Fonte: Próprio autor.

Para a segunda abordagem, onde cada sequência pode ter uma quantidade variada de observações, modelos sequenciais são os mais indicados, pois, a priori, não se sabe qual o tamanho que a sequência terá quando da sua última observação. Sendo assim, modelos como HMM (*Hidden Markov Model*), CRF (*Conditional Random Fields*) e rede neurais recorrentes (RNNs - do inglês *Recurrent Neural Networks*) são possíveis candidatos. Vale ressaltar que os modelos que assumem a condição markoviana de que uma dada observação no tempo t depende apenas da observação no tempo $t-1$ (como o HMM e o CRF) podem não capturar longas dependências em uma sequência. Nesses casos, para problemas assim caracterizados, as RNNs podem ser mais efetivas.

2.7.2 Redes neurais recorrentes

Apesar da sua capacidade de resolver problemas complexos, as FCs e as CNNs, em suas configurações padrão, não são tão eficazes na solução de problemas que possuem dependências temporais entre os dados observados. Desse modo, não são indicadas para o processamento de linguagem natural, reconhecimento de ações e gestos em vídeos, dentre outros. Para esses tipos de problema, foram propostas as redes neurais recorrentes.

2.7.2.1 Arquitetura padrão

Para cada nova observação, uma RNN recebe a própria saída calculada na observação anterior. Por causa dessa realimentação é que surge o termo *recorrente*. Assim, em sua versão mais simples, uma RNN pode ser representada como descrito na Equação (2.3), onde σ corresponde a uma função de ativação do tipo sigmoide. Nessa equação, t representa o instante de tempo atual e $t-1$ o instante de tempo anterior. $\mathbf{W}$, $\mathbf{U}$ e $\mathbf{b}_x$ representam pesos treináveis, responsáveis por aplicar uma transformação na combinação entre a entrada, formada pela observação x_t, e a saída do instante anterior, h_{t-1}, de maneira a obter uma relação entre a observação atual e as observações anteriores.

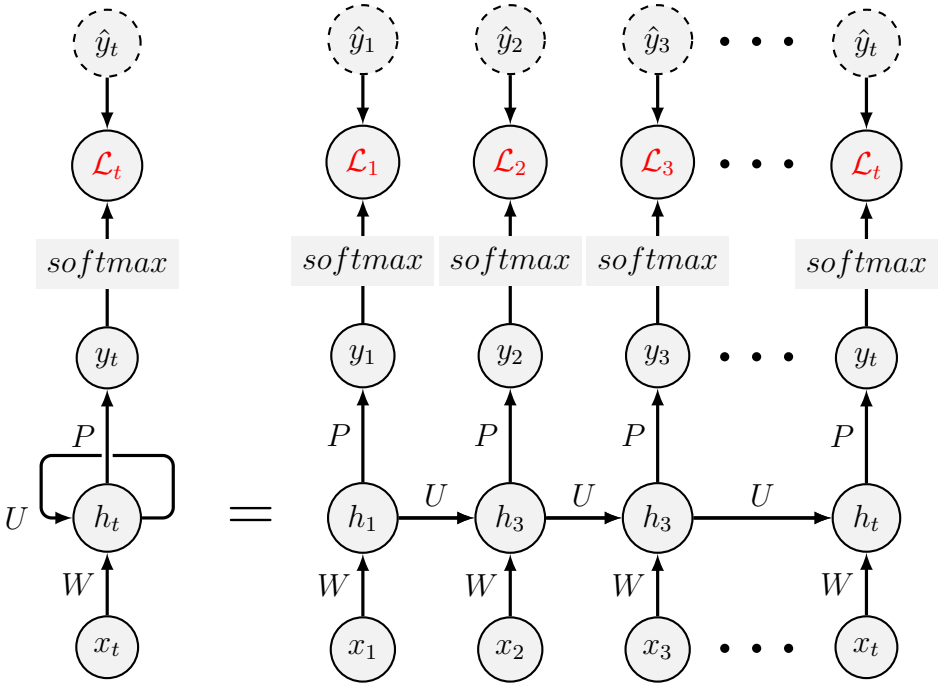
$$\mathbf{h}_t = \sigma(\mathbf{W}^T\mathbf{x}_t + \mathbf{U}^T\mathbf{h}_{t-1} + \mathbf{b}_x) \tag{2.3}$$

Assim, seja um problema de classificação de uma sequência S = $\{(\mathbf{x}_1,\mathbf{y}_1)$, $(\mathbf{x}_2,\mathbf{y}_2)$, $(\mathbf{x}_3,\mathbf{y}_3)$, ..., $(\mathbf{x}_T,\mathbf{y}_T)\}$ com T amostras, onde, para cada instante de tempo t, o par $(\mathbf{x}_t,\mathbf{y}_t)$ representa respectivamente a t-ésima observação e seu respectivo rótulo. Para poder treinar os parâmetros do modelo, é necessário definir uma função de custo. Nesse sentido, no instante t, ao observar $\mathbf{x}_t$, o modelo descrito na Equação (2.3) fornece $\mathbf{h}_t$, o qual pode ser transformado por uma matriz de parâmetros treináveis $\mathbf{P}$ e o vetor de *bias* $\mathbf{b}_h$, e ativar uma função *softmax*, conforme mostrado na Equação (2.4). Logo, $\hat{\mathbf{y}}_t$ representa a estimativa de $\mathbf{y}_t$ fornecida pelo modelo recorrente no instante t. Com isso, pode-se definir a função de

custo $\mathcal{L}$ que servirá de base para o treinamento dos parâmetros que definem o modelo. A Equação (2.4) traz uma representação completa do modelo, onde d é o número de classes, e a Figura 8 mostra uma representação gráfica do desdobramento dessa equação para a sequência S.

$$\begin{aligned} \mathbf{h}_t &= \sigma(\mathbf{W}^T\mathbf{x}_t + \mathbf{U}^T\mathbf{h}_{t-1} + \mathbf{b}_x) \\ \hat{\mathbf{y}}_t &= softmax(\mathbf{P}^T\mathbf{h}_t + \mathbf{b}_h) \\ \mathcal{L}_t &= -\sum_{i=1}^{d} \hat{\mathbf{y}}_t^{(i)} log(\mathbf{y}_t^{(i)}) \end{aligned} \tag{2.4}$$

Figura 8 – RNN e a sua forma desdobrada para uma sequência de tamanho t.



Fonte: Próprio autor.

Quanto maior for T, maior será o tamanho da sequência que se deseja representar. No entanto, como a função sigmoide $\sigma(\mathbf{x})$ sempre transforma os valores em $\mathbf{x}$ para o intervalo [0,1], a memória representada pela recorrência não é tão efetiva. Isso se deve ao fato de que a multiplicação repetida de valores entre 0 e 1 faz com que os resultado sejam cada vez menores. Fazendo com que grande parte da memória seja esquecida durante a evolução do modelo sobre a sequência. Além disso, como mostrado em Pascanu, Mikolov e Bengio (2013), esse tipo de arquitetura possui duas propriedades que podem prejudicar a resolução de problemas com longas sequências: a explosão e o desvanecimento do gradiente. Sendo assim, a fim de solucionar tais problemas, foram propostas as RNNs baseadas no

mecanismo de portas, as quais podem assumir três principais funções: leitura, escrita e esquecimento.

2.7.2.2 LSTM - *Long-Short Term Memory*

A LSTM é uma arquitetura de redes neurais recorrentes proposta por Hochreiter e Schmidhuber (1997) com o objetivo de solucionar o problema de desvanecimento do gradiente por meio do uso de portas (*gates*) que controlam o fluxo de informação. Dessa forma, ela possui quatro portas treináveis, as quais são responsáveis por capturar dependências longas e curtas em uma sequência. Uma célula LSTM recebe como entrada um vetor de observação (*input*), um estado oculto (*hidden state*) e uma célula de eco (*echo cell*). O vetor de entrada representa a observação atual; o estado oculto representa a memória de curto prazo e escolhe quais informações devem ser levadas em consideração na próxima observação; já a célula de eco representa a memória de longo prazo, assim, a cada nova observação, ela armazena informações importantes sobre a observação atual e esquece parte das informações armazenadas nas observações passadas, caso as considere menos significativas.

A LSTM tem sido usada principalmente para o processamento de linguagem natural (VASWANI et al., 2017; DEVLIN et al., 2019), mas, nos últimos anos, também tem sido usada nas tarefas de reconhecimento em vídeos. As Equações (2.5) - (2.10) representam as portas e as ativações de uma célula LSTM, respectivamente:

- *Forget gate* (f_t): desempenha a função da memória de longo prazo ao esquecer parte da memória armazenada na *echo cell*. Ela representa a porta de esquecimento.

$$\mathbf{f}_t = \sigma(\mathbf{W}_f^T \mathbf{x}_t + \mathbf{U}_f^T \mathbf{h}_{t-1} + \mathbf{b}_f). \tag{2.5}$$

- *Input gate* (i_t): seleciona parte da informação a ser armazenada e propagada para a *echo cell* da próxima observação, representando assim a porta de escrita.

$$\mathbf{i}_t = \sigma(\mathbf{W}_i^T \mathbf{x}_t + \mathbf{U}_i^T \mathbf{h}_{t-1} + \mathbf{b}_i). \tag{2.6}$$

- *Output gate* (o_t): seleciona parte da informação de entrada a ser propagada para a próxima observação por meio do *hidden state*.

$$\mathbf{o}_t = \sigma(\mathbf{W}_o^T \mathbf{x}_t + \mathbf{U}_o^T \mathbf{h}_{t-1} + \mathbf{b}_o). \tag{2.7}$$

- *Update gate* (g_t): normaliza a observação a fim de armazená-la na *echo cell* da próxima iteração. Parte dessa informação será esquecida após passar pela *input gate*, o que constitui a memória de curto prazo.

$$\mathbf{g}_t = \tanh(\mathbf{W}_g^T \mathbf{x}_t + \mathbf{U}_g^T \mathbf{h}_{t-1} + \mathbf{b}_g). \tag{2.8}$$

- *Next echo cell* (c_t): esquece parte das observações passadas e armazena parte da observação atual.

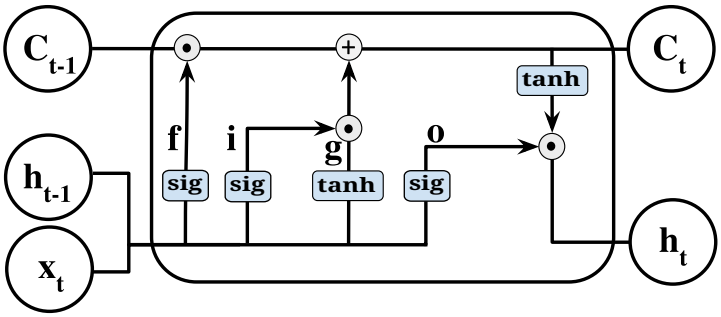
$$\mathbf{c}_t = f_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t. \tag{2.9}$$

- *Next hidden state* (h_t): seleciona parte da informação normalizada da *echo cell*, por meio da *output gate*. Juntamente com porta *update gate* representa a porta de leitura.

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t). \tag{2.10}$$

onde $\sigma(\bullet)$ representa a função de ativação sigmoide, $\tanh(\bullet)$ a função tangente hiperbólica, W_* e U_* são matrizes de pesos treináveis, assim como os vetores de *bias* b_*. Na Figura 9, é possível ver a representação gráfica de uma célula LSTM.

Figura 9 – Representação gráfica de uma célula (camada) LSTM.



Fonte: Próprio autor.

2.7.2.3 GRU - *Gated Recurrent Unit*

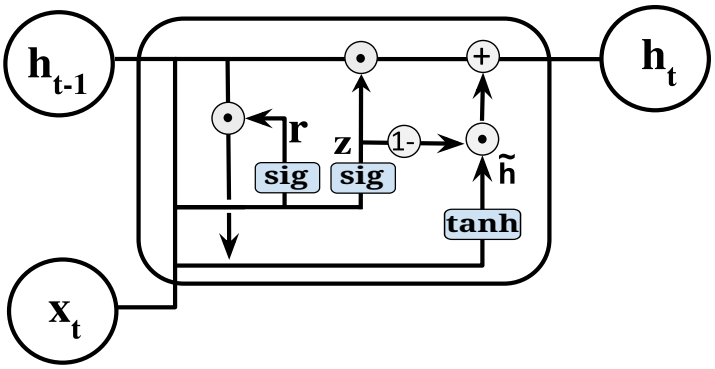
Apesar da efetividade da LSTM na resolução de problemas sequencias, o uso de quatro portas a torna pouco eficiente do ponto de vista computacional, quando possuem alta capacidade de aprendizagem (grande número de neurônios). Sendo assim, Chung et al. (2014) propuseram uma arquitetura semelhante à LSTM, a GRU, também baseada em portas de esquecimento, leitura e escrita, mas com uma representação interna reduzida. Para isso, a porta de esquecimento da GRU também desempenha a função de porta de leitura, e apenas o estado é realimentado (não possui célula de eco). Dessa maneira, diminui-se a capacidade do modelo e consequentemente o seu custo computacional.

Desde a sua criação, o modelo recorrente GRU tem se mostrado muito eficiente e é utilizado em um grande número de problemas, como Niu et al. (2019), Ghanbari e Ohler (2020), Vaswani et al. (2017), Wang et al. (2017), Nallapati et al. (2016) e Choi et al. (2016). No entanto, uma vez que não possui a célula de eco, não é indicado para problemas onde existem longas dependências entre as observações de diferentes instantes de tempo. Nestes casos, o LSTM ainda é o mais eficaz. Nas Equações (2.11) a (2.14), observa-se uma

descrição formal do funcionamento de uma célula GRU. Já na Figura 10, vê-se como a mesma pode ser graficamente representada.

$$r_t = \sigma(W_r^T x_t + U_r^T h_{t-1} + b_r) \tag{2.11}$$
$$z_t = \sigma(W_z^T x_t + U_z^T h_{t-1} + b_z) \tag{2.12}$$
$$\tilde{h}_t = \tanh(W_h^T x_t + r_t \odot U_h^T h_{t-1} + b_h) \tag{2.13}$$
$$h_t = (1 - z_t) \odot \tilde{h}_t + z_t \odot h_{t-1} \tag{2.14}$$

Figura 10 – Representação gráfica de uma célula (camada) GRU.



Fonte: Próprio autor.

2.8 Redes Neurais Bayesianas e incerteza

As redes neurais profundas são geralmente treinadas por algoritmos de otimização baseados no gradiente descendente estocástico (SGD - do inglês *Stochastic Gradient Descent*). Como o SGD usa o gradiente das funções de transferência aplicadas aos pesos que parametrizam o modelo, ele precisa que a função de custo e as funções de transferência sejam diferenciáveis em relação aos pesos. Isso implica que os pesos devem ser variáveis determinísticas. Portanto, em consequência, a maioria dos modelos de redes neurais profundas é determinístico e, por esse motivo, eles são incapazes de fornecer sua incerteza sobre suas predições. Como apresentado em Paulino et al. (2018), a incerteza é a falta de conhecimento absoluto sobre os estados da natureza, ou, em relação aos modelos preditivos, é a falta de conhecimento sobre a distribuição dos seus parâmetros. Admitir que os parâmetros do modelo são determinísticos, é assumir que não houve incerteza no seu processo de inferência. Isso claramente não é verdade, pois, para isso, seria necessário observar todas os possíveis elementos do conjunto que representa o domínio do problema, o que não é factível. Assim, pela necessidade de calcular a incerteza nesse tipo de modelo, surgiu a ideia de criação das Redes Neurais Bayesianas (BNNs - do inglês *Bayesian Neural Networks*).

Em um modelo Bayesiano, a distribuição a posteriori deve ser inferida aplicando-se a seguinte regra de Bayes:

$$p(\boldsymbol{\theta}|\mathcal{D}) = \frac{p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}}, \tag{2.15}$$

onde $p(\boldsymbol{\theta}|\mathcal{D})$ é a distribuição a posteriori sobre $\boldsymbol{\theta}$ após observar os dados $\mathcal{D}$; $p(\mathcal{D}|\boldsymbol{\theta})$ é a probabilidade de $\mathcal{D}$ dado $\boldsymbol{\theta}$, também conhecida como verossimilhança; $p(\boldsymbol{\theta})$ é a crença a priori sobre a distribuição de $\boldsymbol{\theta}$; e $\int p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})d\boldsymbol{\theta}$ é o termo de normalização (também conhecido como evidência ou probabilidade marginal). Uma leitura ampla sobre o método e a tomada de decisão bayesiana pode feita em Paulino et al. (2018).

2.8.1 Principais abordagens

2.8.1.1 Truque de reparametrização e *Bayes By BackProp*

Em muitos casos, o termo de evidência na Equação (2.15) torna a distribuição a posteriori difícil de calcular. No entanto, alguns trabalhos tentaram resolver esse problema por meio da Inferência Variacional (VI - do inglês *Variational Inference*) (BLEI; KUCUKELBIR; MCAULIFFE, 2017). Graves (2011) propôs em detalhes como usar a VI em redes neurais bayesianas para que sua distribuição a posteriori pudesse ser aproximada por uma distribuição gaussiana com parâmetros conhecidos.

Embora eficaz, a VI ainda não era uma tarefa fácil de realizar. Portanto, Kingma e Welling (2013) propuseram uma maneira de treinar uma BNN com VI, considerando uma técnica chamada truque de reparametrização, que consiste em amostrar os valores de ativação Z de uma camada l de uma distribuição gaussiana padrão fatorada (média zero e variância unitária) .

Nesse sentido, a camada l deve gerar dois valores, μ e σ, que representam, respectivamente, a média e a variância de uma distribuição gaussiana fatorada $N(\mu, \sigma)$. Em seguida, a ativação de l é extraída de $\mathbf{z} \sim N(\mu, \sigma)$. Com o objetivo de aproximar $\mathbf{z}$ de uma distribuição gaussiana padrão fatorada, $N(0,1)$, os autores utilizaram a inferência variacional. Um problema enfrentado é que, agora, como $\mathbf{z}$ é estocástico, o algoritmo SGD não pode ser usado para treinar os parâmetros de l. Para resolvê-lo, os autores propuseram reparametrizar $\mathbf{z}$ para que μ e σ fossem determinísticos em relação a $\mathbf{z}$ e, por consequência, diferenciáveis em relação a uma função de custo $\mathcal{L}$. Dessa forma, o algoritmo SGD pôde ser usado para treinar os parâmetros da camada l. A Equação (2.16) apresenta essa abordagem, onde o ruído ϵ é o responsável pela estocasticidade em $\mathbf{z}$.

$$\begin{aligned} \mathbf{z} &\sim N(\mu,\sigma) \\ \mathbf{z} &= \mu + \sigma\epsilon \end{aligned} \tag{2.16}$$

Mesmo com uma contribuição significativa, Kingma e Welling (2013) usaram $\mathbf{z}$ como a última camada de um *encoder*, não em todas as ativações ou pesos da rede. Portanto, Blundell et al. (2015) propuseram usar essa abordagem para criar um BNN considerando cada peso do modelo como uma distribuição em vez de uma variável determinística. O truque de reparametrização permitiu que eles usassem o algoritmo *back propagation* para treinar o modelo via SGD e usassem a VI para aproximar a distribuição a posteriori dos pesos de uma distribuição fatorada com parâmetros conhecidos. Essa abordagem é chamada *Bayes By BackProp* (BBB).

2.8.1.2 *MC e Variational Dropout*

Outras abordagens, como *MC Dropout* (GAL; GHAHRAMANI, 2016a) e *Variational Dropout* (VD) (KINGMA; SALIMANS; WELLING, 2015), usam o *dropout* para obter uma aproximação de um modelo bayesiano.

No *MC Dropout*, o modelo deve ter uma operador de *dropout* antes de cada uma de suas camadas parametrizadas. Assim, a aproximação bayesiana é alcançada desativando pesos aleatoriamente com base em uma distribuição de Bernoulli com a probabilidade de $1-p$, onde p é um hiperparâmetro. O nome *MC dropout* advém do processo de predição do modelo, que determina a distribuição das classes por meio da média de uma simulação de Monte Carlo (MC) de S predições com o *dropout* sempre ativado. Este processo é apresentado na Equação (2.21). Note-se que, como a cada simulação o *dropout* faz com que os resultados sejam distintos, mesmo que a entrada seja a mesma para as S predições, todas as simulações serão possivelmente diferentes. Dessa maneira, poder-se-á estimar um valor de incerteza medindo o quanto as S predições destoaram entre si.

O *Variational Dropout* utiliza a reparametrização local e a VI para treinar e aproximar uma rede neural de um modelo bayesiano. No truque de reparametrização utilizado pelo BBB (Equação (2.16)), após uma camada i receber $\mathbf{x}_i$ como entrada, ela primeiro amostra os pesos $\boldsymbol{\theta}_i$ de uma distribuição gaussiana fatorada $N(\mu_i, \sigma_i)$ e calcula a ativação $\hat{\mathbf{y}}_i = \boldsymbol{\theta}_i^T \mathbf{x}_i$ como o produto interno entre $\mathbf{x}_i$ e $\boldsymbol{\theta}_i$. Por outro lado, na reparametrização local utilizada no VD, as ativações são amostradas diretamente de uma distribuição gaussiana fatorada, conforme mostrada na Equação (2.17).

$$\begin{aligned} \mu_i &= \boldsymbol{\theta}_i^T \mathbf{x}_i \\ \sigma_i &= (\boldsymbol{\theta}_i^2)^T \mathbf{x}_i^2 \\ \hat{\mathbf{y}}_i &\sim N(\mu_i, \sigma_i) \\ \hat{\mathbf{y}}_i &= \mu_i + \sigma_i \epsilon \end{aligned} \tag{2.17}$$

onde, $\mathbf{b}^2 = \mathbf{b} \odot \mathbf{b}$.

Essa técnica de reparametrização local pode ser usada em conjunto com um ruído $\xi_i \sim N(1, \alpha)$ para obter a distribuição a posteriori $p(\omega_i|D) = N(\theta_i, \alpha\boldsymbol{\theta}_i^2)$, onde ω_i é o parâmetro variacional, $\boldsymbol{\theta}_i$ é o peso do modelo na camada i e $\alpha = p/(1-p)$, sendo p o hiperparâmetro utilizado no *MC dropout*. A Equação (2.18) apresenta a abordagem do *Variational Dropout*.

$$\hat{\mathbf{y}}_i = \boldsymbol{\theta}_i^T(\mathbf{x}_i \odot \xi_i) \tag{2.18}$$

Como ξ_i é extraído de uma distribuição gaussiana, a distribuição marginal $p(\hat{\mathbf{y}}_i|\mathbf{x}_i)$ também é uma distribuição gaussiana. Assim, pode-se amostrar $\hat{\mathbf{y}}_i$ diretamente de sua distribuição marginal $p(\hat{\mathbf{y}}_i|\mathbf{x}_i)$, conforme apresentado na Equação (2.19).

$$\begin{aligned} \mu_i &= \boldsymbol{\theta}_i^T\mathbf{x}_i \\ \sigma_i &= \alpha(\boldsymbol{\theta}_i^2)^T\mathbf{x}_i^2 \\ \hat{\mathbf{y}}_i &\sim N(\mu_i,\sigma_i) \\ \hat{\mathbf{y}}_i &= \mu_i + \sigma_i\epsilon \end{aligned} \tag{2.19}$$

Mesmo que p em *MC Dropout* seja um hiperparâmetro, em uma das propostas do VD, α pode ser tomado como um parâmetro treinável, o que possibilita atribuir diferentes níveis de importância para cada elemento de $(\boldsymbol{\theta}_i^2)^T\mathbf{x}_i^2$.

Em um modelo bayesiano, independentemente da abordagem empregada para inferir a distribuição a posteriori, a predição de uma observação $\mathbf{x}^*$ é calculada integrando-se a probabilidade de $\mathbf{x}^*$ sobre toda a posteriori (Equação (2.20)). Como esse processo envolve uma integração impossível de calcular, uma aproximação imparcial pode ser obtida por meio de uma simulação de Monte Carlo, conforme apresentada na Equação (2.21).

$$p(\mathbf{y}^*|\mathbf{x}^*) = \int p(D|\boldsymbol{\theta})p(\boldsymbol{\theta}|D)d\boldsymbol{\theta} \tag{2.20}$$

$$\approx \frac{1}{S}\sum_{s=1}^{S} p(\mathbf{y}^*|\mathbf{x}^*;\theta_s). \tag{2.21}$$

Aqui, S é o número de amostras, $\mathbf{y}^*$ é a distribuição de probabilidade estimada das classes para $\mathbf{x}^*$ e $\theta_s \sim p(\theta|D)$ é o s-ésimo parâmetro θ extraído da posteriori $p(\theta|D)$. Para o modelo VD, essa distribuição a posteriori é $p(\omega|D)$. No entanto, para o *MC Dropout*, ela é representada pela função de *dropout* presente em cada camada com parâmetros treináveis de rede. Análises mais detalhadas sobre o *MC Dropout* podem ser vistas em Gal (2016).

2.8.2 Incerteza

Existem dois tipos principais de incerteza na modelagem bayesiana: aleatória e epistêmica. Aleatória é a incerteza de um evento (também conhecido como incerteza irredutível). Em um problema de classificação, essa incerteza está relacionada ao evento que gerou as evidências (dados). Portanto, embora alguns trabalhos proponham maneiras de avaliar a incerteza aleatória de um modelo (KENDALL; GAL, 2017; HAFNER et al., 2018), essa não é uma tarefa fácil, pois, na maioria dos casos, não se sabe como as evidências foram geradas ou qual o evento as gerou.

A incerteza epistêmica, diferentemente da aleatória, avalia a incerteza do modelo sobre os dados e pode ser facilmente calculada quando o modelo é estocástico. Esse tipo de incerteza pode ser diminuído pela observação de mais dados. Portanto, ela é importante quando se quer saber qual classe precisa de mais dados para melhorar a predição do modelo. Uma explicação detalhada sobre a incerteza em redes neurais bayesianas também é dada em Gal (2016).

Neste trabalho, o interesse principal é determinar a incerteza do modelo sobre sua predição, que corresponde à sua incerteza epistêmica. Nesse sentido, em teoria, quanto mais dados ele recebe, mais confiante ele fica em suas predições. Nesse caso, é possível usar a incerteza epistêmica para perceber quando o modelo deve esperar por mais observações para aumentar sua certeza sobre a predição.

Segundo Gal (2016), a incerteza epistêmica de um modelo de rede neural bayesiana pode ser estimada pela entropia ou pela informação mútua. Sendo assim, no caso de uma simulação MC com S amostras, um modelo com C ações pode calcular a entropia dessas predições (amostras) usando a Equação (2.23) e a informação mútua pela Equação (2.24). A Equação (2.22) foi separada para facilitar o entendimento da Equação (2.23). Ela é responsável por calcular a média das S predições para cada uma das C classes. Dessa forma, de um ponto de vista mais pragmático, o termo $p(y = c|x; \theta_s)$ correspondente ao valor do índice c do vetor resultante da função de ativação *softmax* aplicada sobre a saída do modelo definido pelos parâmetros θ_s, após receber a entrada x.

$$\mathbb{E}_{pred}(x,c) = \frac{1}{S}\sum_{s=1}^{S} p(y = c|x; \theta_s) \tag{2.22}$$

$$\mathbb{H}(x) = -\sum_{c=1}^{C} \mathbb{E}_{pred}(x,c)\ \log(\mathbb{E}_{pred}(x,c)) \tag{2.23}$$

$$\mathbb{I}(x) = \mathbb{H}(x) + \frac{1}{SC}\sum_{c=1}^{C}\sum_{t=1}^{S} p(y = c|x; \theta_s) \log p(y = c|x; \theta_s) \tag{2.24}$$

3 RECONHECIMENTO DE GESTOS DINÂMICOS

> [...] one of the major points of gesture modes of operation is their naturalness. If you take away that advantage, it is hard to see why the user benefits from a gestural interface at all.
>
> Wexelblat (1997)

3.1 O problema

Como mencionado antes, reconhecer gestos dinâmicos não é uma tarefa simples devido à sua natureza temporal. Assim, mesmo utilizando DL, são necessárias técnicas sofisticadas para capturar informações temporais. Por exemplo, arquiteturas DL complexas como convolução 3D (JI et al., 2013; HARA; KATAOKA; SATOH, 2018), fluxo duplo (SIMONYAN; ZISSERMAN, 2014b; FEICHTENHOFER; PINZ; ZISSERMAN, 2016) ou fluxo duplo com convolução 3D (CARREIRA; ZISSERMAN, 2017b) são frequentemente usadas para executar reconhecimento de ações. Esses tipos de abordagens exigem um alto poder de processamento, implicando em uma grande restrição de seu uso em ambientes onde o tempo de resposta do reconhecedor é uma prioridade.

Objetivando contribuir para a resolução desse problema, em Barros et al. (2014b) os autores utilizaram uma técnica (chamada neste trabalho de *representação star*) capaz de condensar informações temporais dos gestos em uma única imagem em escala de cinza. Para esse fim, eles geraram uma imagem contendo o histórico de movimento calculado por meio da soma dos valores absolutos das diferenças entre *frames* consecutivos de um vídeo. Com essa abordagem, o problema do reconhecimento de gestos em vídeos pode ser visto como um problema de classificação de imagens. Dessa forma, pode ser usado TL para reconhecer gestos dinâmicos aplicando CNNs treinadas para outra tarefa de reconhecimento de imagens.

Embora essa abordagem possa ser interessante para o reconhecimento de gestos, ela tem três desvantagens principais: (*i*) gestos que podem ser distinguidos apenas por sua sequência temporal são particularmente difíceis de reconhecer, uma vez que a informação temporal foi reduzida inteiramente a uma representação espacial; (*ii*) o processo resulta em apenas uma imagem em escala de cinza, o que pode dificultar o uso da transferência de aprendizado, uma vez que CNNs pré-treinadas, em sua maioria, recebem como entrada uma

imagem com três canais (por exemplo, uma imagem RGB); (*iii*) por fim, a representação dificulta a utilização de modelos de natureza sequencial, uma vez que fornece apenas uma imagem para todo o vídeo.

Mesmo assim, considerando interessante a possibilidade de representar o gesto de um vídeo em apenas uma imagem, neste capítulo, será proposta uma nova *representação star* para gestos calculados a partir de cada vídeo de entrada, que, além de ser uma imagem RGB, também melhora a codificação das informações temporais do gesto. Para poder testar a proposta, será utilizado um classificador de gestos dinâmicos baseado em arquiteturas de DL, composto por um *ensemble* de CNNs previamente treinadas e fundidas por um mecanismo de atenção chamado *soft-attention*. Mesmo sabendo que o resultado de um *ensemble* de modelos geralmente é melhor que o de um modelo individual (GOODFELLOW; BENGIO; COURVILLE, 2016), o objetivo principal do seu uso é mostrar que o *soft-attention* é uma opção mais eficaz para fundir características extraídas por diferentes CNNs em comparação aos operadores de soma, média ou mesmo concatenação. Além disso, a fim de possibilitar o reconhecimento de gestos de modo *online*, será apresentada ainda uma versão iterativa da proposta anterior que possibilita fornecer uma representação de movimento cada vez que um novo *frame* do vídeo é recebido.

Sendo assim, um dos objetivos deste capítulo é o de obter resultados que fortaleçam a primeira hipótese levantada na Seção 1.3 de que: por meio de uma representação compacta de movimento de um vídeo, um método baseado apenas em imagens RGB pode ser tão efetivo para o reconhecimento de gestos dinâmicos quanto os métodos multimodais são. Para isso, os principais resultados das duas propostas serão comparados com os resultados de outros trabalhos utilizando um conjunto de dados multimodal. Em consequência, as principais contribuições deste capítulo são:

- Uma representação condensada do movimento presente em um vídeo RGB.
- Uma técnica capaz de fundir informações de várias CNNs de maneira a ponderar a informação que cada uma oferece para a classe a ser predita.
- Um modelo de reconhecimento de gestos dinâmicos baseado em arquitetura profunda capaz de fundir a saída de várias CNNs que recebem como entrada uma representação condensada de movimento.
- Uma representação iterativa da informação de movimento que pode ser utilizada em conjunto com modelos de natureza sequencial.
- Um modelo de reconhecimento de gestos dinâmicos baseado em arquitetura profunda capaz de predizer sequencialmente o gesto presente em um vídeo.

3.2 Trabalhos relacionados

O reconhecimento de gestos dinâmicos é um campo de pesquisa com crescente interesse nos últimos anos. Vários trabalhos têm concentrado esforços em desenvolver interfaces mais intuitivas para interagir com máquinas e outros dispositivos. No entanto, como um mesmo gesto pode ser representado de diferentes maneiras, reconhecer um gesto dinâmico é uma questão desafiadora. Além disso, a maneira como os gestos são realizados não depende apenas dos movimentos do corpo, mas também do aspecto cultural das pessoas que os executam. Consequentemente, ainda há muito trabalho a ser feito para se obter uma interface capaz de fornecer comunicação eficaz entre humanos e máquinas com base em gestos.

Por esse motivo, conjuntos de dados bem estruturados que representam uma variedade de significados de gestos são de suma importância para o desenvolvimento desta área de pesquisa. Competições como *Chalearn 2014: Looking At people (Track 3: Gesture Recognition*, lançaram um desafio para o reconhecimento de gestos, por meio dos conjuntos de dados Montalbano $V1$ (ESCALERA et al., 2013) e $V2$ (ESCALERA et al., 2014), que atendem a alguns dos requisitos mencionados acima. Cada um desses conjuntos de dados compreende aproximadamente 14.000 gestos representados em vídeos capturados de 27 pessoas diferentes para 20 classes distintas. O sensor usado para capturar os dados foi o Microsoft Kinect 360. Portanto, os dados têm natureza multimodal, pois consistem em RGB, profundidade, máscara de usuário, juntas $3D$ de esqueleto e áudio. A principal diferença entre as versões $V1$ e $V2$, além de uma anotação aprimorada na segunda versão, é a informação de áudio, presente apenas na primeira.

Embora o Montalbano não seja um conjunto de dados focados na HMI, devido ao seu tamanho e numerosas amostras, muitos trabalhos testam suas abordagens usando esses dois conjuntos de dados, como Neverova et al. (2016), Efthimiou et al. (2016), Li et al. (2017), Joshi et al. (2017) e Wang e Wang (2017a). Analisando tais trabalhos, eles podem ser separados em dois grupos principais. Primeiro, aqueles que visam solucionar o problema original do desafio (reconhecer uma sequência de gestos dinâmicos não segmentados de um vídeo) e, segundo, os que classificam os gestos segmentados usando os rótulos disponíveis nos conjuntos de dados. Ambos os grupos podem reconhecer os gestos usando dados multimodais ou apenas um tipo de informação.

No primeiro grupo, que tentou solucionar o problema original do desafio, a maioria das abordagens utilizam informações multimodais. Por exemplo, Neverova et al. (2016) utilizaram todas as informações disponíveis nos dois conjuntos de dados. Eles aplicaram todas as informações diferentes a um conjunto de CNNs. Como resultado, os autores obtiveram uma acurácia de 96,81% quando usaram todos os dados, incluindo o áudio, 93,1% usando apenas esqueleto 3D e 95,06% usando RGB e Profundidade (RGB-D).

Alguns trabalhos utilizaram outras fontes de dados, visando o mesmo objetivo. Por exemplo, Neverova et al. (2014) usaram juntas de esqueleto e profundidade para alcançar uma distância de *Levenshtein*[1] de 0,85, enquanto Pavlakos et al. (2014) fizeram uso de juntas de esqueleto, RGB e áudio, atingindo 0,88.

Em Pigou et al. (2018), os autores aplicaram informações de cor e profundidade como entradas de uma CNN 3D, seguidas por um classificador LSTM bidirecional. Com essa arquitetura, eles atingiram um índice de Jaccard[2] (JI - do inglês *Jaccard Index*) de 0,906 usando informações RGB-D. Uma restrição dessa abordagem é que o gesto deve ser feito dentro de uma janela de tempo de 64 *frames*. Consequentemente, o modelo pode ser prejudicado por gestos realizados com um número de *frames* diferente de 64. Além disso, mesmo competindo na prmeira modalidade, o método levanta grandes dúvidas quanto a sua possibilidade de reconhecimento de gestos *online*, uma vez que nenhum resultado sobre tempo de processamento do modelo foi dado. Além disso, visualizando os resultados, alguns vídeos atingiram um JI muito baixo, como 0,37, o que lança dúvida sobre os valor tão alto de 0,906.

No segundo grupo, os gestos segmentados foram usados para treinar e testar os reconhecedores de gestos. Diferentemente do primeiro grupo, seu objetivo principal foi reconhecer os gestos sem ter de enfrentar os problemas de detectar quando um gesto inicia e termina em uma sequência de *frames*.

Dentro desse último grupo, o trabalho apresentado por Li et al. (2017) alcançaram 91,6% de acurácia usando apenas as informações do esqueleto. Os autores representaram as articulações do esqueleto como um vetor e a sequência de esqueletos como uma imagem, onde as coordenadas (x, y, z) de cada articulação representam um píxel (R, G, B) na imagem. Usando diferentes representações da informação do esqueleto, Liu e Zhao (2019) e Liu et al. (2019) obtiveram, respectivamente, 93,2% e 93,8%. Por outro lado, em Chen e Koskela (2014), os autores usaram informações de profundidade e esqueleto para classificar os gestos, alcançando apenas 85,5% de acurácia. Outros trabalhos, como Chen e Koskela (2014), Yao, Gool e Kohli (2014), Wu e Shao (2014), Fernando et al. (2015), Escobedo e Camara (2015), Joshi et al. (2017), também usaram tipos diferentes de informações, mas não alcançaram um resultado melhor do que Liu et al. (2019). Além disso, nesses trabalhos, os vídeos foram cortados em clipes contendo apenas os gestos. Esse processo de recorte foi implementado usando os rótulos relacionados a cada gesto presente no conjunto de dados.

[1] A distância de *Levenshtein* é comumente utilizada para medir a semelhança entre duas sequências de caracteres. Ela quantifica quantas operações de deleção, inserção e substituição são necessárias para que as duas sequências sejam iguais. Como as sequências podem ser de tamanhos distintos, para garantir que a distância esteja entre [0,1], divide-se o valor calculado pelo número de elementos da sequência de maior tamanho.

[2] O índice de Jaccard é calculado como a interseção sobre a união das duas sequências. Assim, duas sequências são iguais quando tem esse índice é igual a 1 e são completamente diferentes quando o índice é igual a 0.

É importante mencionar que as articulações dos esqueletos são uma fonte de dados que condensa informações dinâmicas e de estrutura relacionadas ao gesto. Consequentemente, é adequado para ser usado no reconhecimento dinâmico de gestos. No entanto, esse tipo de dado é geralmente fornecido por sensores específicos, como o *Microsoft Kinect*, que foi adotado para adquirir o conjunto de dados mencionado. A Tabela 1 mostra um resumo dos melhores resultados obtidos nos conjuntos de dados Montalbano $V1$ e $V2$ utilizando diferentes tipos de informações.

Como antes mencionado, infelizmente os sensores do tipo Kinect não são facilmente encontrados em locais comuns, que geralmente possuem câmeras de vigilância convencionais. Portanto, abordagens que usam apenas imagens RGB tornam-se mais interessantes que métodos multimodais, os quais carecem de mais de um tipo de informação.

Tabela 1 – Lista dos principais trabalhos que utilizaram o conjunto de dados Montalbano. Os resultados representam a acurácia média entre as classes.

Trabalho	Tipo de dados	Resultados
Neverova et al. (2016)	Áudio, RGB-D e Esqueleto	96.81%
Neverova et al. (2016)	RGB-D	95,06%
Neverova et al. (2016)	Áudio	94,96%
Liu et al. (2019)	Esqueleto	93,80%
Efthimiou et al. (2016)	Áudio e RGB	93,00%
Escobedo e Camara (2015)	Esqueleto e RGB-D	88,38%
Wu et al. (2016)	Profundidade	82,62%
Fernando et al. (2015)	Esqueleto e Áudio	80,29%
Cao, Zhang e Lu (2015a)	RGB	60,07%

Fonte: Próprio autor.

No entanto, até a elaboração deste manuscrito, dentre os trabalhos que focaram no reconhecimento de gestos usando apenas informações RGB nos conjunto de dados Montalbano, o melhor resultado obtido foi 60% de acurácia. Duas possibilidades podem explicar esses resultados: ou as abordagens propostas foram focadas no reconhecimento de gestos usando apenas informações multimodais e não fizeram um esforço significativo no uso de apenas um tipo de informação, ou os gestos são muito difíceis de distinguir um do outro quando se usa exclusivamente informações de cor.

Sem usar o conjunto de dados Montalbano e considerando apenas imagens RGBs, os autores de Barros et al. (2014b) usaram uma variante da técnica da imagem do histórico de movimento (MHI - do inglês *Moviment History Image*) (BOBICK; DAVIS, 2001) para representar as informações de movimento contidas em uma sequência de vídeo. Essa representação (definida antes como *representação star*) foi usada juntamente com uma CNN para reconhecer gestos dinâmicos. Os resultado relatado usando um conjunto de dados desenvolvido pelos autores foi de 91,67% de acurácia. É importante mencionar que os gestos desse conjunto de dados são muito diferentes entre si (alta variância interclasses),

o que pode facilitar o processo de classificação. Mesmo assim, a *representação star* é particularmente interessante e, se melhorada, pode ser aplicada a conjuntos de dados mais desafiadores que o utilizado pelos autores.

Para um melhor entendimento, considerando um vídeo em escala de cinza com N *frames* contendo um gesto dinâmico, a *representação star* pode ser calculada em duas etapas principais, a saber:

Primeira etapa: calcula-se a soma acumulada da diferença absoluta entre os *frames* consecutivos (dois a dois) em tons de cinza do vídeo contendo um gesto. A Equação (3.1) representa esse processo.

$$\mathbf{M}(i,j) = \sum_{k=2}^{N} |\mathbf{I}_{k-1}(i,j) - \mathbf{I}_k(i,j)| w_k, \tag{3.1}$$

em que: N, chamado tamanho da memória, é o número de *frames* de cada clipe de vídeo que contém um gesto; (i,j) são as coordenadas de um píxel em um *frame*; $\mathbf{M}$ é a imagem resultante; $\mathbf{I}_k$ é o k-ésimo *frame* de um vídeo que contém um gesto dinâmico; $|\bullet|$ é o operador de módulo e $w_k = k/N$ é responsável por ponderar a diferença absoluta, dependendo de N.

Segunda etapa: máscaras de Sobel são aplicadas sobre a matriz $\mathbf{M}$ nas direções X e Y, obtendo as matrizes $\mathbf{M}_X$ e $\mathbf{M}_Y$, respectivamente. Por fim, a *representação star* é definida como três imagens em escala de cinza, correspondentes às matrizes $\mathbf{M}$, $\mathbf{M}_X$ e $\mathbf{M}_Y$.

Segundo os autores, os dois canais extras, $\mathbf{M}_X$ e $\mathbf{M}_Y$, oferecem uma melhor discriminação quanto ao tipo de movimento presente em $\mathbf{M}$. No entanto, ao usar uma rede CNN, geralmente as primeiras camadas convolucionais aprendem e assumem a função desses tipos de máscaras, durante o estágio de treinamento. Isso elimina a necessidade de aplicar as máscaras de Sobel sobre a matriz $\mathbf{M}$, fazendo com que apenas $\mathbf{M}$ seja repassada à rede CNN repetindo-a nos seus três canais.

Com isso em mente, este trabalho propõe alterações significativas sobre a *representação star* a fim de gerar uma imagem RGB, realmente colorida, que codifica o movimento do gesto contido em um vídeo. A nova representação é aplicada a uma arquitetura DL baseada em um conjunto de duas CNNs fundidas por um mecanismo de atenção (*soft-attention*). Para comprovar a qualidade da proposta, a mesma foi avaliada nos conjuntos de dados Montalbano, GRIT (TSIRONI et al., 2017) e isoGD (WAN et al., 2016). Além disso, uma versão iterativa da proposta será apresentada. Ela possibilitará a representação de movimento em modelos de natureza sequencial. Assim, uma arquitetura de DL baseada numa CNN como extrator de características e uma rede neural recorrente é utilizada a fim de fornecer uma predição do possível gesto para cada *frame* da sequência. Dessa maneira, ela poderá ser utilizada por sistemas interacionais que funcionam de forma *online*. Essa

proposta também será avaliada no conjunto de dados Montalbano e os seus resultados serão comparados com os da primeira proposta.

3.3 Proposta 1: *Star RGB* - condensando a informação de movimento

Nesta seção, será descrita a primeira solução proposta neste trabalho para o reconhecimento de gestos dinâmicos utilizando câmera estática. Duas etapas principais formam esta proposta:

- **Pré-processamento**: cada vídeo de entrada é representado como uma imagem RGB usando uma versão modificada da *representação star* mencionada na Seção 3.2. De antemão, todas as justificativas que serão apresentadas para a proposta desta etapa estão amparadas em vários experimentos, os quais não serão apresentados aqui para não estender a seção, mas que, para conhecimento, podem ser visualizados no Apêndice A.
- **Classificação**: um classificador de gestos dinâmicos é treinado usando um conjunto de CNNs. Especificamente, a imagem obtida na etapa de pré-processamento é passada como entrada para duas CNNs pré-treinadas. Os resultados dessas duas CNNs passam por um mecanismo de *soft-attention* e, depois de uma média ponderada, os resultados são enviados para uma camada totalmente conectada. Finalmente, um classificador *softmax* indica a classe à qual o gesto pode pertencer.

Os detalhes dessas duas etapas são fornecidos nas subseções a seguir.

3.3.1 Pré-processamento

A *representação star* não leva em consideração as informações de cor de cada *frame*, apenas a informação de intensidade. Consequentemente, a representação resultante é uma imagem em escala de cinza, calculada usando a Equação (3.1). Portanto, o primeiro objetivo aqui é o de representar as informações temporais presentes em um vídeo colorido usando uma versão aprimorada da proposta em Barros et al. (2014b).

Assim, para aproveitar as informações de cor, a diferença entre dois *frames* consecutivos, calculada como na Equação (3.1) utilizando os *frames* em tons de cinza, pode ser substituída pela Distância Euclidiana, conforme apresentada na Equação (3.2).

$$\mathbf{M}(i,j) = \sum_{k=2}^{N} ||\mathbf{I}_{k-1}(i,j) - \mathbf{I}_k(i,j)||_2, \tag{3.2}$$

onde $\mathbf{I}_k(i,j)$ representa o vetor RGB de um píxel na posição (i,j) do k-ésimo *frame*; e $||\bullet||_2$ representa a norma L_2.

No entanto, a Distância Euclidiana considera apenas a norma do vetor e avalia apenas a intensidade de cada imagem. Assim, uma solução mais efetiva seria incluir informações de magnitude e fase ao calcular a distância entre vetores RGB. Isso permitiria avaliar não apenas as mudanças na intensidade da imagem, mas também sua matiz e saturação. Nesse sentido, Samatelo e Salles (2012) propuseram uma métrica baseada na similaridade cosseno, descrita como mostrado nas Equações (3.3) e (3.4). Essas métricas são as utilizadas nesta proposta, a fim de construir uma versão melhorada da *representação star*.

$$\lambda = 1 - \cos(\theta) = 1 - \frac{\mathbf{I}_{k-1}(i,j)^T \mathbf{I}_k(i,j)}{||\mathbf{I}_{k-1}(i,j)||_2 ||\mathbf{I}_k(i,j)||_2}, \tag{3.3}$$

onde θ é o ângulo entre $\mathbf{I}_{k-1}(i,j)$ e $\mathbf{I}_k(i,j)$.

$$\mathbf{D}_k(i,j) = \left(1 - \frac{\lambda}{2}\right) . \left| ||\mathbf{I}_{k-1}(i,j)||_2 - ||\mathbf{I}_k(i,j)||_2 \right| . \tag{3.4}$$

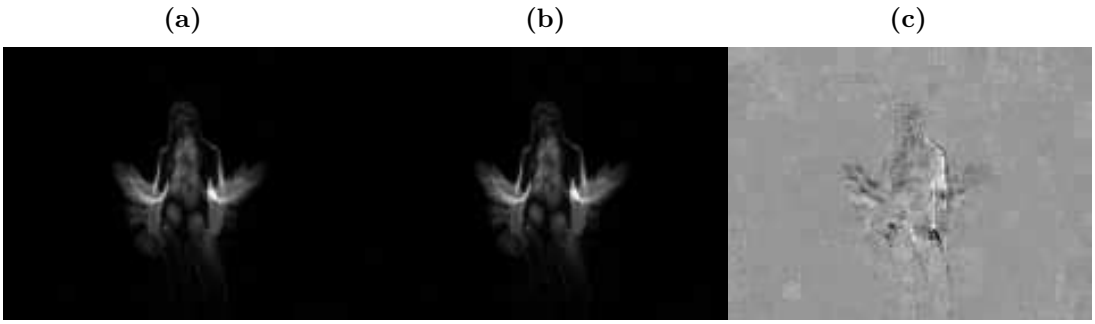
Como $\mathbf{D}_k(i,j) \approx ||\mathbf{I}_{k-1}(i,j) - \mathbf{I}_k(i,j)||_2$, pode-se usar a Equação (3.4) na Equação (3.2), obtendo-se, finalmente, a Equação (3.5), que é a representação proposta, chamada aqui de *representação star-cosseno*. Essa nova equação substitui a da *representação star*, dada pela Equação (3.1).

$$\mathbf{M}(i,j) = \sum_{k=2}^{N} \mathbf{D}_k(i,j) w_k. \tag{3.5}$$

Na Equação (3.5), a distância entre duas imagens consecutivas é calculada usando a diferença de intensidades, escalada por um valor numérico que depende do ângulo entre cada píxel RGB das imagens. Portanto, as informações de intensidade e cromaticidade também são levadas em consideração no processo de representação. A Figura 11 mostra os resultados da Equação (3.1) (*representação star*) e da Equação (3.5) (*representação star-cosseno*). Note-se que w_k pode ser usado para ponderar a informação temporal, assim como na Equação (3.1).

É possível perceber que com a nova proposta, mais informação do movimento pode ser extraída. No entanto, mesmo com essa representação, o resultado ainda é uma imagem em escala de cinza. Sendo assim, como as CNNs pré-treinadas geralmente recebem uma imagem RGB como entrada, a imagem resultante não é compatível para o uso direto em tais arquiteturas. Além disso, essa abordagem, mesmo utilizando $w_k = k/N$, ainda não resolve satisfatoriamente o problema da perda de informações temporais (veja-se os

Figura 11 – Comparando as abordagens. a) *representação star* proposta por Barros et al. (2014b); b) a *representação star-cpsseno*; c) a diferença entre ambas as representações. Em ambas as representações (a) e (b) foi usado $w_k = 1$ para todas as imagens.



Fonte: Próprio autor.

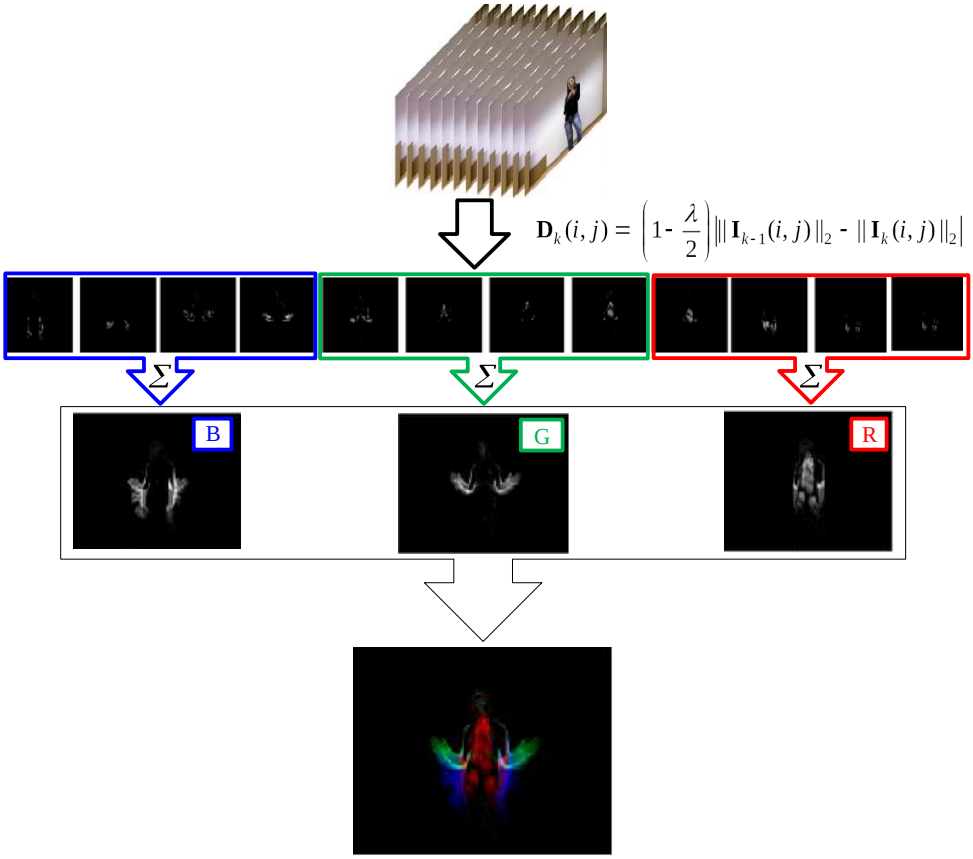
resultados do Apêndice A). Portanto, movimentos com o mesmo caminho, mas executados em direções diferentes, terão representações semelhantes.

Assim, para melhorar a representação temporal e, simultaneamente, criar uma imagem RGB como saída, é proposta a abordagem ilustrada na Figura 12 e descrita abaixo:

- Cada vídeo colorido contendo um gesto dinâmico completo é igualmente dividido em três subvídeos com $\lfloor \frac{N}{3} \rfloor$ *frames* cada, que, possivelmente, representarão as etapas *pre-stroke* (início), *stroke* (execução) e *post-stroke* (término) de um gesto dinâmico, conforme definido em McNeill (1992) e discutido em Liu et al. (2019). Se o número de *frames* que formam o vídeo não for divisível por três, o subvídeo central conterá $N - 2\lfloor \frac{N}{3} \rfloor$ *frames*;
- Para cada subvídeo resultante, a matriz $\mathbf{M}$ da *representação star-cosseno* é calculada usando a Equação (3.5).
- Então, o vídeo inteiro é representado por uma única imagem RGB, onde o canal R contém a matriz $\mathbf{M}$ calculada a partir do primeiro subvídeo, o canal G tem a matriz $\mathbf{M}$ do subvídeo central e o canal B a matriz $\mathbf{M}$ do último subvídeo.
- A imagem RGB é então normalizada para conter valores entre [0,1].

Como a saída da abordagem proposta é uma imagem com três canais, mesmo que cada canal não possua especificamente informações espectrais, para fins de representação, essa imagem é chamada *Star RGB*. Importante salientar que aqui, a w_k Equação (3.5) será sempre constante e igual a 1, uma vez que experimentos preliminares (Apêndice A) mostraram que a ponderação temporal torna-se desnecessária para a representação **Star**

Figura 12 – Representação *Star RGB* para o gesto *basta*.



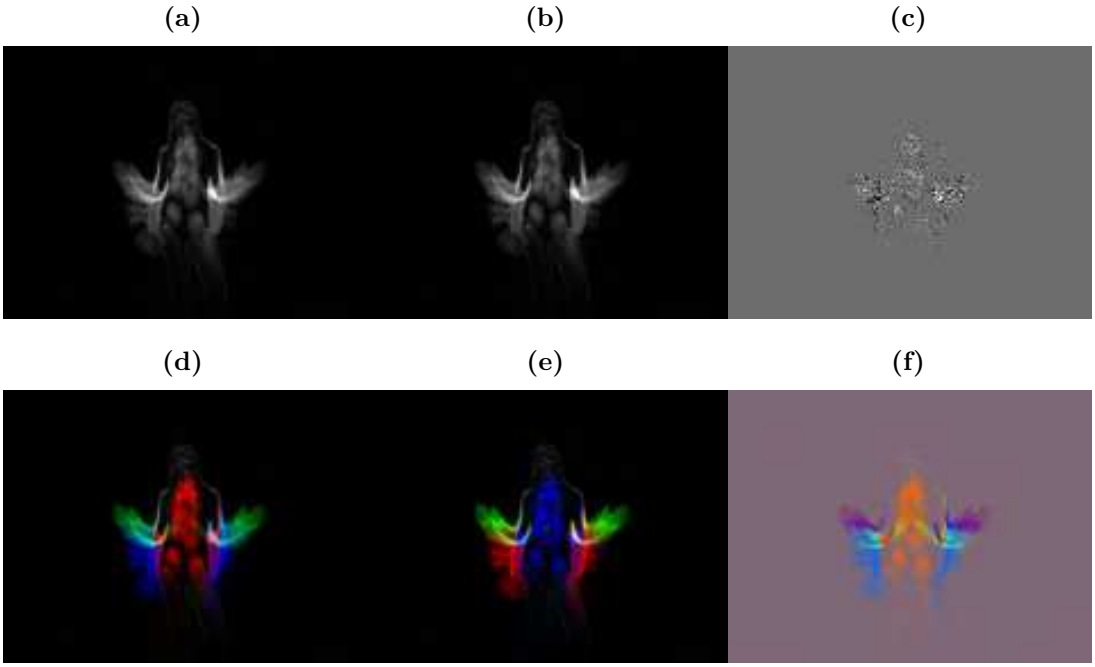
Fonte: Próprio autor.

RGB. Isso porque, possivelmente, a aplicação de $w_k \in (0,1]$ sobre o movimento entre duas imagens reduz a informação acumulada em cada canal R, G e B.

Além de resultar em uma representação multicanal e esparsa de um vídeo em cores, observa-se que a representação *Star RGB* tem outra vantagem quando a direção do movimento é uma característica importante para a distinção do gesto. Essa consideração é ilustrada pela simulação de dois gestos com os mesmos movimentos, mas com direções opostas. Por exemplo, a Figura 13 mostra as representações *Star RGB* calculadas a partir de dois vídeos, onde o segundo vídeo é gerado invertendo a ordem dos *frames* do primeiro. Portanto, ao comparar as Figuras 13a e 13b, nota-se que a *representação star* produz imagens semelhantes em escala de cinza para os dois vídeos, o que é evidente na Figura 13c que mostra a diferença entre eles. Por outro lado, a representação *Star RGB* produz duas imagens coloridas diferentes para cada vídeo, como mostradas nas Figuras 13d e 13e, enquanto a Figura 13f mostra a diferença entre cada canal das imagens 13d e 13e no formato RGB.

Esse resultado sugere que a representação *Star RGB* melhora a *representação star*, codificando mais informações temporais do que a abordagem anterior apresentada em Barros et al. (2014b). Além disso, provavelmente torna o modelo do classificador mais

Figura 13 – Comparando resultados entre as representações *star* e *Star RGB*. a) *representação star* de um vídeo em sua sequência original de *frames*; b) *representação star* de um vídeo em sua sequência invertida de *frames*; c) a diferença entre a) e b); d) representação *Star RGB* de um vídeo em sua sequência original de *frames*; e) representação *Star RGB* de um vídeo em sua sequência invertida de *frames*; f) a diferença entre d) e e). Observe-se que a imagem c) é quase zero. A maioria dos valores dos píxeis estão próximos de zero (algo em torno de $1e-7$) e foi normalizada para facilitar a visualização da diferença entre as imagens (a) e (b).



Fonte: Próprio autor.

robusto a movimentos semelhantes que representam diferentes gestos.

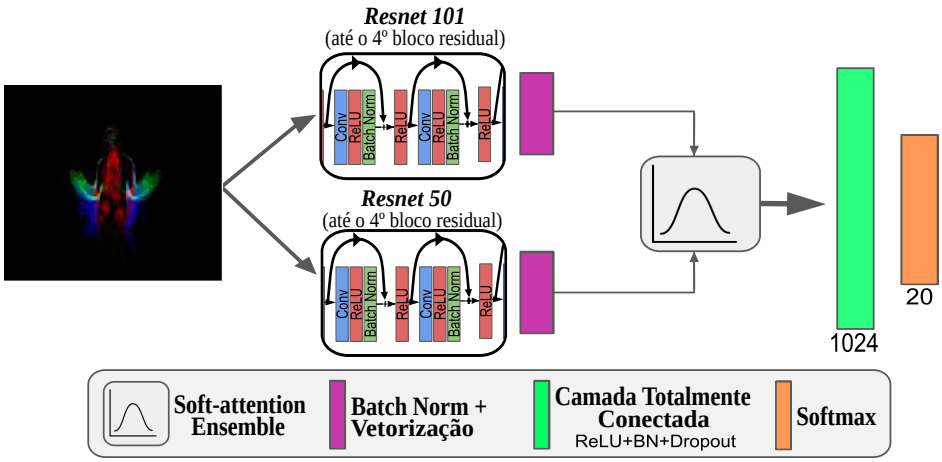
Em resumo, o *Star RGB* melhora a representação das informações temporais de um vídeo, possivelmente associando os canais de cores aos momentos de *pre-stroke*, *stroke* e *post-stroke* de um gesto dinâmico. Além disso, é mais adequado para modelos de treinamento baseados em CNNs, destinadas à classificação de imagens. Por fim, seu cálculo é fácil e rápido, podendo inclusive, ser utilizado em dispositivos para processamento na borda, termo conhecido como *edge computing* (SHI et al., 2016).

3.3.2 Classificação

Após o pré-processamento de uma sequência de vídeo para criar a representação *Star RGB* correspondente, o próximo passo é a classificação. A abordagem proposta para o classificador de gestos dinâmicos é mostrada na Figura 14 e compreende três principais partes: (*i*) um extrator de características com base em CNNs pré-treinadas; (*ii*) um conjunto de CNNs onde as características extraídas são fundidas por um mecanismo de

atenção; (*iii*) um classificador formado por duas camadas totalmente conectadas, sendo a última normalizada por uma função *softmax*. Cada uma dessas partes é explicada a seguir.

Figura 14 – Proposta de classificador de gesto dinâmico.



Fonte: Próprio autor.

3.3.2.1 Extrator de características

O extrator de características é baseado na CNN *Resnet* (HE et al., 2016b), especializada na classificação de imagens, pré-treinada usando o conjunto de dados *ImageNet*. Esse conjunto de dados foi lançado no concurso ILSVRC-2014 e contém mais de 1,2 milhão de imagens distribuídas em mais de 1.000 categorias distintas (RUSSAKOVSKY et al., 2015). A *Resnet* é uma das arquiteturas de CNN com maior uso na tarefa de classificação de imagens. Além disso, essa arquitetura integra as principais soluções estado da arte de vários problemas de visão computacional. Esses excelentes resultados são uma consequência dos seus blocos residuais, que podem atenuar o problema de desvanecimento do gradiente, mesmo em uma arquitetura muito profunda.

A *Resnet* foi escolhida após um procedimento de seleção empírica, onde as *Resnet 101* e *50* obtiveram os melhores resultados. Esse procedimento avaliou o desempenho de sete arquiteturas distintas (*Resnet (18, 34, 50, 101, 121)*, VGG16 (SIMONYAN; ZISSERMAN, 2014d) e DenseNet (GUO et al., 2016)) ao inserir a imagem RGB gerada na etapa de pré-processamento.

3.3.2.2 *Ensemble*: fundindo as características

Os mapas de características correspondentes à saída do quarto bloco residual de cada Resnet são passados sequencialmente para o mecanismo de atenção, referido aqui como *soft-attention*. Esse mecanismo permite fundir os mapas de características das CNNs

por meio de uma soma ponderada, onde os pesos são calculados de acordo com sua importância para a classe prevista.

O *soft-attention* foi escolhido após as seguintes considerações: (*i*) deve avaliar as características de acordo com sua importância para a tarefa, diferentemente de outros tipos de fusão, como somatório, média aritmética, concatenação, dentre outros; (*ii*) a natureza sequencial do conjunto de atenções evita a concatenação de todos os mapas de característica das entradas, o que aumentaria o tamanho do vetor de entrada gerado pelo conjunto.

A arquitetura do *soft-attention* é compartilhada por todos os mapas de características e é composta por uma camada totalmente conectada com 128 neurônios usando a ReLU (NAIR; HINTON, 2010b) como função de ativação e uma camada de saída de 1 neurônio com função de ativação linear. Assim, ela recebe um vetor (mapa de características vetorizado de uma CNN) como entrada e gera apenas um valor. A operação sequencial da *soft-attention* é mostrada na Figura 15.

Seja N_{CNN} o número de mapas de características das CNNs que serão fundidos pelo *soft-attention*. Primeiramente, cada mapa de características vetorizado de uma CNN é dado como entrada ao mecanismo de atenção, gerando um vetor com N_{CNN} elementos, e logo após é normalizado usando uma função *softmax*. Em seguida, a soma ponderada dos N_{CNN} mapas de características vetorizado é calculada usando os elementos do vetor normalizado como coeficientes de ponderação. Neste trabalho, utilizou-se $N_{CNN} = 2$, dado que o extrator de características é baseado em duas CNNs. Vale ressaltar que, ao usar esse mecanismo, todos os mapas de características devem ter as mesmas dimensões.

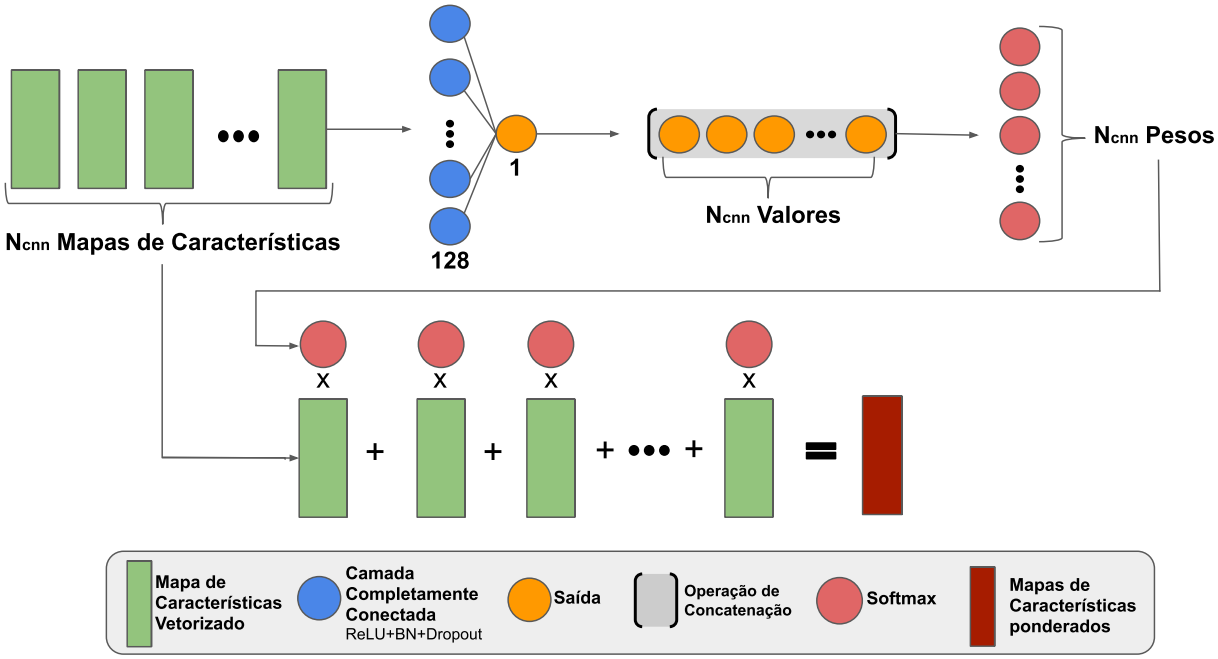
É importante mencionar que a técnica de *batch normalization* (IOFFE; SZEGEDY, 2015) é aplicada antes que os mapas de características sejam passados para o *soft-attention*. Isso é feito uma vez que os mapas de características podem não ter a mesma média e o mesmo desvio padrão. Dessa forma, o uso direto do mecanismo de atenção não forneceria um resultado preciso quando se tratasse da importância de cada mapa de características para o problema.

3.3.2.3 Classificador

A saída do *soft-attention* alimenta um classificador composto de duas camadas ocultas de 1024 neurônios com *batch normalization*, *dropout* e a função de ativação ReLU; e uma camada de saída de C neurônios com uma função de ativação *softmax*, sendo C o número de classes de gestos pertencentes a cada conjunto de dados utilizado. A função de ativação *softmax* na última camada fornece um *score* entre 0 e 1 relacionado a cada imagem de entrada indicando seu grau de pertencimento a um dos C gestos.

Além disso, a função de custo usada neste trabalho é a média da entropia cruzada,

Figura 15 – *Soft-attention ensemble.*



Fonte: Próprio autor.

calculada para cada *minibatch* e regularizada pela norma $L1$ sobre o conjunto de pesos, que inclui os dois extratores (CNN), *soft-attention* e classificador de características. Esse tipo de regularização foi aplicado a fim de forçar a esparsidade dos pesos, o que, em teoria, pode lidar melhor com imagens de entrada esparsas (veja-se a Figura 12).

Para facilitar a apresentação dos resultados, a solução completa (Star RGB, extratores, *soft-attention* e classificador) será chamada aqui de $\boldsymbol{Star\ RGB}_{SoftAtt}$.

3.3.3 Experimentos

Esta seção discute os conjuntos de dados usados ao longo dos experimentos, a implementação e o treinamento da arquitetura proposta e, finalmente, os resultados obtidos na etapa de avaliação.

Assim, são realizados três experimentos diferentes: o primeiro experimento é realizado para avaliar a arquitetura proposta e o impacto do mecanismo de atenção no desempenho do classificador de gestos; o segundo experimento tem como objetivo avaliar o uso do *Star RGB* na arquitetura proposta; já o terceiro tem o objetivo de testar o *Star* $RGB_{SoftAtt}$ em uma aplicação em tempo real.

Os principais experimentos que avaliam a efetividade do *Star* $RGB_{SoftAtt}$ foram realizados em três conjuntos de dados distintos, o quais serão apresentados a seguir.

3.3.3.1 Conjunto de dados de gestos Montalbano

Esse conjunto de dados foi lançado no desafio *Chalearn: Look at People* - 2014 (ESCALERA et al., 2014) e compreende 13.206 gestos culturais/antropológicos italianos (6.862 para treinamento, 2.765 para validação e 3.579 para testes), distribuídos entre 20 classes distintas. Todos os gestos foram capturados usando um sensor Kinect 360 e apresentam informações multimodais: RGB, profundidade, esqueleto e máscara do usuário. No desafio, os candidatos poderiam usar qualquer dado fornecido para reconhecer uma sequência de gestos (entre 8 e 20 por vídeo) realizados em um vídeo contínuo. A Figura 16 apresenta uma amostra de cada gesto contido neste conjunto de dados.

Esse é um dos maiores conjuntos de dados atualmente lançados, focado apenas no problema do reconhecimento de gestos dinâmicos, o que justifica sua escolha neste trabalho. Além disso, como mencionado por Tsironi et al. (2017), os outros conjuntos de dados disponíveis para o reconhecimento de gestos dinâmicos baseados em imagens RGB são muito pequenos e são capturados geralmente com foco apenas nas informações das mãos e não no corpo inteiro.

Como o objetivo aqui não é o de competir no *Chalearn*, os gestos de vários vídeos foram segmentados (transformados em pequenos clipes), de acordo com os rótulos fornecidos no conjunto de dados. Assim, cada vídeo de treinamento ou do conjunto de teste contém apenas um gesto dinâmico. Dessa forma, o problema não é mais o de reconhecer uma sequência de gestos em um vídeo, mas sim o de classificá-los dentre as 20 diferentes classes de gestos dinâmicos.

Após a segmentação de todos os vídeos, a técnica proposta foi aplicada, conforme apresentada na Seção 3.3.2. A Figura 17 apresenta uma amostra de cada gesto mostrado na Figura 16 após o cálculo do *Star RGB*.

3.3.3.2 Conjunto de dados de gestos GRIT

Para avaliar a proposta em um conjunto de gestos usados na interação humano-robô, utilizou-se o conjunto de dados GRIT (do inglês *Gesture Commands for Robot InTeraction*), que compreende 543 gestos, distribuídos em nove classes distintas. Ao contrário do conjunto de dados Montalbano, este contém um gesto dinâmico por vídeo. Além disso, como foi criado para a interação com robôs, seus gestos são bastante diferenciados. A Figura 18 ilustra uma representação de cada um dos nove gestos disponíveis no conjunto de dados. Além disso, a Figura 19 ilustra o resultado do *Star RGB* aplicado a amostras de cada uma das nove classes de gestos presentes no GRIT.

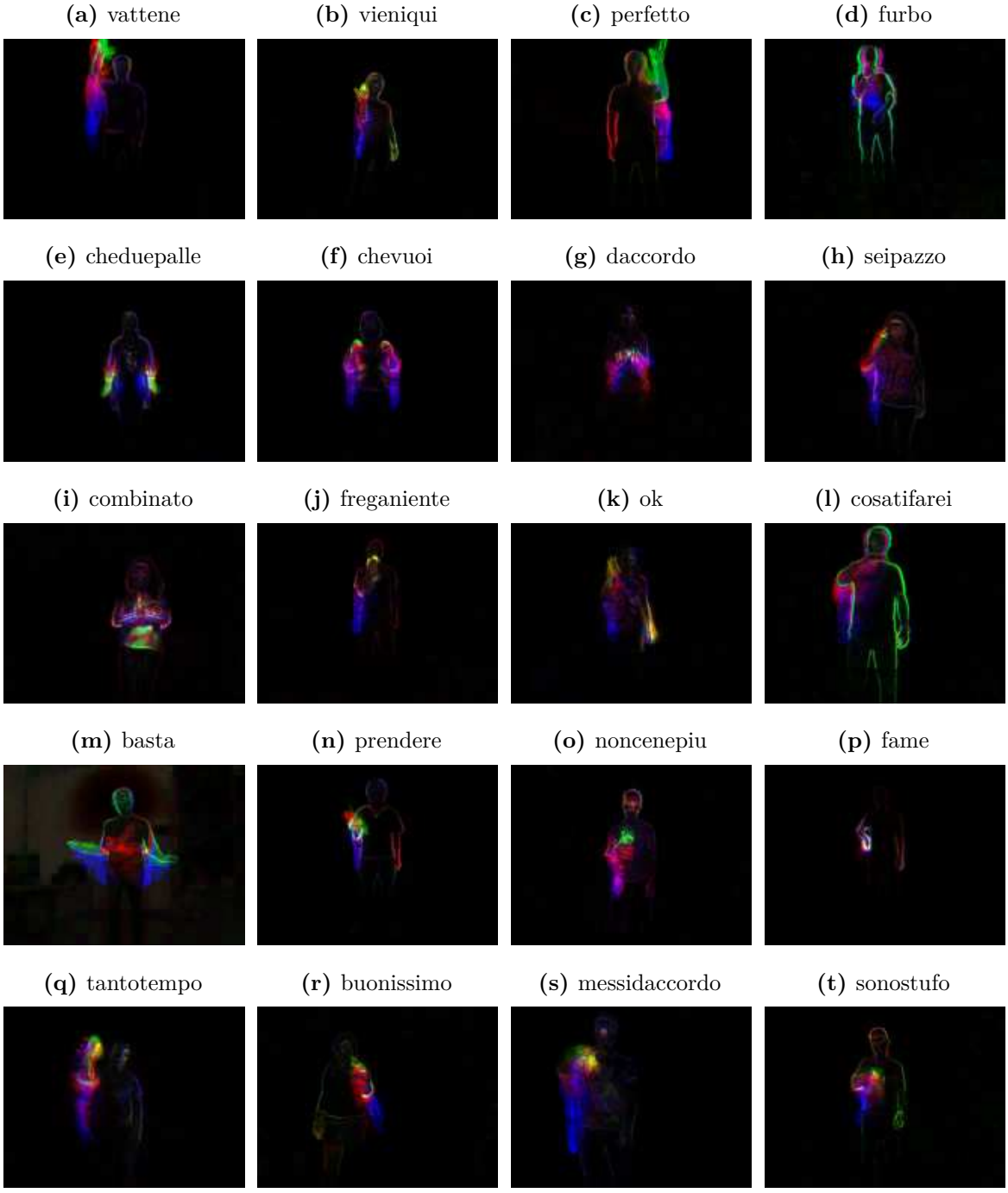
Figura 16 – Amostra de cada um dos 20 gestos presentes no conjunto de dados Montalbano (ESCALERA et al., 2014). Cada figura representa um gesto emblemático italiano. Tradução: (a) *Vattene* (vá embora), (b) *Viene qui* (venha aqui), (c) *Perfetto* (Perfeito), (d) *E un furbo* (Astuto), (e) *Che due palle* (Sem graça), (f) *Che vuoi* (O que você quer?), (G) *Vanno d'accordo* (Eles ficaram juntos), (h) *Sei pazzo* (você está louco?), (i) *Cos hai combinato* (O que você fez?), (j) *Nonme me frega niente* (Não me interessa), (k) *Ok* (Ok), (l) *Cosa ti farei* (O que você faria?), (M) *Basta* (já basta) , (n) *Le vuoi prendere* (você quer pegar), (o) *Non ce ne piu* (nada de mais), (p) *Ho fame* (estou com fome), (q) *Tanto tempo fa* (isso foi há muito tempo), (r) *Buonissimo* (delicioso), (s) *Si sono messi d'accordo* (Eles concordaram), (t) *Sono stufo* (estou cansado disso)

Fonte: Montado pelo próprio autor a partir dos vídeos do conjunto de dados Montalbano.

3.3.3.3 Conjunto de dados de gestos IsoGD

O IsoGD (WAN et al., 2016) é um grande conjunto de dados que contém 47.933 gestos, distribuídos entre 249 classes distintas. Foi lançado no desafio *ChaLearn LAP*

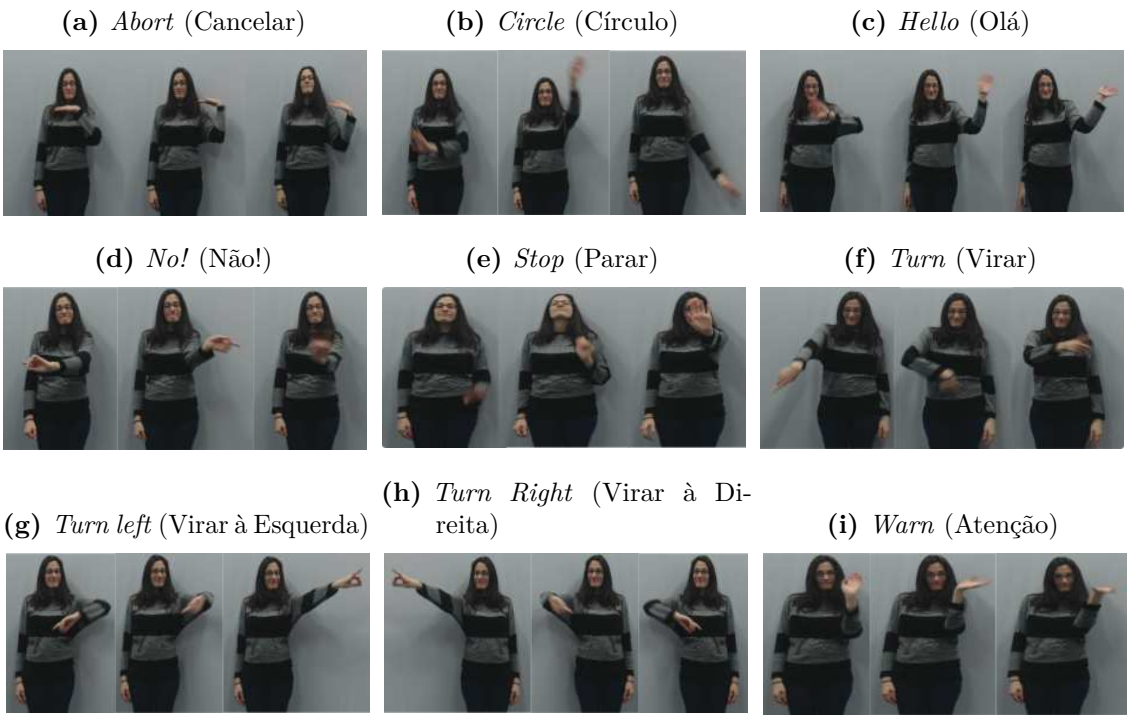
Figura 17 – Amostras do *Star RGB* calculado para os gestos presentes no conjunto de dados Montalbano.



Fonte: Próprio autor.

RGB-D Isolated Gesture Dataset - 2016 e foi capturado usando um sensor do tipo Kinect 360 a uma taxa de 20 FPS, contendo imagens RGB e de profundidade. Como no conjunto de dados GRIT, cada clipe de vídeo representa apenas um gesto e possui, em média, 47 imagens por gesto. Ele contém muitas categorias diferentes, como gestos da linguagem corporal, mudras indianos, gesticulações associadas à fala, ilustradores, emblemas, sinais e

Figura 18 – Amostra de cada uma das nove classes de gestos presentes no conjunto de dados GRIT.



Fonte: Montado pelo próprio autor a partir dos vídeos do conjunto de dados GRIT.

pantomimas.

Além de ter muito mais classes que o Montalbano e o GRIT, esse conjunto de dados tem outra particularidade: por não ter sido levantado em um laboratório, e sim pelos próprios usuários em suas residências, em alguns vídeos, diferentes partes do corpo podem estar oclusas durante o movimento e os usuários podem ter executados alguns gestos de maneira incorreta ou simplesmente não tê-los executado. Portanto, o IsoGD é consideravelmente mais complexo do que os outros dois conjuntos de dados usados neste trabalho. Ele proporciona um ambiente mais desafiador para testar a efetividade tanto do *Star RGB*, quanto do *Star* $RGB_{SoftAtt}$. A Figura 20 mostra algumas amostras do conjunto de dados IsoGD e sua respectiva imagem obtida com o *Star RGB*.

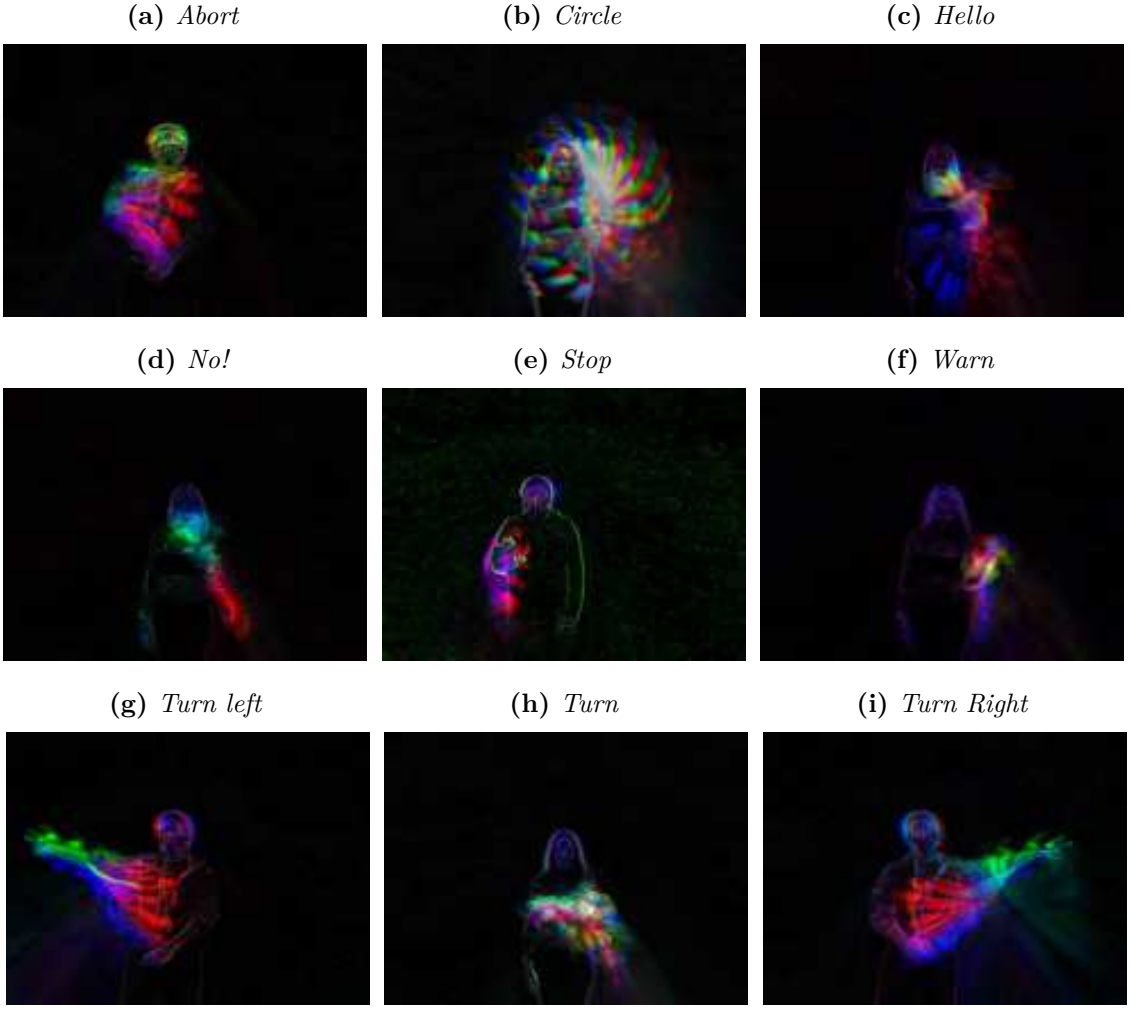
3.3.3.4 Implementação e treinamento

A arquitetura proposta foi implementada usando o *pytorch* V1.0, um software de código aberto desenvolvido pelo grupo de pesquisa em inteligência artificial do *Facebook*[3] para aprendizado de máquina (PASZKE et al., 2017).

O computador usado nas experiências tem a seguinte configuração:

[3] <https://research.fb.com/category/facebook-ai-research/>

Figura 19 – Amostras do *Star RGB* calculado para os gestos presentes no conjunto de dados GRIT.

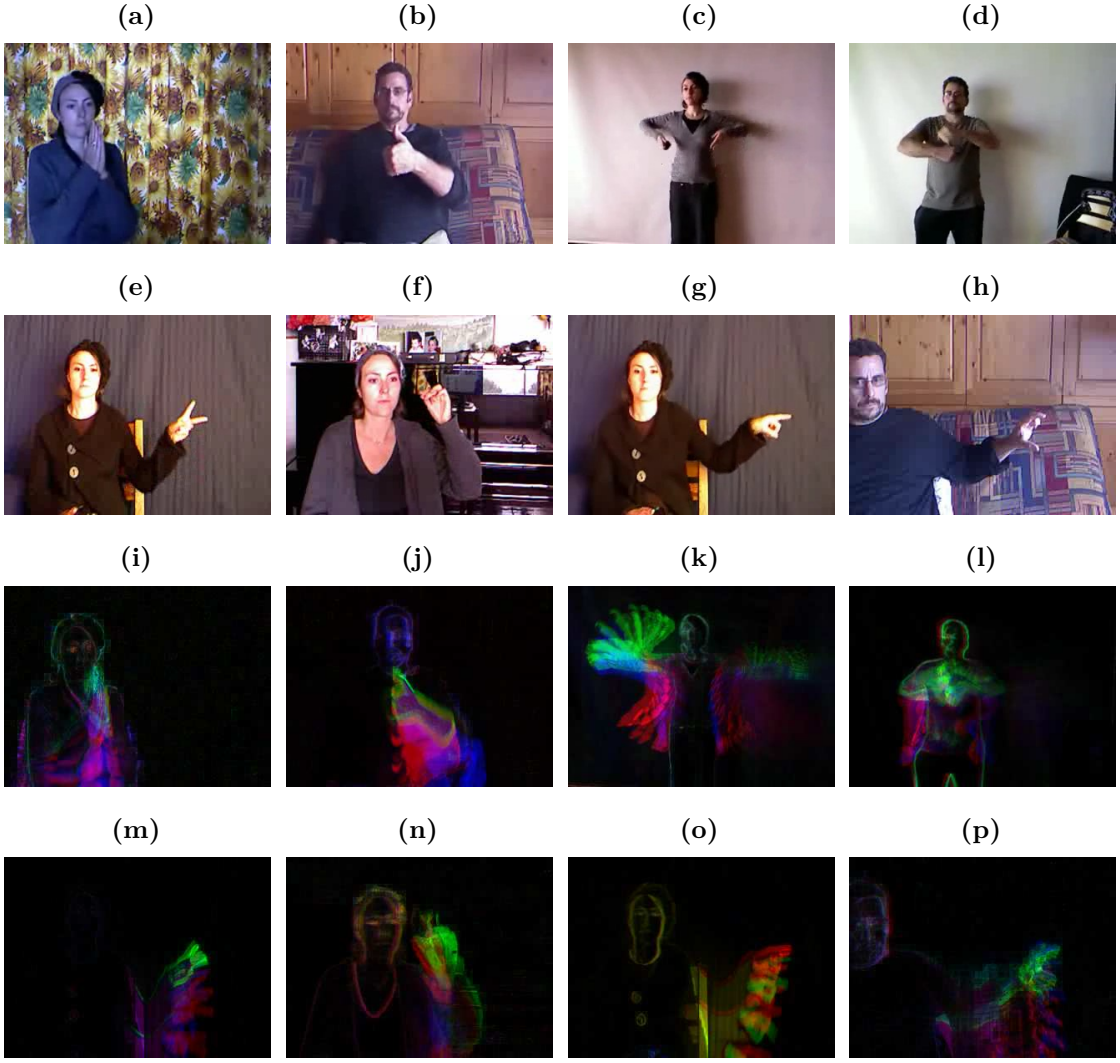


Fonte: Próprio autor.

- sistema operacional Linux Ubuntu Server, distribuição 16.04;
- processador Intel Core $i7 - 7700$, $3{,}60GHz$ com 4 núcleos físicos;
- 32 GB de RAM;
- $1TB$ de unidade de armazenamento (disco rígido);
- placa gráfica (GPU) Nvidia Titan V, com $12GB$ de memória dedicada.

Durante a etapa de treinamento, algumas técnicas de *data augmentation* foram aplicadas. Especificamente, para os conjuntos de dados Montalbano e IsoGD, foi realizado um recorte aleatório de tamanho 110×120 píxeis, um espelhamento horizontal aleatório, uma rotação aleatória entre $\pm 5^o$, e um ruído gaussiano, com média $\mu = 0$ e desvio padrão $\sigma = 1$. Por outro lado, para o conjunto de dados GRIT, devido aos gestos das classes *Virar à direita* e *Virar à esquerda*, apenas o espelhamento não foi realizado. Além disso, a Tabela 2 apresenta os hiperparâmetros usados durante o treinamento dos modelos em

Figura 20 – Amostras de gestos do conjunto de dados isoGD e suas respectivas representações *Star RGB*. Amostra/*Star RGB* = (gesto): (a)/(i) = (11 - mudra2/anjali), (b)/(j) = (8 - mudra1/shikhara), (c)/(k) = (181 - sinais de helicóptero/mover para a direita), (d)/(l) = (207 - sinais de um árbitro de luta livre/reversão), (e)/(m) = (222 - sinais de cirurgião/tesoura reta), (f)/(n) = (21 - gestos italianos/*bellissima* - linda), (g)/(o) = (220 - sinais de cirurgião/bisturi) e (h)/(p) = (223 - sinais de cirurgião/seringa).



Fonte: As imagens (a)-(h) foram extraídas do conjunto de dados isoGD, a demais são do próprio autor.

cada um dos três conjuntos de dados. Os hiperparâmetros apresentados foram obtidos por meio de experimentos empíricos. Apesar de poder ter utilizado técnicas automáticas como *grid search*, *random search* ou otimização bayesiana, optou-se por monitorar o efeito de cada hiperparâmetro sobre cada modelo treinado. Essa decisão foi tomada a fim de adquirir conhecimento técnico que ajude a delimitar melhor o conjunto de valores para os hiperparâmetros que possivelmente alcançarão melhores resultados em tais abordagens automáticas nos experimentos futuros. Nesse sentido, ao acelerar-se o processo

de treinamento com a escolha de valores próximos da região ótima de busca, além de otimizar-se o tempo de execução dos experimentos, evita-se o consumo desnecessário de energia elétrica e o desgastes dos dispositivos computacionais.

Tabela 2 – Hiperparâmetros utilisados no treinamento dos modelos.

Hiperparâmeters	Montalbano	GRIT	isoGD
Tamanho do lote	64	8	128
Número máximo de épocas	100	100	200
Taxa de aprendizagem	1e−4 (CNN) 1e−3 (FC)	1e−4 (CNN) 1e−3 (FC)	5e−4 (CNN) 5e−3 (FC)
Decaimento da taxa de aprendizagem		1% por época	
Dropout	0,2	0,2	0,15
Otimizador		Adam	

Fonte: Próprio autor.

Finalmente, em relação à etapa de treinamento, é adotada uma estratégia de parada antecipada. Assim, o treinamento é executado até que uma acurácia média maior que 99% sobre o conjunto de treino seja alcançada nas últimas 5 épocas do treinamento ou após a acurácia de validação ultrapassar os 97%.

Devido à compatibilidade de implementação, como cada conjunto de dados possui imagens de diferentes dimensões, é necessário redimensionar cada imagem de entrada. Portanto, em relação à etapa de treinamento, cada imagem é redimensionada para 120×160 píxeis antes dos procedimentos de *data augmentation*. Para as etapas de teste e validação, cada imagem é recortada a partir do seu ponto central, resultando em uma imagem de 110×120. Importante salientar que para todos os conjuntos de dados os resultados dos testes foram avaliados apenas uma vez utilizando os modelos que obtiveram as melhores acurácias médias sobre seus respectivos conjuntos validação.

3.3.3.5 Avaliação de desempenho

Para problemas de classificação com várias classes, é comum analisar o resultado usando a acurácia como métrica. No contexto do reconhecimento de gestos, a acurácia é calculada como sendo o número de gestos classificados corretamente dividido pelo número total de gestos presentes no conjunto de dados de teste.

Para os conjuntos de dados Montalbano e IsoGD, a acurácia alcançada por cada classe será apresentada em tabelas. Além disso, para melhorar a visualização do comportamento do classificador, os resultados do Montalbano serão descritos por meio de uma matriz de confusão, que apresenta uma linha para cada classe alvo e uma coluna para cada classe prevista. Assim, o valor de cada célula dessa matriz (linha r, coluna c) indica o número de gestos pertencentes à classe r que foram classificados como classe c. Como o

IsoGD possui 249 classes, por questões de dimensionalidade, a matriz de confusão de sua predição não será apresentada.

Visando uma comparação justa, para o conjunto de dados GRIT, o procedimento de avaliação original foi executado conforme descrito em Tsironi et al. (2017). Assim, realizaram-se cinco experimentos do tipo *hold-out*. Em cada rodada, o conjunto de dados é embaralhado e dividido em dois subconjuntos: um subconjunto de treinamento (compreendendo 80% do conjunto de dados) e um subconjunto de teste (compreendendo 20% do conjunto de dados). Sendo assim, em vez de usar uma matriz de confusão e apenas as métricas de acurácia para avaliar os resultados do modelo, utilizou-se uma tabela com as métricas de acurácia, precisão, sensibilidade e *F1-score* para todas as classes, calculadas como a média e o desvio padrão das cinco rodadas.

3.3.4 Resultados e discussões

Nesta seção, serão apresentados os resultados obtidos para os três conjuntos de dados: Montalbano, GRIT e IsoGD.

3.3.4.1 Resultados para conjunto de dados Montalbano

O modelo treinado de acordo com a configuração descrita anteriormente atingiu uma acurácia média de 94,58%, quando executado sobre o conjunto de teste. A Tabela 3 ilustra a acurácia obtida por cada gesto e a Figura 21 mostra a matriz de confusão das predições do modelo. Observe-se que, para a maioria dos gestos, o valor de acurácia é maior que 90%, enquanto que para os gestos *fame*, *cheduepalle*, *combinato*, *daccordo* e *tantotempo* a acurácia ultrapassou os 98%. Em apenas alguns poucos casos, como nos gestos *vattene*, *ok* e *noncenepiu*, a acurácia não ultrapassou os 90%. No entanto, mesmo no pior dos casos, (*noncenepiu*), o valor de acurácia foi de 86,63%.

Para demonstrar o impacto do uso do *soft-attention* no desempenho do classificador de gestos, a arquitetura original foi modificada. Assim, dois experimentos foram realizados. Em relação ao primeiro, foram testados diferentes técnicas de fusão de características no lugar do *soft-attention*: soma, média e concatenação. Os resultados desse experimento estão resumidos na Tabela 4, onde é possível notar que o *soft-attention* alcança o melhor resultado, superando outras técnicas em mais de 0,71% de acurácia. Em um segundo experimento, a fim de mostrar a influência de cada CNN em cada classe, removeu-se o *soft-attention*, de maneira que a 4ª camada convolucional de cada *Resnet* fosse diretamente passada às camadas totalmente conectadas. Como resultado, o classificador de gestos que usa apenas a *Resnet 50* alcançou 91,95% de acurácia, enquanto que a *Resnet 101* alcançou 90,70%. A Tabela 5 mostra tais resultados.

Tabela 3 – Resultados dos experimentos utilizando o conjunto de dados Montalbano.

Gesto	Acurácia(%)	Gesto	Acurácia(%)
fame	99.46	*cosatifarei*	94.68
cheduepalle	99.42	*prendere*	94.02
combinato	98.91	*buonissimo*	93.82
daccordo	98.77	*seipazzo*	92.97
tantotempo	98.27	*chevuoi*	92.42
basta	97.24	*vieniqui*	92.31
sonostufo	97.14	*freganiente*	91.76
messidaccordo	96.11	*vattene*	88.76
furbo	96.07	*ok*	87.36
perfetto	95.51	*noncenepiu*	86.63

Fonte: Próprio autor.

Perceba-se que, para os gestos *fame*, *cheduepalle*, *combinato*, *tantotempo* e *messidaccordo*, a *Resnet 50* teve um desempenho pior que a *Resnet 101*. No entanto, em outros casos, por exemplo, *vattene*, *freganiente* e *cosatifarei*, a *Resnet 50* supera a *Resnet 101*. No entanto, para todos os gestos, o *ensemble* (fusão) das duas CNNs dá uma melhor acurácia de classificação, exceto no gesto *basta*, que o *Resnet 50* alcançou 98,6%.

Tabela 4 – Acurácia do conjunto de dados Montalbano para diferentes técnicas de fusão de características.

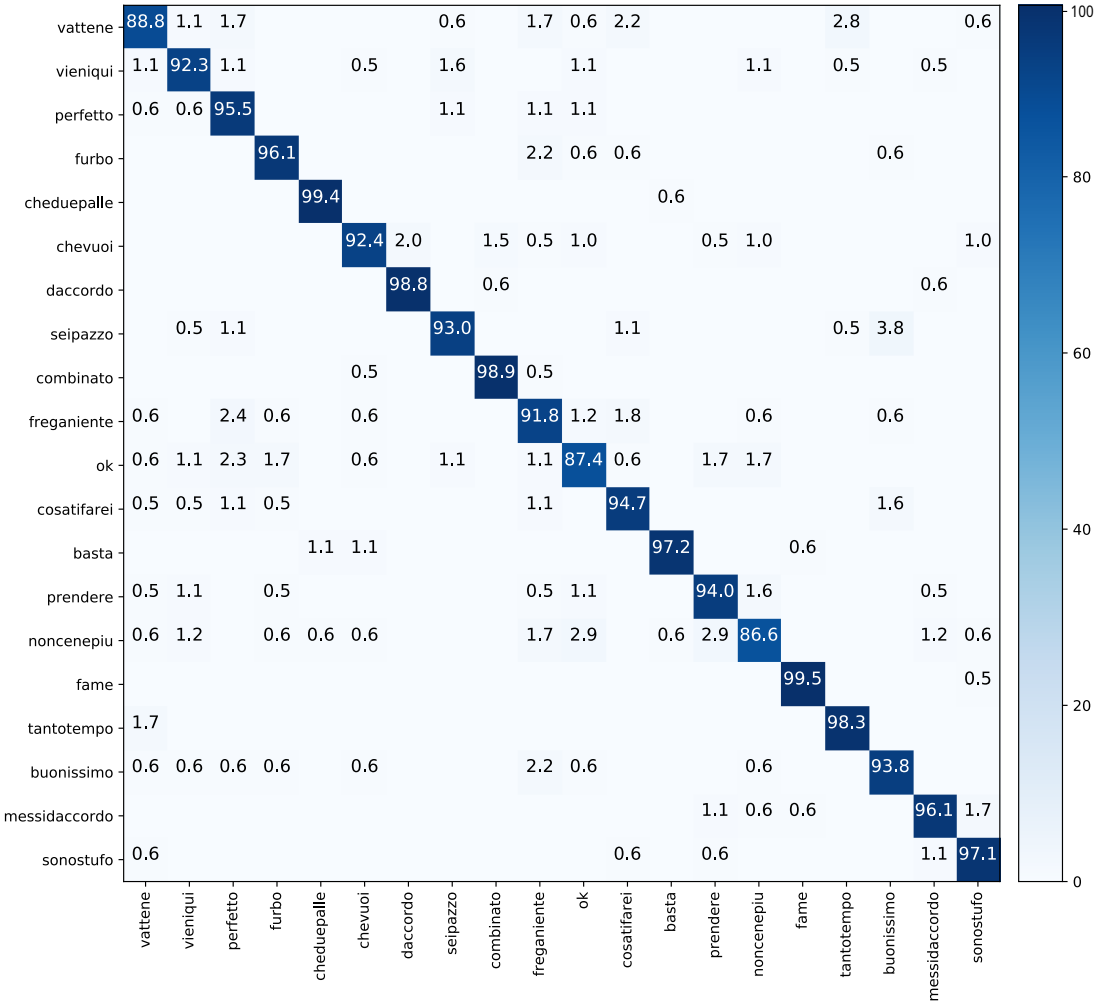
Método de Fusão	Acurácia
Concatenação	93,35%
Soma	93,37%
Média	93,88%
Soft-attention	**94,58%**

Todos os experimentos foram executados de maneira a garantir o determinismo entre eles. Dessa forma, não se faz necessário um teste estatístico sobre os resultados.

Fonte: Próprio autor.

Assim, em relação ao número de classes do problema e ao uso exclusivo de informações RGB para representar e reconhecer gestos dinâmicos, consideram-se os resultados aqui apresentados competitivos com outras abordagens que empregam dados multimodais. Além disso, o método proposto nesta tese traz um dos melhores resultados para a classificação de gestos no conjunto de dados Montalbano, superando outros trabalhos que empregam mais de uma fonte de informação, por exemplo, Joshi et al. (2017), Cao, Zhang e Lu (2015a) e Escobedo e Camara (2015). Uma comparação entre essas diferentes abordagens, incluindo o método aqui proposto, no conjunto de dados Montalbano é mostrada na Tabela 6. Como pode ser visto, esta proposta supera quase todas as outras, ficando levemente atrás

Figura 21 – Matriz de confusão das predições feitas pelo modelo $starRGB_{SoftAtt}$ treinado com o conjunto de dados Moltalbano.



Fonte: Próprio autor.

da proposta presentada por Pigou et al. (2018), que alcançou 97,23% utilizando RGB, profundidade e esqueleto.

3.3.4.2 Resultados para o conjunto de dados GRIT

Os resultados dos cinco experimentos de *hold outs* podem ser vistos na Tabela 7. Como esperado, a proposta apresentada alcançou melhores resultados ao considerar o conjunto de dados GRIT, superando os resultados em Tsironi et al. (2017). A melhoria foi superior a 6% para todas as métricas: acurácia, precisão, sensibilidade e *F1-Score*. Além disso, o melhor resultado obtido aqui foi de 100% de acurácia contra 92,59% alcançado pelos autores (apresentado apenas no trabalho). Esse resultado implica que o *Star RGB* realmente melhora a representação dinâmica de gestos. Assim, o *Star* $RGB_{SoftAtt}$ pode contribuir também para o campo de pesquisa em reconhecimento de gestos dinâmicos para a interação humano-robô.

Tabela 5 – Resultado dos experimentos no conjunto de dados Montalbano, usando cada *Resnet* individualmente e combinando-as através do *soft-attention*.

Gesto	*Resnet 50* (%)	*Resnet 101* (%)	*Ensemble* (%)
fame	96,76	97,84	99,46
cheduepalle	87,86	99,42	99,42
combinato	95,11	98,91	98,91
daccordo	98,16	98,77	98,77
tantotempo	96,53	98,27	98,27
basta	98,90	97,24	97,24
sonostufo	95,43	96,57	97,14
messidaccordo	92,22	96,11	96,11
furbo	94,94	91,57	96,07
perfetto	92,13	91,57	95,51
cosatifarei	92,02	81,91	94,68
prendere	91,85	92,93	94,02
buonissimo	91,57	91,01	93,82
seipazzo	92,97	90,81	92,97
chevuoi	91,41	91,92	92,42
vieniqui	87,91	84,07	92,31
freganiente	91,76	78,82	91,76
vattene	88,76	72,47	88,76
ok	83,33	82,76	87,36
noncenepiu	79,07	81,40	86,63

Fonte: Próprio autor.

Tabela 6 – Comparando resultados entre trabalhos que objetivaram reconhecer os gestos do conjunto de dados Montalbano utilizando os vídeos segmentados.

Trabalho	Tipo de dados	Resultado
Pigou et al. (2018)	Esqueleto e RGB-D	97,23%
Liu et al. (2019)	Esqueleto	93,80%
Efthimiou et al. (2016)	Áudio e RGB	93,00%
Escobedo e Camara (2015)	Esqueleto e RGB-D	88,38%
Wu et al. (2016)	Profundidade	82,62%
Fernando et al. (2015)	Esqueleto e Áudio	80,29%
Cao, Zhang e Lu (2015a)	RGB	60,07%
Star RGB$_{SoftAtt}$	**RGB**	**94,58%**

Fonte: Próprio autor.

3.3.4.3 Resultados para o conjunto de dados IsoGD

O modelo foi treinado de acordo com a configuração descrita anteriormente e atingiu uma acurácia média de 52,18% quando executado no conjunto de dados de teste.

Embora o valor da precisão não seja tão alto quanto no conjunto de dados Montalbano, levando em consideração o elevado número de classes do problema, o número de

Tabela 7 – Comparação dos resultados obtidos pela proposta contra os obtidos pelos autores em Tsironi et al. (2017) usando o conjunto de dados GRIT.

Métrica	Tsironi et al. (2017)	**Star RGB**
Acurácia	$91.67\% \pm 1.13\%$	$98.35\% \pm 0.90\%$
Precisão	$92.35\% \pm 0.98\%$	$98.55\% \pm 0.80\%$
Sensibilidade	$91.90\% \pm 1.05\%$	$98.33\% \pm 0.95\%$
F1-score	$92.13\% \pm 1.00\%$	$98.34\% \pm 0.93\%$

Fonte: Próprio autor.

diferentes categorias de gestos e o uso exclusivo de informações RGB para representar e reconhecer gestos dinâmicos, pode-se considerar que os resultados aqui apresentados são competitivos quando comparados aos resultados de outras abordagens que empregam dados multimodais. Como pode ser visto na Tabela 8, a solução proposta alcançou boas posições entre outras abordagens que usam apenas imagens RGB para resolver o problema.

Tabela 8 – Comparação dos resultados obtidos para o conjunto de dados de teste do IsoGD entre os trabalhos que publicaram seus resultados utilizando apenas RGB.

Trabalho	Tipo de dados	Resultado
Narayana, Beveridge e Draper (2018)	RGB	41.27%
Zhu et al. (2017)	RGB	43.88%
Duan et al. (2018)	RGB	45.65%
Narayana, Beveridge e Draper (2018)	RGB Flow	50.96%
$\boldsymbol{Star\ RGB_{SoftAtt}}$	**RGB**	**52.18%**
Duan et al. (2018)	RGB Flow	58.38%
Lin et al. (2018)	RGB	59.03%

Fonte: Próprio autor.

3.3.5 Experimento em tempo real

Para validar a proposta em um ambiente real, foi realizado ainda um experimento de reconhecimento dinâmico de gestos em tempo real. Para esse fim, treinou-se o modelo usando todo o conjunto de dados GRIT. Observe-se que, para ter uma avaliação justa de seu desempenho em tempo real, o modelo deve receber apenas clipes de vídeo que contenham gestos e tentar reconhecê-los.

Portanto, primeiro, foi implementada uma abordagem de segmentação simples para obter clipes de vídeo advindos de um *streaming* em tempo real, compostos de *frames* consecutivos que podem conter um gesto (uma segmentação mais robusta será proposta no Capítulo 5). Tal segmentação foi implementada de maneira que, para cada dois *frames* consecutivos, foi calculada a média de todos os píxeis da imagem resultante da diferença entre eles e comparada a um valor de limiar. Quando o limiar era excedido, considerava-se

Tabela 9 – Tempos médios de resposta do sistema calculados para todos os gestos realizados. Observe-se que apenas o modelo pode estar em execução na CPU ou GPU. O processo de cálculo do *Star RGB* e os serviços de comunicação sempre são executados na CPU. Todos os valores estão arredondados para o maior inteiro mais próximo.

Dispositivo	Tempo médio (ms)			
	Modelo	*Star RGB*	Comunicação	Resposta
CPU	110	1	11	122
GPU	29	-	-	41

Fonte: Próprio autor.

que um gesto tinha sido iniciado. Em seguida, os *frames* eram armazenados em um *buffer* até que a média caísse para um valor abaixo do mesmo limiar, indicando que o gesto havia finalizado. Finalmente, o clipe formado pelos *frames* em *buffer* era passado para para o processo que gerava o *Star RGB*.

Em termos de implementação, foi utilizada uma Chamada de Procedimento Remoto (RPC - do inglês *Remote Procedure Call*) para passar a imagem resultante da aplicação do *Star RGB* sobre a sequência de *frames* para o método de predição do modelo treinado, que estava sendo executado em um servidor remoto. Um diagrama de todo o experimento é mostrado na Figura 22.

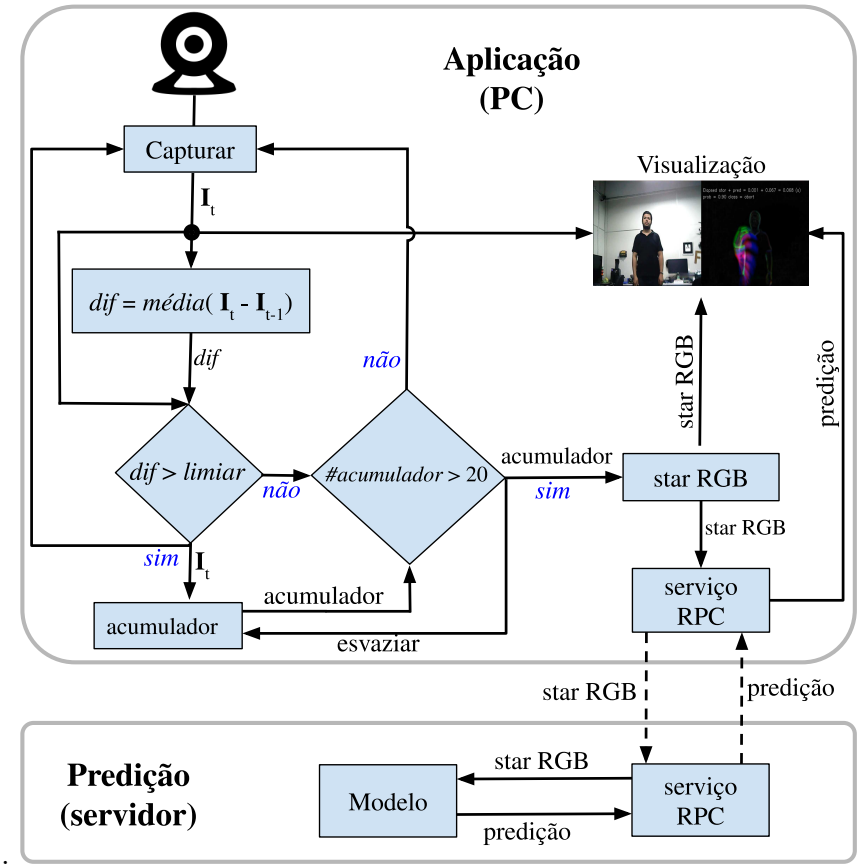
Para os experimentos, foram usados: (*i*) uma *webcam* de baixa resolução (1,3 MP); (*ii*) a biblioteca Opencv 4.3[4] para processar as imagens; (*iii*) a biblioteca gRPC[5] para criar e gerenciar o serviço, que chama o método de predição do modelo; (*iv*) a biblioteca ProtoBuf[6] para serializar os dados enviados ao servidor remoto (imagem RGB e o vetor com as predições); (*v*) um computador com 16 GB de memória e um processador Intel Core $i5-3570$, 3.40 GHz para capturar e processar as imagens; e (*vi*) um servidor com 96 GB de memória, processador Intel Core $i7-7700$, 3.60 GHz e uma Nvidia Titan V de 12 GB.

Durante os experimentos, uma pessoa executava várias vezes e, em ordem aleatória, os gestos pertencentes às classes do conjunto de dados GRIT. O tempo médio de resposta do sistema foi de $41ms$ quando o modelo estava sendo executado na GPU e $122ms$ quando estava sendo executado na CPU. A Tabela 9 mostra com mais detalhes os tempos de resposta do sistema.

Esses resultados mostram a eficácia da proposta, mesmo quando ela é usada em um ambiente diferente daquele em que os dados de treinamento foram capturados. Outro aspecto importante é a possibilidade de usar a abordagem proposta em aplicativos em tempo real, pois o tempo de resposta pode atender aos requisitos de tempo para diferentes

[4] <https://github.com/itseez/opencv>
[5] <https://grpc.github.io/grpc/python/>
[6] <https://developers.google.com/protocol-buffers/docs/encoding>

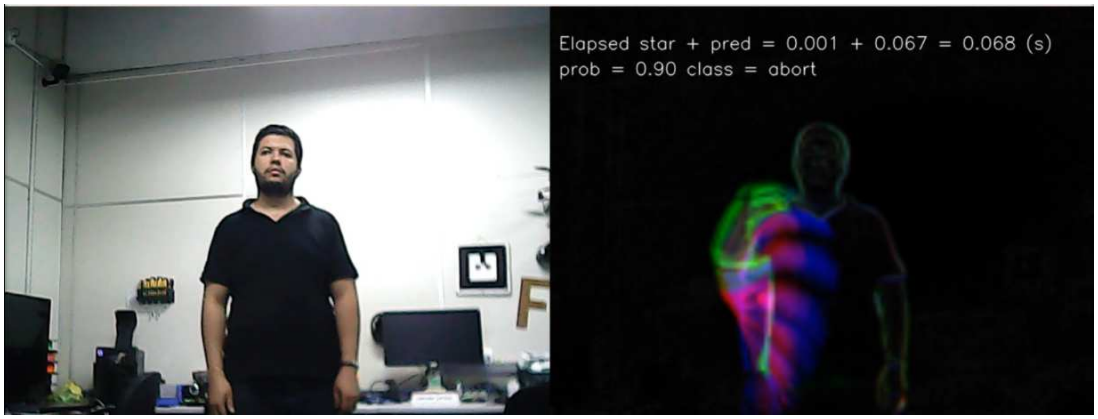
Figura 22 – Diagrama do experimento em tempo real



Fonte: Próprio autor.

aplicativos de interação humano-máquina, mesmo quando o modelo está sendo executado exclusivamente na CPU. A Figura 23 mostra uma representação do experimento em tempo real [7].

Figura 23 – Exemplo de um gesto "abort" no experimento em tempo real.



Fonte: Próprio autor.

[7] Versão completa do experimento em tempo real: <https://youtu.be/Nb6bBZr0mJY>

3.4 Proposta 2: *Star iRGB* - uma representação iterativa

Mesmo com os resultados significativos alcançados pelo *Star* $RGB_{SoftAtt}$, o processo do *Star RGB* ainda necessita receber o vídeo completo contendo o gesto a ser reconhecido. Isso dificulta a sua utilização em modelos de natureza sequencial, os quais esperam receber as observações uma a uma. Sendo assim, propõe-se aqui uma representação iterativa de movimento que possa fornecer uma imagem *Star RGB* para cada *frame* do vídeo. Com isso, além de poder ser utilizada em conjunto com modelos sequenciais para o reconhecimento de gestos *offline*, essa junção possibilitará ainda o reconhecimento *online* ou mesmo a antecipação de gestos.

Nesse sentido, seja uma janela de tamanho N, uma sequência de *frames* $\mathcal{I} = \{\mathbf{I}_1, \mathbf{I}_2, \mathbf{I}_3, ..., \mathbf{I}_S\}$ com $S \geq N+1$, um fator de amortecimento $\alpha \in [0,1]$, seu complementar $\beta = 1 - \alpha$, e $\mathbf{D}_s$ o resultado da Equação (3.4) aplicada sobre os *frames* consecutivos $\mathbf{I}_s$ e $\mathbf{I}_{s-1}$, com $1 \leq s \leq S$. Dessa forma, para $s \geq N$, pode-se calcular uma versão amortecida e iterativa da *representação star-cosseno*, $\mathbf{M}_s$, utilizando uma média móvel exponencial, como apresentado na Equação(3.6).

$$
\begin{aligned}
\mathbf{M}_s &= \alpha\mathbf{D}_s + \beta\mathbf{M}_{s-1} \\
&= \alpha\mathbf{D}_s + \beta(\alpha\mathbf{D}_{s-1} + \beta\mathbf{M}_{s-2}) \\
&= \alpha\mathbf{D}_s + \beta(\alpha\mathbf{D}_{s-1} + \beta(\alpha\mathbf{D}_{s-2} + \beta\mathbf{M}_{s-3})) \\
&\cdots \\
&= \alpha \sum_{k=0}^{N-1} \beta^k \mathbf{D}_{s-k} \\
&= \alpha(\beta^0\mathbf{D}_s + \beta^1\mathbf{D}_{s-1} + \cdots + \beta^{N-1}\mathbf{D}_{s-(N-1)}) \\
&= \alpha(\mathbf{D}_s + \beta\mathbf{D}_{s-1} + \cdots + \beta^{N-1}\mathbf{D}_{s-(N-1)}) \\
\frac{1}{\alpha}\mathbf{M}_s &= \mathbf{D}_s + \beta\mathbf{D}_{s-1} + \cdots + \beta^{N-2}\mathbf{D}_{s-N+2} + \beta^{N-1}\mathbf{D}_{s-N+1} \\
\frac{1}{\alpha}\mathbf{M}_{s-1} &= \mathbf{D}_{s-1} + \beta\mathbf{D}_{s-2} + \cdots + \beta^{N-2}\mathbf{D}_{s-N+1} + \beta^{N-1}\mathbf{D}_{s-N} \\
\frac{\beta}{\alpha}\mathbf{M}_{s-1} &= \beta\mathbf{D}_{s-1} + \beta^2\mathbf{D}_{s-2} + \cdots + \beta^{N-1}\mathbf{D}_{s-N+1} + \beta^N\mathbf{D}_{s-N} \\
\frac{1}{\alpha}\mathbf{M}_s - \frac{\beta}{\alpha}\mathbf{M}_{s-1} &= \mathbf{D}_s - \beta^N\mathbf{D}_{s-N} \\
\mathbf{M}_s - \beta\mathbf{M}_{s-1} &= \alpha(\mathbf{D}_s - \beta^N\mathbf{D}_{s-N}) \\
\mathbf{M}_s &= \alpha(\mathbf{D}_s - \beta^N\mathbf{D}_{s-N}) + \beta\mathbf{M}_{s-1}.
\end{aligned}
\tag{3.6}
$$

Sendo assim, considerando apenas uma mudança no tamanho da sequência de *frames* $\mathcal{I} = \{\mathbf{I}_1, \mathbf{I}_2, \mathbf{I}_3, ..., \mathbf{I}_S\}$ para $S \geq vN + 1$, uma imagem de v canais que representa a

informação de movimento da sequência $\mathcal{I}$ pode ser obtida iterativamente como se segue:

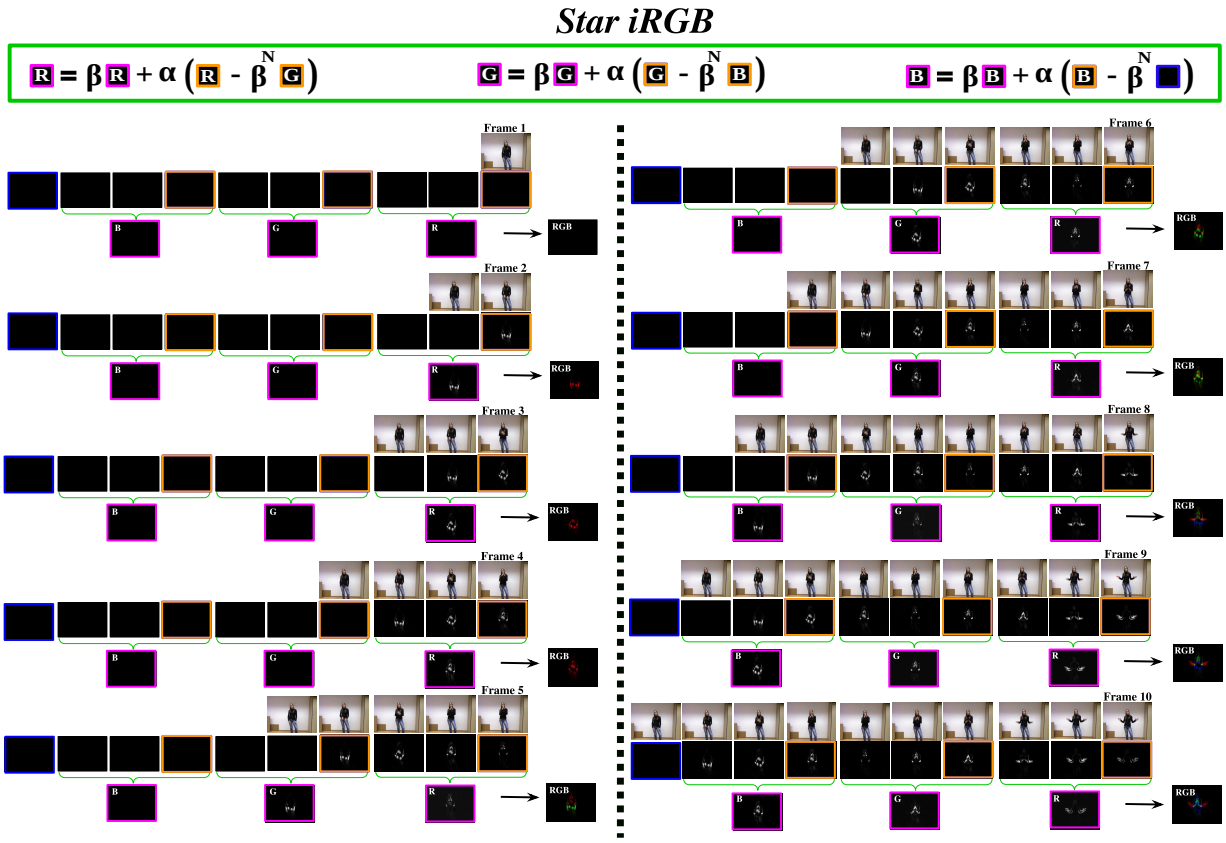
$$\begin{aligned}
\mathbf{M}_s^{(c_1)} &= \alpha(\mathbf{D}_s - \beta^N \mathbf{D}_{s-N}) + \beta \mathbf{M}_{s-1}^{(c_1)} \\
\mathbf{M}_s^{(c_2)} &= \alpha(\mathbf{D}_{s-N} - \beta^N \mathbf{D}_{s-2N}) + \beta \mathbf{M}_{s-1}^{(c_2)} \\
\mathbf{M}_s^{(c_3)} &= \alpha(\mathbf{D}_{s-2N} - \beta^N \mathbf{D}_{s-3N}) + \beta \mathbf{M}_{s-1}^{(c_3)} \\
&\dots \\
\mathbf{M}_s^{(c_v)} &= \alpha(\mathbf{D}_{s-(v-1)N} - \beta^N \mathbf{D}_{s-vN}) + \beta \mathbf{M}_{s-1}^{(c_v)} \\
\mathbf{M}_s^{(v)} &= \left[\mathbf{M}_s^{(c_1)}, \mathbf{M}_s^{(c_2)}, \mathbf{M}_s^{(c_3)}, \dots, \mathbf{M}_s^{(c_v)}\right] \\
\mathbf{M}_s^{(norm)} &= \frac{\mathbf{M}_s^{(v)} - min(\mathbf{M}_s^{(v)})}{max(\mathbf{M}_s^{(v)}) - min(\mathbf{M}_s^{(v)})}.
\end{aligned} \tag{3.7}$$

onde $\mathbf{M}_s^{(c_1)}, \mathbf{M}_s^{(c_2)}, \mathbf{M}_s^{(c_3)}, \dots, \mathbf{M}_s^{(c_v)}$ representam os v canais que, após concatenados pelo operador $[\bullet]$, formam a imagem $\mathbf{M}_s^{(v)}$ obtida no instante de tempo correspondente ao n-ésimo *frame* da sequência; $\mathbf{M}_s^{(norm)}$ é a forma normalizada de $\mathbf{M}_s^{(v)}$; e min e max são operadores que retornam respectivamente o menor e o maior valor dentre todos os píxeis de cada um dos v canais da imagem $\mathbf{M}_s^{(v)}$.

Com essa forma iterativa, após $vN + 1$ observações, é possível obter uma representação de movimento atualizada a cada novo *frame*, o que permite iterar sobre sequências (vídeos) de tamanho variável. O principal problema aqui é a necessidade de esperar ao menos $vN + 1$ *frames* para que seja possível obter a primeira representação. Ou mesmo, condicionar o seu uso a vídeos com, pelo menos, $vN + 1$ *frames*. Nesse sentido, propõe-se adicionar vN *frames* contendo apenas valores iguais a 0 antes da primeira observação, x_1. Assim, já no segundo *frame* da sequência, poder-se-á obter uma representação de movimento. Com isso, será possível iterar sobre toda a sequência, fornecendo uma representação de movimento para cada *frame* de entrada. Dessa forma, é importante reformular o problema: seja uma sequência de *frames* $\mathcal{I} = (\mathbf{I}_{(-vN)}, \mathbf{I}_{(-vN+1)}, ..., \mathbf{I}_0, \mathbf{I}_1, \mathbf{I}_2 ..., \mathbf{I}_S)$, onde os *frames* com índices não estritamente positivos são imagens cujos elementos são iguais a 0. Ao ser dada como entrada para a Equação (3.7), a sequência $\mathcal{I}$ fornecerá S representações de movimento, onde a primeira é uma representação vazia, pois a equação de similaridade (3.5) necessita de pelo menos duas imagens.

Neste trabalho será utilizado um número de canais $v = 3$, fazendo com que a imagem fornecida pela Equação (3.7) seja uma representação iterativa e amortecida do *Star RGB*. Essa nova representação possibilita que modelos temporais sejam utilizados para reconhecer iterativamente os gestos representados por cada imagem *Star RGB* obtida a cada instante de tempo t, $1 \leq t \leq S$. A ela será dada o nome de *Star iRGB*, e um exemplo do seu processo de criação pode ser visualizado na Figura 24.

Figura 24 – Processo completo do *Star iRGB* sobre um clipe de vídeo com 10 *frames* que contêm parte de um gesto da classe *basta* do conjunto de dados Montalbano. Na execução foram utilizados $\alpha = 0.6$ e $N = 3$. Os 10 *frames* de entrada geram 10 imagens do tipo *Star RGB* seguindo a Equação (3.7), sendo que, como definido, a primeira imagem resultante é sempre vazia.



Fonte: Próprio autor.

Note-se uma característica peculiar da *Star iRGB*: o movimento presente nas N primeiras representações estará contido apenas no primeiro canal da imagem, o canal R, enquanto que os outros canais estarão vazios (com todos os valores iguais a 0). Após o *frame* N, a cada novo *frame*, o canal G começa a armazenar a informação de movimento contida em R, sendo que o canal B continua vazio até o *frame* $2N$. Só partir daí que ele iniciará o armazenamento da informação de movimento contida em G. Isso permite que o movimento seja passado gradualmente entre todos os canais, de maneira a permitir que o modelo determine, a cada novo *frame* recebido, qual o canal da imagem *Star iRGB* oferece-o a informação mais significativa para a solução do problema. Aqui, diferentemente do *Star RGB*, não se tem mais o canal G como sendo o possível armazenador da informação de *stroke* do gesto. Esse papel pode ser assumido por qualquer um dos três canais. Inclusive, a depender do tamanho N da janela, pode ser que tal informação esteja distribuídas em mais de um canal. O que pode servir como um mecanismo de regularização para o modelo.

3.4.1 Modelo de Classificação

Para o modelo de classificação, ao invés de utilizar duas CNNs fundidas pelo *soft-attention*, propõe-se utilizar uma CNN como extrator de características e uma rede neural recorrente para melhorar a captura da informação temporal e classificar o gesto presente em cada *frame* do vídeo, representado por uma imagens do tipo *Star iRGB*.

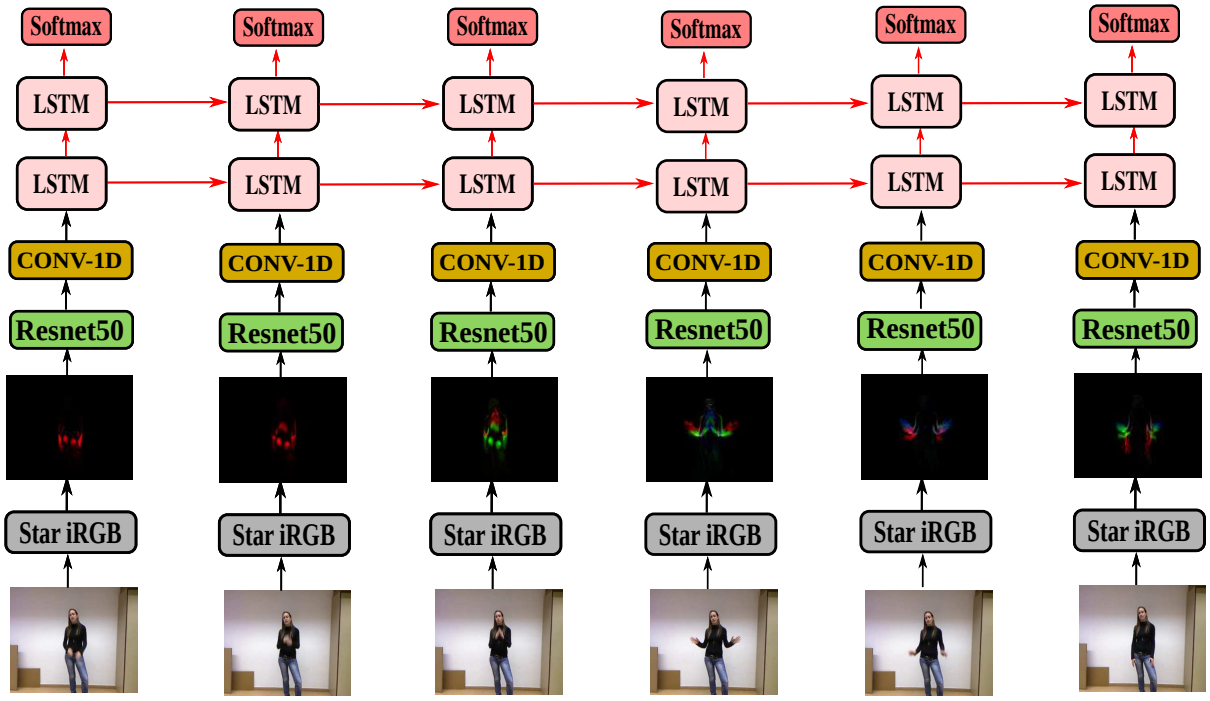
Uma vez que, para o reconhecimento de um gesto, o extrator será utilizado tantas vezes quanto o número de *frames* do vídeo, sendo ele o componente do modelo que possivelmente demandará mais tempo de processamento, é importante ele seja rápido e não ofereça grandes prejuízos aos resultados. Nesse sentido, decidiu-se utilizar a *Resnet 50*. A escolha de tal CNN se deu pelo seu desempenho na tarefa de classificação de gestos com o modelo *Star* $RGB_{SoftAtt}$ quando comparado com a sua versão mais profunda, a *Resnet 101*. Esta última tem mais que o dobro do seu número de camadas, e mesmo assim, os resultados foram semelhantes aos dela (Tabela 4). Dessa forma, pelos mesmos motivos, não será utilizada a arquitetura completa do *Star* $RGB_{SoftAtt}$.

Da *Resnet 50*, utiliza-se apenas a sua parte convolucional, seguida de um operador GAP (*Global Average Pooling*) que fornece um vetor de saída $\mathbf{v}_{gap} \in \mathbb{R}^{2048\times1}$. Sobre esse vetor, aplica-se uma operação de convolução com um *kernel* treinável de dimensão $\mathbb{R}^{3\times1}$ e *stride 2*. Essa última operação, além de diminuir a dimensionalidade das características extraídas pela CNN ($\mathbb{R}^{2048\times1} \rightarrow \mathbb{R}^{1023\times1}$), ainda tem a função de filtrar possíveis ruídos que possam prejudicar a representação temporal da RNN. Para o modelo de RNN, optou-se por utilizar uma LSTM, devido à sua capacidade de capturar longas dependências entre os elementos de uma sequência. A arquitetura da LSTM tem duas camadas, cada uma com estados ocultos de 512 neurônios, seguidas de um classificador do tipo *softmax*. Dessa forma, para cada imagem de entrada, é possível estimar uma probabilidade dela pertencer a uma das possíveis classes de gestos. Para fins de representação, a arquitetura proposta será chamada de *Star* $iRGB_{lstm}$.

A Figura 25 ilustra o modelo proposto. Nele, a primeira camada da LSTM recebe como entrada no instante t: (i) o vetor $\mathbf{v}^{(t)} \in \mathbb{R}^{1023\times1}$ resultante do operador de convolução unidimensional aplicado sobre o vetor $\mathbf{v}_{gap} \in \mathbb{R}^{2048\times1}$; (ii) o estado oculto $\mathbf{h}_1^{(t)} \in \mathbb{R}^{512\times1}$; e (iii) a célula $\mathbf{c}_1^{(t)} \in \mathbb{R}^{512\times1}$. A segunda camada recebe como entrada: (i) o estado oculto $\mathbf{h}_1^{(t)}$ resultante da primeira camada; (ii) o estado oculto $\mathbf{h}_2^{(t)} \in \mathbb{R}^{512\times1}$; e (iii) a célula $\mathbf{c}_2^{(t)} \in \mathbb{R}^{512\times1}$. Em seguida, uma camada totalmente conectada recebe como entrada o estado $\mathbf{h}_2^{(t)}$, aplica sobre ele uma transformação linear usando uma matriz $\mathbf{P} \in \mathbb{R}^{512\times C}$ (onde C é o número de classes de gestos) e o normaliza usando a função *softmax*. Dessa forma, a saída do modelo é um vetor $\hat{\mathbf{s}}^{(t)}$ composto por C valores normalizados entre 0 e 1, onde cada valor está relacionado a um possível gesto.

Como classificador, poder-se-ia utilizar uma RNN bidirecional (SCHUSTER; PA-

Figura 25 – Representação completa do *Star iRGB*$_{lstm}$ proposto.



Fonte: Próprio autor.

LIWAL, 1997), ou mesmo arquiteturas sequenciais mais sofisticadas, como aquelas derivadas da arquitetura *Transformer* (VASWANI et al., 2017). Porém, como aqui, um gesto ainda é representado na forma de um clipe de vídeo e um dos objetivos deste trabalho é poder utilizar as principais propostas de forma *online*, tais arquiteturas não são indicadas. Elas necessitam receber uma sequência de observações de tamanho fixo para que possam realizar uma predição. Assim, para poderem ser usadas, necessitariam de um processo de *padding*, como o apresentado na Seção 2.7. Desse modo, mesmo utilizando uma janela deslizante de tamanho M com sobreposição de $M-1$ *frames*, a primeira predição só seria realizada a partir do *frame* M. O que inviabilizaria a utilização da proposta em sistemas que demandam uma predição a cada novo *frame* observado (como as abordagens de antecipação de reconhecimento *online*, tratadas nos próximos capítulos). Tal condição poderia ser satisfeita caso a primeira observação fosse precedida de $M-1$ observações nulas ou que ela fosse repetida $M-1$ vezes. Contudo, do ponto de vista de representação, seria o mesmo que tentar reproduzir o que já está sendo representado no *Star iRGB*. Sendo assim, tais arquiteturas não são indicadas neste trabalho quando o movimento é representado pelo *Star iRGB*. Logo, de forma geral, neste e nos próximos capítulos, as arquiteturas de classificação utilizadas nas principais propostas serão baseadas apenas em RNNs unidirecionais.

Em relação ao *Star RGB*$_{SoftAtt}$, o *Star iRGB*$_{lstm}$ demanda mais processamento, uma vez que ele é executado tantas vezes quanto o número de *frames* que o vídeo possui;

enquanto o outro é executado apenas uma vez, independentemente do número de *frames*. No entanto, além de possibilitar a utilização do *Star iRGB* para reconhecimento e (ou) antecipação de gestos em tempo real, essa nova proposta provavelmente melhorará os resultados do reconhecimento de gestos *off-line*, uma vez que a representação temporal possivelmente será enfatizada pela LSTM.

Um aspecto a ser salientado é que, no processo iterativo apresentado por Tsironi et al. (2017), a LSTM recebia como entrada as características extraídas de uma imagem em tons de cinza resultante da aplicação da Equação (3.1) (*representação star*) sobre duas imagens consecutivas. Sendo assim, a LSTM era a principal encarregada por correlacionar temporalmente o movimento representado em cada entrada. Por outro lado, no *Star* $iRGB_{lstm}$, além da LSTM realizar a mesma tarefa, a entrada possui a informação de movimento presente nos últimos *frames*, o que possivelmente irá melhorar a sua representação temporal. Por exemplo, considerando $N = 3$, cada canal da imagem *Star iRGB* possui informação de movimento de quatro *frames* consecutivos. Em outras palavras, quando a i-ésima imagem *Star iRGB* da sequência alimentar o *Star* $iRGB_{lstm}$, a LSTM receberá como entrada a informação de movimento dos nove *frames* anteriores, o que, supostamente, contribuirá para a sua memória de curto prazo. Enquanto isso, a sua memória de longo prazo poderá ser fortalecida por meio da recorrência. Dessa forma, o modelo poderá tornar-se muito mais robusto para o reconhecimento de gestos, tanto de curta quanto de longa duração. Outro aspecto importante é que ao iniciar a sequência com *frames* vazios, a esparsidade das primeiras representações do *Star iRGB* fortalecerá a informação de que a sequência está em seu início.

3.4.2 Experimentos

Para mostrar a efetividade da proposta, foi utilizado novamente o conjunto de dados Montalbano. Assim, cada *frame* de cada sequência contendo um gesto foi passado para a Equação (3.7), que forneceu uma representação *Star iRGB*, a qual alimentou o modelo de aprendizagem. Dessa forma, a cada novo *frame* extraído da sequência, o *Star* $iRGB_{lstm}$ estimou a probabilidade dele pertencer a uma das 20 classes de gestos do conjunto de dados. O reconhecimento se deu pela classe cuja probabilidade estimada no último *frame* da sequência era a máxima.

O conjunto de dados Montalbano possui quase 2 milhões de imagens quando somadas as de treino, teste e validação. Os vídeos possuem uma taxa de exibição de 20 FPS. Assim, para poder diminuir o tempo gasto no treinamento, todo o conjunto de dados foi reduzido pela metade por meio de um processo de reamostragem que transformou a taxa de exibição dos vídeos para 10 FPS. Com essa taxa, ainda é possível assistir aos vídeos sem grandes perdas de continuidade dos movimentos. Sendo assim, a reamostragem não afetará

de maneira negativa o reconhecimento dos gestos. Porém, diminuirá consideravelmente o tempo de treinamento dos modelos.

O processo de avaliação dos resultados será o mesmo utilizado anteriormente: uso da acurácia média de reconhecimento. Os resultados também serão apresentados na forma de tabelas e de uma matriz de confusão.

Por ser um problema de natureza sequencial, um possível modelo clássico a ser utilizado como *baseline* é o HMM. No entanto, como o extrator proposto é baseado em uma rede neural convolucional, a CNN e o HMM deveriam ser treinados separadamente, o que poderia retardar consideravelmente a obtenção dos resultados. Assim, como modelo básico de comparação, a LSTM será substituída por uma RNN padrão (*Star* $iRGB_{rnn}$) com o mesmo número de neurônios. Além disso, para justificar o uso da LSTM, será ainda utilizada como *baseline* uma RNN do tipo GRU (*Star* $iRGB_{gru}$) também com o mesmo número de neurônios. Todos os resultados aqui obtidos serão comparados com os do *Star* $RGB_{SoftAtt}$.

Mesmo que o modelo proposto também utilize uma LSTM na classificação dos gestos, não se faz necessário testá-lo sobre o conjunto de dados GRIT (TSIRONI et al., 2017). Recorde-se que o *Star* $RGB_{SoftAtt}$ já obteve melhores resultados quando treinado e testado em tal conjunto de dados. Considere-se ainda que os gestos do GRIT são mais simples bem mais separáveis que o Montalbano. Dessa maneira, caso o *Star* $iRGB_{lstm}$ ultrapasse os resultados alcançados pelo *Star* $RGB_{SoftAtt}$ sobre o Montalbano, a superioridade do *Star* $iRGB_{lstm}$ sobre a proposta dos autores apresentada para o GRIT estará consequentemente comprovada. Caso contrário, experimentos nesse sentido deverão ser realizados.

Um processo de otimização bayesiana, implementado utilizando a biblioteca *hyperopt*, foi utilizado para encontrar os melhores hiperparâmetros de cada modelo. A Tabela 10 apresenta o conjunto de hiperparâmetros que alcançaram os melhores resultados sobre o conjunto de validação.

O treinamento destes modelos demandam mais tempo do que para o *Star* $RGB_{SoftAtt}$. Assim, os experimentos utilizaram um outro servidor de GPUs com as seguintes configurações:

- Sistema operacional Linux Ubuntu Server, distribuição 18.04;
- 2 processadores Intel(R) Xeon(R) Silver 4214 CPU @ $2.20GHz$ com 12 núcleos físicos cada;
- 64 GB de RAM;
- $240GB$ de unidade de armazenamento (SSD);
- 4 placas gráficas (3 Nvidia Titan V e 1 Titan XP), com $12GB$ de memória cada.

Mesmo utilizando 4 GPUs, cada treinamento com 100 épocas durou em média 10

Tabela 10 – Lista dos hiperparâmetros utilizados para o treinamento dos modelos e geração do *Star iRGB*. Para taxa de aprendizado são apresentados [TA1, TA2] correspondentes às taxas de aprendizado utilizadas para treinar o extrator e o classificador, respectivamente.

Hiperparâmetros	Modelos		
	Star $iRGB_{rnn}$	Star $iRGB_{gru}$	Star $iRGB_{lstm}$
Geração do *Star iRGB*			
Termo de amortização (α)	0,6	0,6	0,6
Tamanho da janela (N)	5	5	5
Treinamento			
Tamanho do *batch*	96	96	96
Truncamento da sequência	32	32	32
Tamanho da sequência	16	16	16
Quantidade máxima de épocas	100	100	100
Taxa de aprendizado (TA)	[5e-4, 1e-3]	[1e-4, 5e-3]	[1e-4, 1e-3]
Decaimento da TA (por época)	1%	1%	1%
Taxa de decaimento dos pesos	1e-5	1e-5	1e-5
Truncamento de gradiente (Norma)	5,0	5,0	5,0
Dropout	0,15	0,15	0,15
Otimizador	Adam	Adam	Adam

Fonte: Próprio autor.

horas, sendo que o processo completo de busca pelos melhores hiperparâmetros, para os três modelos, durou 20 dias.

As configurações dos experimentos foram as mesmas utilizadas no *Star* $RGB_{SoftAtt}$, exceto o processo de *data augmentation*. Aqui, a entrada do modelo é uma sequência e não uma imagem apenas. Assim, o processo de *data augmentation* utilizado no treinamento do *Star* $RGB_{SoftAtt}$ foi adaptado de maneira a aplicar todas as transformações igualmente em todas as imagens da sequência. O avaliação dos resultados sobre o conjunto de teste também foi realizado uma única vez, utilizando a versão de cada modelo que obteve o melhor resultado sobre o conjunto de validação.

3.4.3 Resultados e discussões

Após o treinamento utilizando o conjunto de dados Montalbano, o modelo proposto *Star* $iRGB_{lstm}$ obteve uma acurácia média de reconhecimento na última observação de 94,96%. Esse valor é 0.4% maior que o obtido pelo *Star* $RGB_{SoftAtt}$ sobre este mesmo conjunto de dados. A Tabela 11 apresenta os resultados dos três modelos de *baselines* e do modelo proposto. Perceba-se que o modelo proposto alcançou uma acurácia média entre classes maior que a do *Star* $RGB_{SoftAtt}$ e dos dois *baselines* (*Star* $iRGB_{rnn}$ e *Star* $iRGB_{gru}$): 94,58%, 91,11% e 94,73%, respectivamente. Como já era esperado, por ser um modelo mais simples, pois usa uma RNN padrão, o *Star* $iRGB_{rnn}$ alcançou o pior resultado

dentre os quatro modelos, 91,11%. Em suma, o modelo proposto, *Star* $iRGB_{lstm}$, é, até aqui, um dos melhores resultados sobre o conjunto de dados Montalbano, utilizando os gestos já segmentados e apenas imagens RGB.

Tabela 11 – Acurácia obtida pelo modelo proposto e os três modelos de *baseline* sobre o conjunto de dados Montalbano.

Modelo	Acurácia média
Star $RGB_{SoftAtt}$	94,58%
Star $iRGB_{rnn}$	91,11%
Star $iRGB_{gru}$	94,73%
Star $\boldsymbol{iRGB_{lstm}}$	**94,96%**

Fonte: Próprio autor.

Para uma comparação mais justa dos resultados, a Tabela 12 apresenta o valores de acurácia obtidos pelo *Star* $iRGB_{lstm}$ e pelo *Star* $RGB_{SoftAtt}$ para cada classe de gestos, bem como a diferença entre eles. Como pode ser visto, tomando como base o modelo proposto, as classes *noncenepiu*, *cosatifarei*, *fame*, *tantotempo* e *messidaccordo* tiveram uma leve piora na acurácia, −1,16%, −0,53%, −1,08%, −0,55%, respectivamente. As diferenças mais significativas foram apenas nas classes *perfetto* (−2.25%) e *furbo* (−6,74%). Por outro lado, quase a totalidade das classes alcançaram mais de 90%, inclusive as classes *ok* e *vatenne* que tiveram um dos piores resultados com o *Star* $RGB_{SoftAtt}$, mas conseguiram uma melhora significativa com o *Star* $iRGB_{lstm}$: 4,02% e 3,75%, respectivamente. No geral, houve mais melhoras que pioras nos resultados, de maneira que, na média, o *Star* $iRGB_{lstm}$ se saiu melhor. Na Figura 26 pode ser vista a matriz de confusão com todos os resultados obtidos pelo modelo *Star* $iRGB_{lstm}$.

Para poder avaliar o tempo de resposta da proposta, um experimento semelhante ao apresentado na Seção 3.3.5 foi realizado. Porém, nele, ao invés de capturar imagens com uma câmera, elas foram extraídas sequencialmente dos clipes de vídeo do conjunto de teste do Montalbano. A cada nova imagem extraída, uma representação *Star iRGB* era gerada por meio da Equação (3.7). Assim, a representação era então enviada para um servidor onde o modelo *Star* $iRGB_{lstm}$, previamente treinado, realizava a predição. A Tabela 13 traz um resumo dos tempos médios de resposta do sistema.

Como pode ser visto, para cada *frame* do vídeo, o tempo de resposta do *Star* $iRGB_{lstm}$ foi em média de $22.82ms$, quando a classificação era realizada em GPU, e $65ms$, quando a mesma era realizada em CPU. Dessa forma, considerando-se que, aqui, o conjunto de dados Montalbano possui uma taxa de exibição de 10 FPS, o modelo proposto pode tanto ser executado em GPU, quanto em CPU, sem prejudicar a resposta do sistema a cada novo *frame* recebido. Isso porque o pior tempo médio de resposta do sistema ($65ms$ quando executado em CPU) é bem menor que o tempo médio entre dois *frames* consecutivos ($100ms$).

Tabela 12 – Comparação dos resultados obtidos pelos modelos *Star iRGB*$_{lstm}$ (%) e *Star RGB*$_{SoftAtt}$ para cada classe de gestos do conjunto de dados Montalbano.

Gesto	*Star iRGB*$_{lstm}$ (%)	*Star RGB*$_{SoftAtt}$ (%)	Diferença (%)
vattene	91,01	88,76	3,75
vieniqui	95,05	92,31	3,26
perfetto	93,26	95,51	-2,25
furbo	89,33	96,07	-6,74
cheduepalle	98,84	99,42	-0,58
chevuoi	93,94	92,42	1,52
daccordo	98,77	98,77	0,00
seipazzo	96,76	92,97	3,79
combinato	98,91	98,91	0,00
freganiente	95,29	91,76	3,53
ok	91,38	87,36	4,02
cosatifarei	94,15	94,68	-0,53
basta	97,79	97,24	0,55
prendere	95,11	94,02	0,07
noncenepiu	85,47	86,63	-1,16
fame	98,38	99,46	-1,08
tantotempo	97,11	98,27	-1,16
buonissimo	94,38	93,82	0,56
messidaccordo	95,56	96,11	-0,55
sonostufo	97,14	97,14	0,00

Fonte: Próprio autor.

Tabela 13 – Tempos médios de resposta do sistema calculados para a predição do gestos no conjunto de dados de teste do Montalbano utilizando o *Star iRGB*$_{lstm}$ previamente treinado. Observe-se que apenas o modelo pode estar em execução na CPU ou na GPU. O processo de cálculo do *Star iRGB* e os serviços de comunicação sempre são executados na CPU. Todos os valores estão arredondados para o maior inteiro mais próximo.

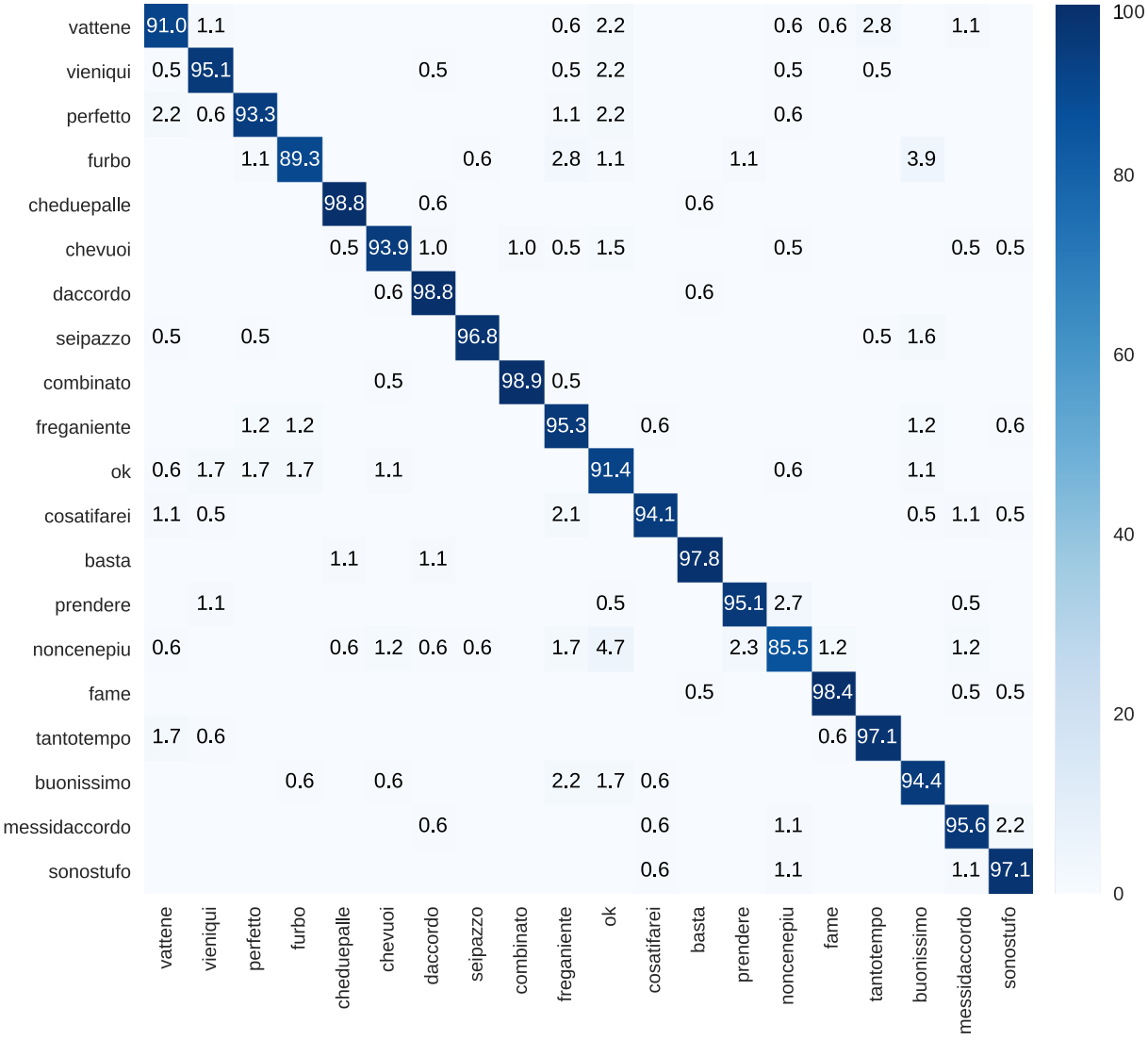
Dispositivo	Tempo médio (ms)			
	Modelo	*Star iRGB*	Comunicação	Resposta
CPU	53	1	11	65
GPU	13	-	-	25

Fonte: Próprio autor.

3.5 Comentários Gerais

Mesmo os resultados sendo promissores, é importante ressaltar mais uma vez algumas questões sobre as propostas apresentadas. Seus usos são restritos a ambientes nos quais as câmeras são estáticas ou naqueles cujo movimento relativo entre o fundo e a pessoa é mínimo. Assim, quando a câmera está em movimento, uma solução possível é a estimativa da homografia, utilizando apenas as características do plano de fundo.

Figura 26 – Matriz de confusão das predições feitas pelo modelo *Star iRGB*$_{lstm}$ treinado com o conjunto de dados Moltalbano.



Fonte: Próprio autor.

Dessa forma, conforme proposto por Wang e Schmid (2013), o movimento da câmera pode ser calculado. O conhecimento sobre os movimentos da câmera pode permitir isolar o movimento real da pessoa relacionada ao gesto e, em seguida, realizar seu reconhecimento. Além disso, é importante realizar um estudo aprofundado a fim de avaliar o comportamento do *Star RGB* e *Star iRGB* na representação de movimentos lentos e rápidos, bem como dos curtos e longos.

Como observado em Tsironi et al. (2017), técnicas como as que estão sendo propostas aqui têm a propriedade de perder detalhes das mãos. Assim, gestos diferentes que dependem da forma das mãos, mas têm o mesmo movimento, podem resultar em falsos positivos. Para ilustrar o efeito dessa restrição, uma técnica de visualização de informações, também conhecida como mapa de saliência (Cam++ (CHATTOPADHAY et al., 2018)) foi aplicada

a alguns gestos pertencentes à classe *noncenepiu*, que são os referidos piores resultados alcançados pelo modelo *Star* $RGB_{SoftAtt}$.

Em geral, os mapas de saliência do gesto *noncenepiu* (veja-se a Figura 27) são semelhantes aos mapas dos gestos *ok*, *freganiente*, *basta* e *prendere* (veja-se a linha 15 da matriz de confusão na Figura 21). Nos vídeos, é possível ver que o *noncenepiu* é feito girando o braço sobre o ponto do cotovelo, com o polegar e o dedo indicador formando um "L" (a forma da mão pode ser vista nas Figuras 28 *a* e 28 *e*). Portanto, ao isolar esse movimento, em alguns indivíduos, ele se torna muito semelhante ao movimento dos gestos *ok*, *prendere*, *freganiente* e outros. Nesses casos, o reconhecimento de gestos possivelmente seria aprimorado usando uma abordagem capaz de reconhecer os detalhes da mão, em vez de apenas os movimentos.

Além disso, a Figura 27 mostra como cada CNN do *Star* $RGB_{SoftAtt}$ extrai diferentes mapas de características relacionados ao *frame* de entrada, enquanto que, quando usa o *soft-attention*, o modelo captura as informações essenciais de cada CNN. Outra observação interessante diz respeito ao gesto *basta*, que é confundido com o gesto *noncenepiu* quando o usuário o executa com os dois braços (vejam-se as Figuras 27 - *i*, *j*, *k*, *l*). Como discutido anteriormente, os movimentos são muito semelhantes, embora as formas das mãos sejam diferentes umas das outras (Figuras 28 *e*, *f*).

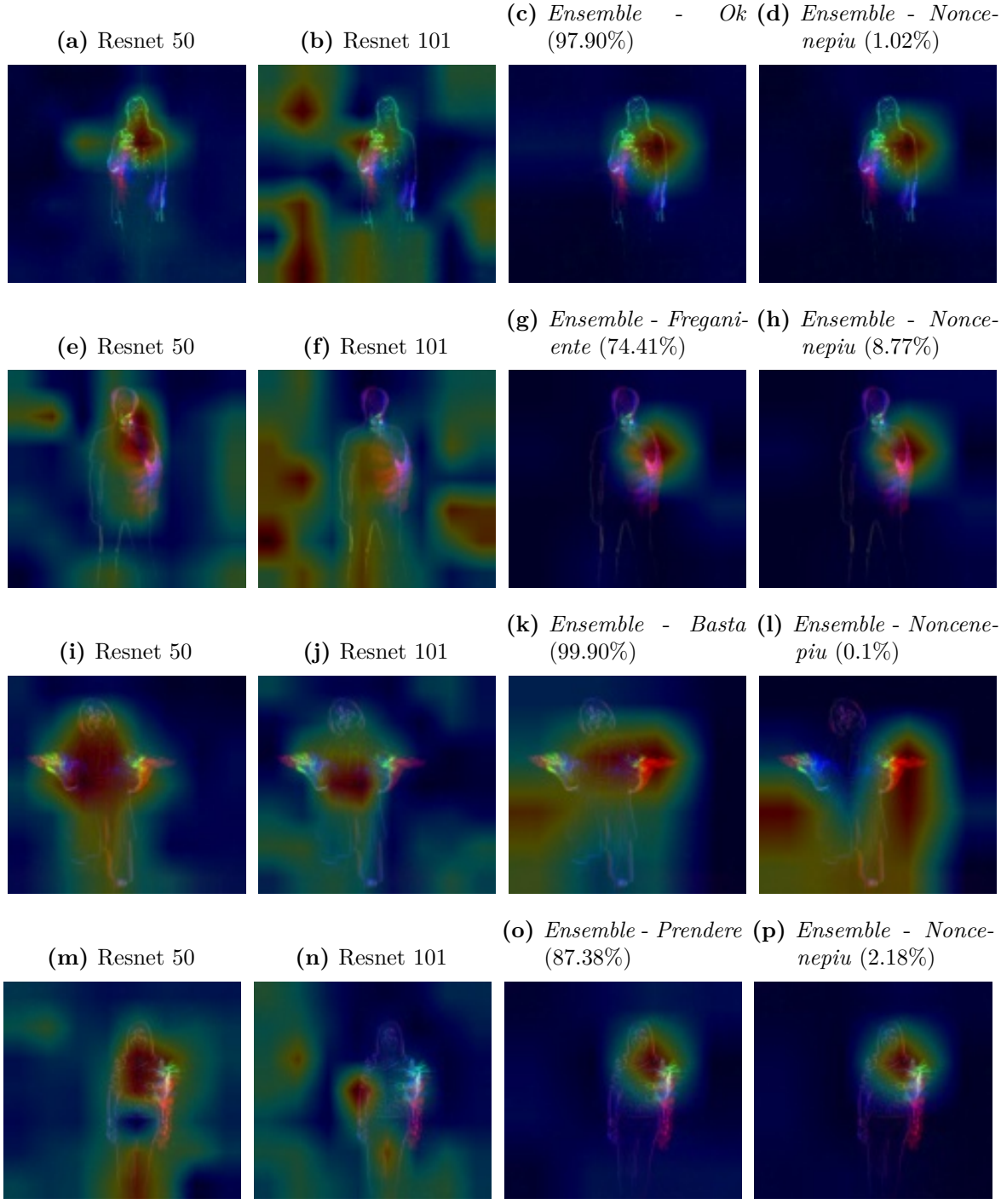
Para fins de esclarecimento, o modelo *Star* $RGB_{SoftAtt}$ foi escolhido para mostrar a importância da informação das mãos para o reconhecimento de gestos, pois, mesmo que o *Star* $iRGB_{lstm}$ tenha alcançado melhores resultados gerais em relação a ele, por ser iterativo, este último daria um mapa de saliência para cada *frame* do vídeo, o que poderia dificultar a visualização e explanação do problema. No entanto, a perda da informação das mãos é uma característica de ambas as representações *Star RGB* e *Star iRGB*.

Quanto ao *Star* $iRGB_{lstm}$, mesmo tendo obtido os melhores resultados no reconhecimento de gestos utilizando o conjunto de dados Montalbano, assim como o *Star* $RGB_{SoftAtt}$, ele ainda depende do recebimento da sequência completa de imagens contendo o gesto a ser reconhecido, mesmo ele sendo capaz de classificar o gesto *frame* a *frame*. Isso é um problema para aplicações em um ambiente real. Uma vez que, nesses casos não basta reconhecer o gesto, é necessário fazê-lo à medida que cada observação é disponibilizada para o sistema. Dessa forma, é necessário saber quando o gesto inicia e termina a fim de saber quando a tomada de decisão deve ser feita. Em situações mais reais, deve-se ter ainda a capacidade de antecipar a predição a fim de fornecer mais naturalidade e agilidade nas decisões a serem tomadas. Como os gestos podem ser entendidos como uma subclasse das ações, essas restrições podem ser estendidas ao reconhecimento e antecipação de ações.

Mesmo o *Star* $iRGB_{lstm}$ sendo mais rápido e ligeiramente melhor em termos de acurácia que o *Star* $RGB_{SoftAtt}$, é importante destacar que, para problemas onde o vídeo é disponibilizado completamente, o *Star* $RGB_{SoftAtt}$ pode ser mais vantajoso. Em um

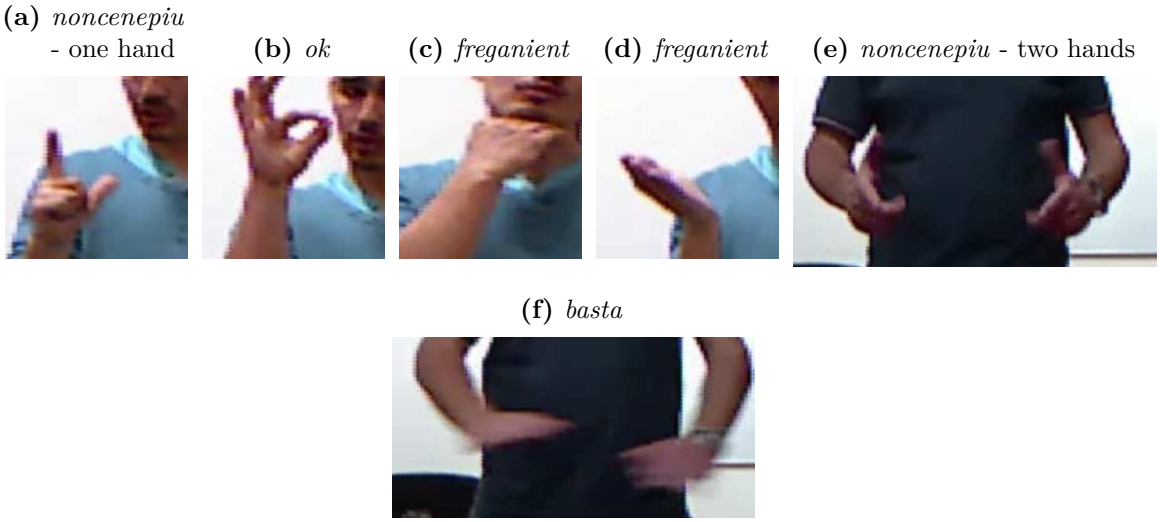
caso hipotético, ao aumentar-se a quantidade média de *frames* por cada gesto, por ser baseado em uma representação única, o tempo total de processamento gasto pelo *Star* $RGB_{SoftAtt}$ quase não se altera. Nele, apenas o processo do *Star RGB* sofre variação, o qual, por exigir pouquíssimo processamento, não influenciaria significativamente no tempo total de processamento. Já no *Star* $iRGB_{lstm}$, mesmo que o tempo de processamento para cada *frame* seja o mesmo ao longo de todo o vídeo, esse aumento no número de *frames* causa um aumento direto na quantidade total de processamento demandada pelo sistema. Assim, o *Star* $iRGB_{lstm}$ pode ser uma opção mais vantajosa que o *Star* $RGB_{SoftAtt}$ apenas quando a aplicação exige um modelo iterativo, como aquelas cujo objetivo é a antecipação, ou as que são executadas de modo *online*.

Figura 27 – Mapas de saliência de alguns gestos da classe *noncenepiu* erroneamente classificados. As porcentagens sobre as imagens das colunas 3 e 4 representam a probabilidade estimada pelo modelo para a classe mais provável e para a classe *noncenepiu*, respectivamente.



Fonte: Próprio autor.

Figura 28 – Amostras da forma das mãos para alguns gestos.



Fonte: Próprio autor.

4 ANTECIPAÇÃO DE AÇÕES

> Prediction is very difficult, especially if it's about the future.
>
> Nils Bohr

Antecipação e reconhecimento de ações são duas tarefas distintas. O reconhecimento, em sua maioria, é baseado em um modelo que usa uma sequência inteira de informações que representa uma ação executada, para associar a ação observada a uma das classes de ações possíveis (KONG; FU, 2018). Se a tomada de decisão depender da execução completa da ação, ela somente poderá ser efetuada após o término da mesma. Sendo assim, normalmente, o reconhecimento de ações não é adequado para sistemas de gerenciamento de risco ou que executam tarefas conjuntas com seres humanos. Por exemplo, em uma situação em que um carro autônomo se aproxima de um pedestre, o carro deve perceber se o pedestre atravessará a via a tempo, a fim de, se necessário, parar, reduzir ou desviar com segurança. Nesse cenário, o modelo deve não apenas reconhecer ações, mas, o que é mais importante, antecipá-las. É importante salientar que, como os gestos são um subgrupo das ações, os conceitos e as propostas apresentadas neste capítulo também podem ser adaptados para gestos.

A antecipação de ação consiste em classificar uma ação antes que ela termine, usando as informações parciais fornecidas até um determinado momento no tempo. Dessa forma, um modelo de antecipação é comumente mais complexo que um modelo de reconhecimento. Isso vem da capacidade de classificar uma ação com base em uma sequência incompleta de dados, o que dificulta a determinação da classe correta. Idealmente, todo modelo de antecipação deve ser capaz de reconhecer uma ação ao receber a sequência completa de dados; por outro lado, nem todo modelo de reconhecimento é capaz antecipar.

A proposta principal deste capítulo está focada na antecipação de ações baseadas em contexto com pequenos conjuntos de dados, onde, em vez de aprender implicitamente o contexto visual, definem-se as informações contextuais em um problema de antecipação de ações. Assim, a solução aqui proposta está baseada na análise do conjunto de dados Acticipate (DUARTE et al., 2018), em que uma pessoa entrega objetos para outras e os recebe de volta. A tarefa de interação diádica exige a predição futura (ou seja, antecipação) do movimento do braço e da cabeça, direção do olhar (*gaze*) e posição do objeto.

Considerando o Acticipate, Schydlo et al. (2018) mostraram que, usando o *gaze* em adição à pose 3D do personagem principal, uma arquitetura de aprendizado profundo, baseada na sequência temporal das observações, era capaz de melhorar o resultado da antecipação de suas ações, em relação a uma versão do mesmo modelo que utilizava apenas

a pose 3D. Ao que parece, o ponto de fixação do *gaze* pode ter dado ao modelo uma informação adicional que o ajudou a melhorar a separação entre as diferentes classes de ações. Conforme definido em Dey e Abowd (1999), contexto é qualquer informação que possa ser usada para caracterizar uma entidade. Portanto, apesar dos autores não terem explicitado, ao considerar a pose (movimento) 3D como a principal fonte de informação da entidade que representa uma ação, o *gaze* em Schydlo et al. (2018), da perspectiva do presente trabalho, pode ser visto como uma informação de contexto. No entanto, mesmo que a informação tenha ajudado na antecipação, como é possível avaliar o papel do contexto nesse tipo de problema?

As principais ações do Acticipate são: *entregar* um objeto para uma pessoa e *pôr* o objeto em um determinado ponto na mesa. Sendo assim, com o intuito de aumentar ambiguidades entre as ações para poder validar a importância de contexto no processo de antecipação, neste capítulo é proposta uma extensão do conjunto de dados Acticipate por meio do acréscimo das ações *receber* e *pegar* (informações detalhadas sobre a extensão das ações serão dadas na Seção 4.3). Por fim, em vez de usar pose 3D e *gaze*, como proposto em Schydlo et al. (2018), utilizar-se-ão apenas as informações extraídas das imagens RGB. Isso torna esta proposta mais geral e menos dependente de sensores intrusivos e (ou) de alto custo, como é o caso dos óculos seguidores da posição dos olhos e das vestimentas para captura de movimento.

Mesmo alcançando resultados satisfatórios em seus experimentos, os principais trabalhos que abordam o problema de antecipação de ações (mais detalhes na Seção 4.1) não trazem uma discussão clara sobre como utilizar suas soluções em uma situação em tempo real, uma vez que eles medem o desempenho do modelo usando a acurácia ou taxa média de observação. Eles não discutem como lidar com a antecipação de ações ou que tipo de função deve ser usada como critério de decisão. Na ausência de tal discussão, não fica claro como fazer uso das suas abordagens em aplicações reais, onde os dados são fornecidos na forma de *stream* de vídeo, como uma câmera RGB, por exemplo.

Outro problema, agora referente à maioria das soluções de aprendizado profundo, é o excesso de confiança em suas previsões. Um modelo determinístico sempre fornecerá uma predição, mesmo quando houver alta incerteza sobre a classe correta, o que influencia na qualidade da decisão a ser tomada por um sistema que utiliza tal modelo. Um modelo confiável deve ter a capacidade de avaliar sua incerteza a cada predição, possibilitando assim que o sistema tome decisões mais confiáveis.

Após levantados todos esses problemas, este capítulo apresentará duas propostas, a saber:

Proposta 1 - Um modelo sensível ao contexto baseado em uma rede neural recorrente que utiliza um limiar adaptativo para a tomada de decisão. Esse limiar

leva em consideração o valor fornecido por uma métrica de incerteza. Com isso, objetiva-se melhorar o critério de tomada de decisão para a antecipação da ação, uma vez que a incerteza possivelmente contribui para mitigar o problema de excesso de confiança de um modelo treinado com um pequeno conjunto de dados.

Proposta 2 - Um modelo de antecipação de gestos que aplica os principais conhecimentos adquiridos na tarefa de antecipação de ações apresentada na Proposta 1. Esta proposta servirá para mostrar como a proposta para a antecipação de ações pode ser aplicada em um problema relativamente mais complexo. A sua aplicação será sobre o conjunto de dados do Montalbano, o que, possivelmente, melhorá os resultados alcançados pelo *Star* $RGB_{SoftAtt}$ e *Star* $iRGB_{lstm}$, apresentados no Capítulo 3.

Dito isso, o principal objetivo deste capítulo é obter resultados que comprovem as duas últimas hipóteses levantadas na Seção 1.3, quais sejam:

- A escolha empírica da informação de contexto pode ser uma alternativa efetiva para distinguir diferentes ações ou gestos representados por movimentos ambíguos.
- Para o problema de antecipação, a incerteza sobre a predição é um limiar eficaz e mais confiável do que o valor da probabilidade estimada pelo modelo.

Por consequência, têm-se aqui as seguintes contribuições:

- Um modelo de arquitetura profunda que usa menos informações do que Schydlo et al. (2018) e supera os resultados na tarefa de antecipação de ações usando o conjunto de dados Acticipate. O modelo também será avaliado sobre o mesmo conjunto de dados, mas com seu número de ações estendido, o que possivelmente gerará mais ambiguidades que as ações originais em Duarte et al. (2018), Schydlo et al. (2018).
- Um novo e efetivo critério de tomada de decisão que pode ser usado para antecipar ações mesmo em situações de alta ambiguidade. A abordagem proposta visa minimizar a incerteza do modelo ao invés de maximizar a probabilidade de uma de suas classes. Com isso, aplicando-se um limiar sobre a incerteza estimada, é possível tomar uma decisão quanto ao melhor momento para se antecipar uma ação.
- Mostrar explicitamente a importância da informação de contexto na tarefa de antecipação de ações.
- Uma descrição formal de três métricas de desempenho que podem ser facilmente usadas para avaliar modelos de antecipação de ação.

- Um modelo de arquitetura profunda que supera os principais resultados obtidos na tarefa de reconhecimento de gestos utilizando o conjunto de dados Montalbano. O modelo também terá a capacidade de antecipação, o que, até onde se sabe, torna este trabalho o primeiro a apresentar resultados para tal tarefa sobre o referido conjunto de dados.

4.1 Trabalhos relacionados

Nos últimos anos, devido à sua importância para a realização de uma interação eficaz, a antecipação de ações tem sido abordada por muitos trabalhos, como Wang e Feng (2019), Pirri et al. (2019), Agethen, Lee e Hsu (2019), Shi, Fernando e Hartley (2018), Hu et al. (2018) e Aliakbarian et al. (2017a).

Em Shi, Fernando e Hartley (2018), os autores propuseram diminuir a dimensionalidade nas RNNs, permitindo o compartilhamento de pesos, e melhorar a representação temporal de uma ação usando um *kernel* RBF (*Radial Base Function*) sobre o estado oculto de uma LSTM. Eles propuseram alimentar uma LSTM com características extraídas por uma CNN. Em seguida, aplicaram um *kernel* RBF sobre os estados ocultos da LSTM e, por último, o resultado do RBF foi dado como entrada para uma MLP. Os autores usaram entre 20% e 50% de cada vídeo para prever as próximas características da sequência e assim realizar a antecipação. O problema aqui é a necessidade de utilização de uma quantidade fixa de *frames* para que a antecipação ocorra, o que claramente prejudica ações que podem ser antecipadas já nos primeiros *frames* de uma sequência.

Em Rodriguez, Fernando e Li (2018), os autores usaram uma arquitetura profunda conhecida como *Autoenconder* para prever o próximo movimento em um vídeo. Tal informação movimento é obtida por uma função de perda aplicada sobre a diferença entre imagens consecutivas em uma sequência. Em seguida, ela é representada na forma em uma imagem com 3 canais de cor, chamada de *Dynamic Image* (BILEN et al., 2016). Assim, assumindo a suposição Markoviana de que o movimento no instante t depende apenas do movimento no instante $t-1$, após obter uma sequência de imagens usando S *frames*, o modelo as utiliza para gerar as próximas k imagens dinâmicas, em que $k \geq 1$. Na sequência, essas imagens geradas alimentam um outro modelo que estima a distribuição de probabilidade sobre as classes de ações por elas representadas. Apesar de ser uma abordagem interessante, além de também depender de uma quantidade fixa de S *frames* para antecipar uma ação, essa abordagem não é indicada quando existem longas dependências entre imagens.

Uma outra desvantagem de ambos os trabalhos é a necessidade de grandes conjuntos de dados para treinar seus modelos, uma vez que eles usam modelos com alta capacidade.

Assim como as propostas do Capítulo 3, eles também usam apenas o movimento como fonte de informação de uma ação, o que pode prejudicar ações relacionadas não apenas ao movimento, mas também às informações de contexto.

Em Aliakbarian et al. (2017b) é proposto um modelo para antecipar ações baseado apenas em imagens RGB. Os autores usam como extrator de características uma CNN pré-treinada do tipo VGG16 e, como classificador, duas LSTMs que predizem as classes correspondentes a cada *frame* de um vídeo. Uma abordagem semelhante também é apresentada em Aliakbarian et al. (2016).

A LSTM também é usada para antecipar ações de motoristas usando apenas imagens RGB (GITE; AGRAWAL; KOTECHA, 2019) ou em combinação com informações de GPS (GITE; AGRAWAL, 2019). Outras abordagens, como Wang, Yuan e Wang (2019a), usam redes adversariais geradoras (GANs - do inglês *Generative Adversarial Networks*) para prever imagens futuras e então antecipar a ação; ou arquiteturas mais sofisticadas, como em Liu et al. (2020), que usa modelos gráficos convolucionais (CGMs - do inglês *Convolutional Graphical Models*) para prever quando um pedestre vai atravessar a rua.

Apesar de todos trabalhos acima apresentarem resultados interessantes em termos de acurácia, nenhum deles explica como a antecipação de ações deve ser realizada em um cenário real, quando não é possível saber o tamanho da sequência de entrada. Eles também não discutem que tipo de critério de tomada de decisão poderia ser usado em tal situação.

Mesmo em trabalhos como Liu et al. (2019), que visam a antecipação *online* de ação em vídeos, os autores apresentam os resultados na forma apenas da acurácia de reconhecimento em cada observação. Nada é mencionado sobre como tomar a decisão de quando a ação deve ser antecipada. Apenas alguns trabalhos tratam dessa questão. Em Jain et al. (2015) e Jain et al. (2016), os autores usam um limiar sobre a distribuição de probabilidade fornecida por um modelo do tipo HMM (*Hidden Markov Model*) para antecipar as manobras de motoristas. No entanto, conforme discutido em Jain et al. (2015), esta abordagem enfrenta problemas em situações ambíguas, onde não é possível ter certeza sobre a ação a ser antecipada, mesmo quando a probabilidade ultrapassa o limiar especificado.

Em complemento aos problemas citados, muitas das abordagens mencionadas não são adequadas para serem aplicadas em pequenos conjuntos de dados, uma vez que a alta capacidade de seus modelos pode levá-los a *overfitting*. Portanto, Schydlo et al. (2018) propõe um método diferente para antecipar a ação no conjunto de dados Acticipate - um pequeno conjunto de dados colaborativo usado para entender o papel do *gaze* na antecipação de ações (DUARTE et al., 2018), explicado na Seção 4.3. Sua abordagem consiste em alimentar uma célula LSTM com a pose 3D de um indivíduo (informações de MoCap - do inglês *Motion Capture*) e *gaze* (ponto de fixação do olhar) e depois passar a saída da LSTM para um classificador *softmax*. Eles treinaram dois modelos com

observações diferentes: um com apenas a pose 3D e outro com pose 3D mais o *gaze*. Como resultado, eles concluíram que as ações no conjunto de dados poderiam ser antecipadas com uma antecedência de até 92ms quando o modelo usava o olhar como uma informação complementar à pose. Esses resultados ajudam a perceber a importância de usar mais do que apenas movimentos (no caso deles, a evolução da pose no tempo) para antecipar ações. No entanto, os autores não se atentaram para o fato de que seu modelo não reconheceu todas as ações (100% de acurácia no reconhecimento de ações), mesmo depois de receber toda a sequência.

Além disso, seus resultados para a antecipação de ações foram mostrados com base em apenas uma amostra de ação (um clipe de vídeo). Resultados mais conclusivos deveriam apresentar estatísticas para todas as classes em todo o conjunto de dados. Em complemento, eles também não responderam à pergunta: quando a saída de um modelo é confiável o suficiente para antecipar uma ação? A partir da falta de comentários, assume-se que isso poderia ser feito usando-se um limiar sobre o valor da probabilidade estimada pelo modelo, assim como mencionado em Jain et al. (2016). No entanto, como abordado em Jain et al. (2015), observada a ambiguidade inerente ao problema de antecipação, dado que a probabilidade de uma das classes do modelo tenha ultrapassado o limiar estabelecido, a ação deve ser predita imediatamente ou é necessário esperar por mais observações a fim de aumentar a confiança na predição? Essas e outras perguntas serão respondidas nas seções subsequentes deste capítulo.

4.2 Antecipação de ações

Nesta seção, serão mostradas as definições utilizadas na antecipação de ações, suas principais propriedades e como o problema é abordado. Assim, os trabalhos que tentam resolver a tarefa de antecipação podem ser divididos em dois grupos principais: (*i*) antecipação de ação[1], em que uma ação deve ser prevista antes de ser totalmente executada (BLOOM; ARGYRIOU; MAKRIS, 2017; HU et al., 2018; WANG; YUAN; WANG, 2019b; WANG et al., 2019; JI et al., 2018); e (*ii*) antecipação de evento, em que um evento deve ser previsto antes mesmo do seu início (FELSEN; AGRAWAL; MALIK, 2017; NEUMANN; ZISSERMAN; VEDALDI, 2019, 2019). Neste trabalho, "antecipação de ação" é entendida como no primeiro conjunto de trabalhos: predição de uma ação usando dados sequenciais, considerando que a ação já deve ter sido iniciada.

[1] Principais termos: *action anticipation, early action recognition, early action prediction e action prediction*

4.2.1 Definição do problema

Para começar, é importante definir formalmente a tarefa de antecipação de uma ação. Sendo assim, seja $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N \mid \mathbf{x}_t \in \mathbb{R}^{c\times 1}\}$ uma sequência com N observações que representam a execução de uma ação específica $y \in \mathcal{Y}$, onde $\mathcal{Y}$ é um conjunto com c classes de ações. Aqui, $\mathbf{x}_t$ representa uma observação feita no momento t. Agora, considerando que $\mathcal{X}_{t_1:t_2}$ representa uma sequência indexada composta pelas observações recebidas entre os tempos t_1 e t_2, defini-se um modelo M para o problema de classificação de ação como uma função de mapeamento, parametrizada por $\boldsymbol{\theta}$, que recebe como entrada $\mathcal{X}_{1:t}$ (t observações de $\mathcal{X}$) e retorna como saída o vetor $\hat{\mathbf{s}} \in [0,1]^{c\times 1}$, representando a probabilidade estimada de que a sequência $\mathcal{X}$ pertença a cada classe de ação.

Nas tarefas de reconhecimento de ações, o modelo M tem todas as observações da sequência $\mathcal{X}$ ($t = N$) disponíveis para estimar a probabilidade $\hat{\mathbf{s}}$. Por outro lado, numa tarefa de antecipação, a ação ainda não foi completamente executada, portanto, apenas uma parte inicial de $\mathcal{X}$ está disponível ($t < N$) para que M possa estimar $\hat{\mathbf{s}}$.

Na Equação (4.1), o parâmetro $\boldsymbol{\theta}$ pode ser estimado resolvendo o problema de otimização da Equação (4.2),

$$\hat{\mathbf{s}} = M(\mathcal{X}_{1:t}, \boldsymbol{\theta}). \tag{4.1}$$

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \{\mathcal{L}(\boldsymbol{\theta}, \mathcal{D})\} \tag{4.2}$$

onde $\mathcal{D} = \{(\mathcal{X}^{(1)}, y^{(1)}), (\mathcal{X}^{(2)}, y^{(2)}), \cdots, (\mathcal{X}^{(K)}, y^{(K)})\}$ é o conjunto de treinamento, em que cada par $(\mathcal{X}^{(i)}, y^{(i)})$ representa a sequência i de uma ação e seu respectivo rótulo; K é o número de sequências no conjunto de treinamento; e $\mathcal{L}$ é uma função de custo.

Durante a etapa de predição, não é possível (ou pelo menos não é comum) conhecer o valor de N e, portanto, não é possível saber quando a ação terminará. Portanto, a cada momento t, M usa a sequência $\mathcal{X}_{1:t}$ já observada, enquanto a função g é responsável por determinar a classe da ação que está sendo executada no instante t.

$$\begin{aligned} \hat{\mathbf{s}} &= M(\mathcal{X}_{1:t}, \hat{\boldsymbol{\theta}}) \\ \hat{y} &= g(\hat{\mathbf{s}}). \end{aligned} \tag{4.3}$$

Para tarefas de reconhecimento de ações, a função discriminante g pode ser definida como:

$$g(\hat{\mathbf{s}}) = \operatorname{argmax}(\hat{\mathbf{s}}), \tag{4.4}$$

pois o modelo M está mais confiante na classe de maior probabilidade estimada, $\hat{\mathbf{s}}$. Por outro lado, para tarefas de antecipação, como M usa apenas parte das observações,

quando a distribuição $\hat{\mathbf{s}}$ está próxima de uma distribuição uniforme, não é possível ter certeza sobre a classe correta. Portanto, a Equação (4.4) não é uma função discriminante adequada para antecipar ações.

Dessa forma, uma opção mais adequada é usar uma função discriminante com um parâmetro de limiar p, conforme apresentada na Equação (4.5).

$$g(\hat{\mathbf{s}},p) = \begin{cases} \text{argmax}(\hat{\mathbf{s}}), & h(\hat{\mathbf{s}}) > p \\ -1, & \textit{caso contrário} \end{cases}. \tag{4.5}$$

Depois que p é especificado como um valor de probabilidade, h pode ser definido como:

$$h(\hat{\mathbf{s}}) = max(\hat{\mathbf{s}}) \tag{4.6}$$

Na Equação (4.5), um valor de $p \geq 0{,}9$, por exemplo, pode significar que o modelo está altamente certo sobre suas previsões e é possível antecipar uma ação, o que favorece o uso desse modelo em tempo real. Por outro lado, quando retorna -1 significa que não tem certeza da classe correta e precisa de mais observações para melhorar a sua certeza.

4.2.1.1 Modelos determinísticos

Esses tipos de modelos geralmente usam a função g na Equação (4.7) para reconhecer ações. Como no reconhecimento de ações geralmente $t = N$, significa que o modelo recebe a sequência completa de amostras que representam uma ação. Dessa forma, modelos de reconhecimento tendem a ser mais confiantes sobre a distribuição de probabilidade $\hat{\mathbf{s}}^{(t)}$. Em consequência disso, eles não são eficazes quando tentam resolver tarefas de antecipação de ação, pois a antecipação pode ser executada mesmo quando $\hat{\mathbf{s}}^{(t)}$ assemelha-se a uma distribuição uniforme.

$$g(\hat{\mathbf{s}}^{(t)}) = \text{argmax}(\hat{\mathbf{s}}^{(t)}) \tag{4.7}$$

Uma função de antecipação efetiva deve levar em consideração a certeza do modelo sobre sua decisão. Portanto, ao considerar as probabilidades estimadas como um valor a ser utilizado, não é possível ter certeza sobre a predição quando $\hat{\mathbf{s}}^{(t)}$ retorna valores semelhantes para diferentes classes. Dessa forma, uma melhor opção para esse tipo de modelo é usar uma função com um parâmetro de limiar p, conforme apresentada na Equação (4.5).

Ao usar p como um valor de probabilidade, h assume a forma de uma função max, como na Equação (4.8).

$$h(\hat{\mathbf{s}}^{(t)}) = max(\hat{\mathbf{s}}^{(t)}) \tag{4.8}$$

4.2.1.2 Modelos estocásticos

Esses tipos de modelos têm a capacidade de fornecer um valor estimado da incerteza sobre suas predições. Portanto, ao usar um modelo estocástico para antecipar a ação, uma função discriminante diferente g deve ser usada, como a apresentada na Equação (4.9), em que u é um limar de valor de incerteza e h é a função que estima a incerteza epistêmica do modelo sobre sua predição $\hat{\mathbf{s}}$.

$$g(\hat{\mathbf{s}},u) = \begin{cases} \text{argmax}(\hat{\mathbf{s}}), & h(\hat{\mathbf{s}}) < u \\ -1, & \textit{caso contrário} \end{cases} \tag{4.9}$$

Para diferentes modelos e diferentes propostas, existem diferentes maneiras de calcular a incerteza de predição. Assim, h pode assumir diferentes formas. Uma função candidata para h será apresentada na Seção 4.4

4.2.2 Métricas de Avaliação

Após determinar como antecipar uma ação, é essencial decidir como determinar a qualidade do modelo M. Portanto, devido à falta de referências de métricas explícitas para avaliar modelos de antecipação, três métricas serão descritas formalmente: (*i*) Acurácia em cada proporção de observação, (*ii*) Acurácia de antecipação e (*iii*) Taxa média de observação.

Acurácia em cada proporção de observação. Considerando que cada sequência $\mathcal{X}$ pode ter um comprimento diferente N, essa métrica ajuda a avaliar todas as sequências em uma escala de tempo normalizada. Assim, a taxa de sucesso ao antecipar uma ação após uma taxa média de observação (TMO) r , com um limiar de antecipação p, pode ser calculada da seguinte forma:

$$ACC(r) = \frac{1}{K}\sum_{i=1}^{K} pred(\mathcal{X}^{(i)}_{1:\lceil rN \rceil}, y^{(i)}, p), \tag{4.10}$$

onde,

$$pred(\mathcal{X}_{1:t}, y, p) = \begin{cases} 1, & g(M(\mathcal{X}_{1:t}), p) = y \\ 0, & \textit{caso contrário} \end{cases}. \tag{4.11}$$

e K representa o número de sequências no conjunto de dados a ser avaliado.

Na Equação (4.10), em termos de r, $t = \lceil rN \rceil \; \forall \; r \in (0,1]$, em que N é o número de observações em uma sequência $\mathcal{X}^{(i)}$. No entanto, em termos de t, $r = t/N \; \forall \; t \in \{1,2,...,N\}$.

Acurácia de antecipação. Em uma situação real, o modelo não tem acesso ao rótulo de cada observação. Portanto, a avaliação da antecipação durante o treinamento deve ser

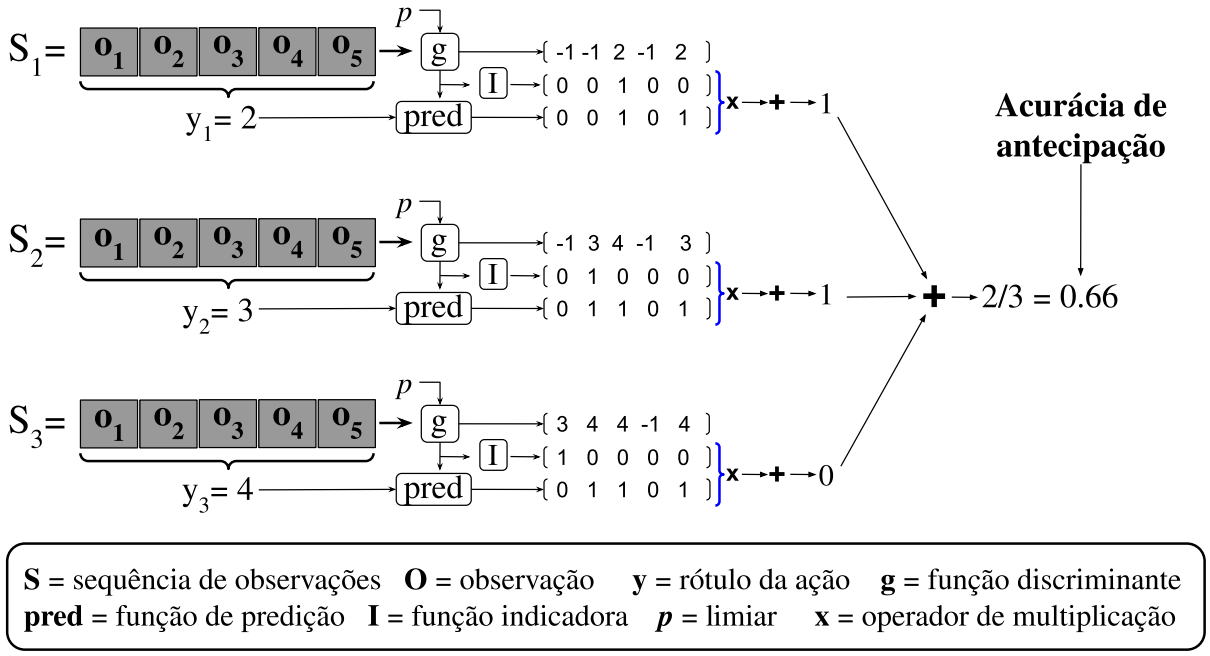
realizada quando o modelo fizer sua primeira previsão para cada sequência. Nesse sentido, essa métrica de classificação mede a taxa de sucesso do modelo M ao antecipar cada ação pela primeira vez. É calculada como a acurácia média de cada classificação. Dessa forma, essa métrica não considera em qual observação a ação foi prevista, e sim, se ela foi prevista corretamente ou não. A Equação(4.12) apresenta como o seu cálculo pode ser feito.

$$ACC_{act} = \frac{1}{K}\sum_{i=1}^{K}\sum_{t=1}^{N-1} I(t)pred(\mathcal{X}_{1:t}^{(i)},y^{(i)},p), \tag{4.12}$$

onde I é uma função indicadora que desabilita as predições caso a antecipação já tenha ocorrido, e pode ser calculada como descrita na Equação (4.13). Um exemplo detalhado de como aplicar essa métrica é apresentado na Figura 29.

$$I(t) = \begin{cases} 0, & (g(M(\mathcal{X}_{1:t}),p) = -1) \vee (t > 1 \wedge \sum_{i=1}^{t-1} I(i) = 1) \\ 1, & \textit{caso contrário} \end{cases} \tag{4.13}$$

Figura 29 – Exemplo gráfico de como calcular a predição da antecipação usando a Equação (4.12).



Taxa média de observação. Essa medida concentra-se na quantidade média de observações necessárias para antecipar corretamente uma ação. Pode ser implementada de acordo com a Equação (4.14). Observe-se que, quando o modelo está correto, ele recebe o valor t, o qual corresponde ao número da observação em que a predição é executada. No entanto, quando erra a antecipação, o mesmo é penalizado, recebendo assim o tamanho N da

sequência.

$$E_{obs} = \frac{1}{K}\sum_{i=1}^{K} obs(\mathcal{X}^{(i)},y^{(i)},p), \tag{4.14}$$

onde,

$$\begin{aligned} obs(\mathcal{X},y,v) &= \frac{1}{N} min(\{f_{pred}(\mathcal{X}_{1:t},y,p,t,N)\}_{t=1}^{N}) \\ f_{pred}(\mathcal{X}_{1:t},y,p,t,N) &= \begin{cases} t, & pred(\mathcal{X}_{1:t},y,p) = 1 \\ N, & caso\ contrário \end{cases}. \end{aligned}$$

4.3 Desenvolvimento da primeira proposta

Para uma melhor compreensão de como as intuições surgiram e resultaram na proposta de antecipação de ação a ser apresentada, é necessário analisar o conjunto de dados utilizado e, assim, perceber como os questionamentos foram surgindo.

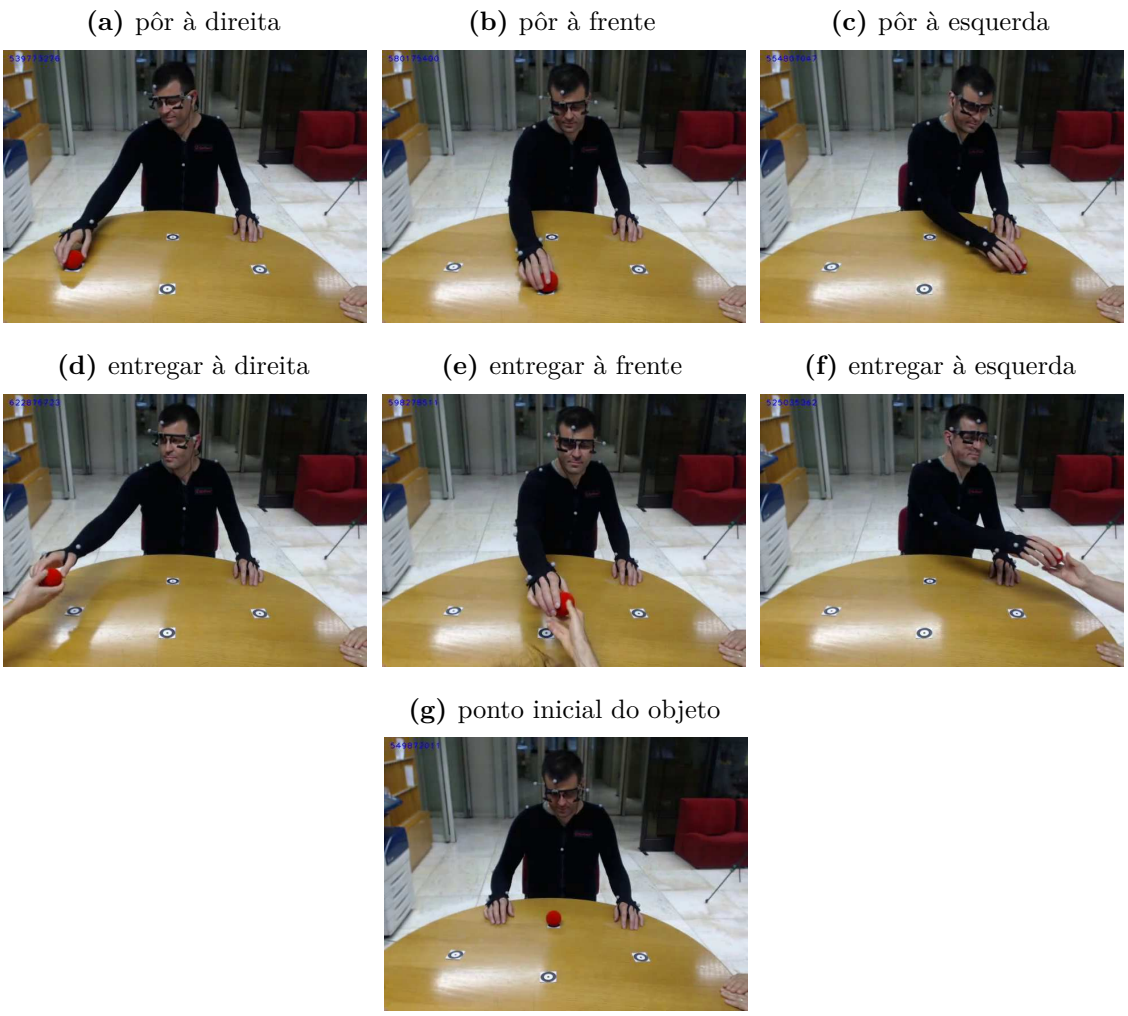
4.3.1 Conjunto de dados Acticipate

O conjunto de dados Acticipate [2] foi adquirido com a proposta de estudar a influência do *gaze* na antecipação de ação e (ou) de intenção (DUARTE et al., 2018; SCHYDLO et al., 2018) em um ambiente colaborativo. Ele é composto por 120 capturas, distribuídas em seis classes. Durante a aquisição, o ator usava óculos que rastreavam a direção do olhar, também conhecido como *eye tracker* (*Pupil Labs* (KASSNER; PATERA; BULLING, 2014)) e uma vestimenta com 25 marcadores. Ele devia executar seis ações diferentes: entregar um objeto para alguém e pôr um objeto em três pontos sobre a mesa, cada uma delas com três variantes. Nas ações de *entregar*, ele devia dar um objeto (neste caso, uma pequena bola vermelha) a um dos três voluntários localizados ao seu lado direito, esquerdo ou à sua frente. Nas ações de *pôr*, ele devia posicionar o mesmo objeto em um dos três pontos da mesa localizados à sua direita, à sua frente ou à sua esquerda. Toda ação começava com o objeto em repouso em um ponto próximo ao ator e terminava quando o objeto retornava ao mesmo ponto. A Figura 30 apresenta uma amostra de cada ação e o ponto inicial do objeto.

Cada captura consiste em dados tridimensionais correspondentes às posições dos marcadores no traje do ator, capturados por um sistema de MoCap da *OptiTrack*[3], a 120Hz; ponto de fixação 2D do olhar capturado pelo *eye tracker* a 60Hz; e um vídeo

[2] Download: <http://vislab.isr.ist.utl.pt/datasets/>
[3] https://optitrack.com/

Figura 30 – Exemplo de cada uma das ações presentes no conjunto de dados Acticipate, além do ponto inicial do objeto



Fonte: Montado pelo próprio autor a partir dos vídeos do conjunto de dados Acticipate.

RGB capturado por uma câmera de frente para o ator, a 30Hz. O conjunto de dados é desbalanceado, uma vez que cada classe tem um número diferente de amostras: 17 (entregar à direita), 23 (entregar à frente), 20 (entregar à esquerda), 24 (pôr à direita), 19 (pôr à frente) e 17 (pôr à esquerda).

Neste capítulo, em qualquer referência ao conjunto de dados Acticipate, o movimento é representado pela mudança de posição dos dois braços.

4.3.2 Análise do conjunto de dados

Ao analisar o conjunto de dados, é possível perceber que, em muitos casos, o movimento não possui informações suficientes sobre a ação para fornecer uma boa antecipação. Por exemplo, cada ação de *pôr* e *entregar* possui movimentos semelhantes, dependendo de sua direção (esquerda, frente ou direita). No entanto, depois de levar em conta as

informações do olhar, percebe-se que a ação pode ser antecipada muito antes. A direção do olhar indica se o usuário posicionará o objeto na mesa ou o entregará a um dos voluntários. Como discutido anteriormente, caso o movimento seja tomado como a entidade principal de cada ação, pode-se considerar a direção do olhar como uma informação de contexto: uma informação adicional que ajuda a caracterizar a entidade movimento. Dessa forma, o movimento juntamente com a direção do olhar fornecem dados suficientes para antecipar ações nesse conjunto de dados. No entanto, se ele fosse dividido em mais ações, o olhar e o movimento continuariam sendo fontes de informação suficientes para antecipá-las?

Uma característica interessante, mas não considerada, desse conjunto de dados é que, uma vez que a interação envolve apenas um objeto, após a execução de cada ação, o ator deve pô-lo em seu ponto de partida. Portanto, quando posto, o ator deve pegá-lo de volta e, sempre que o entregar a alguém, ele deve recebê-lo de volta. Um exemplo simples desse comportamento pode ser visto na Figura 31. Dessa forma, pode-se estender o conjunto de dados de 6 para 12 ações: *entregar*, *pôr*, *pegar* e *receber* (cada uma com as direções esquerda, direita e à frente).

Figura 31 – Amostra de duas ações (pôr no meio e entregar à direita) do conjunto de dados Acticipate

(a) pôr no meio

(b) entregar à direita

Fonte: Montado pelo próprio autor.

Considerando agora o conjunto de dados estendido, ao restringir as análises ao movimento e ao olhar (Figura 32 (a)-(d)), nota-se que, mesmo com a direção do olhar, não é possível executar uma antecipação correta entre as ações *entregar*/*receber* ou *pôr*/*pegar* quando estão na mesma direção. Nesse caso, é necessário aguardar mais observações. Por outro lado, caso não sejam aplicadas restrições sobre o que pode ser analisado em cada imagem (Figura 32 (e-h)), é possível antecipar as ações do conjunto de dados estendido (12 ações) tão rápido quanto em sua configuração original (6 ações). Em alguns casos, como na Figura 32 (f), a ação pode ser antecipada apenas com a observação do primeiro *frame*. Isso é possível após levar em consideração a posição do objeto como uma outra informação essencial de contexto. Por exemplo, a posição inicial do objeto possibilita antecipar uma ação *pegar* observando apenas a primeira imagem.

Algo semelhante ocorre com as ações *receber* quando o objeto está fora de cena sendo mantido por um voluntário. Para tais ações, após ver o primeiro *frame*, não é possível garantir qual é a ação, uma vez que ela depende da direção. No entanto, pode-se dizer que será uma ação do tipo *receber*. Dessa maneira, a antecipação correta ocorre após perceber a direção do olhar ou do movimento, além da presença ou não do objeto na cena e a sua posição. Isso ajuda a eliminar ações menos prováveis e permite focar em informações que ajudam a encontrar a ação correta. A Figura 32 ilustra quatro situações em que existem grandes ambiguidades entre ações, e as informações do objeto são críticas para reduzi-las.

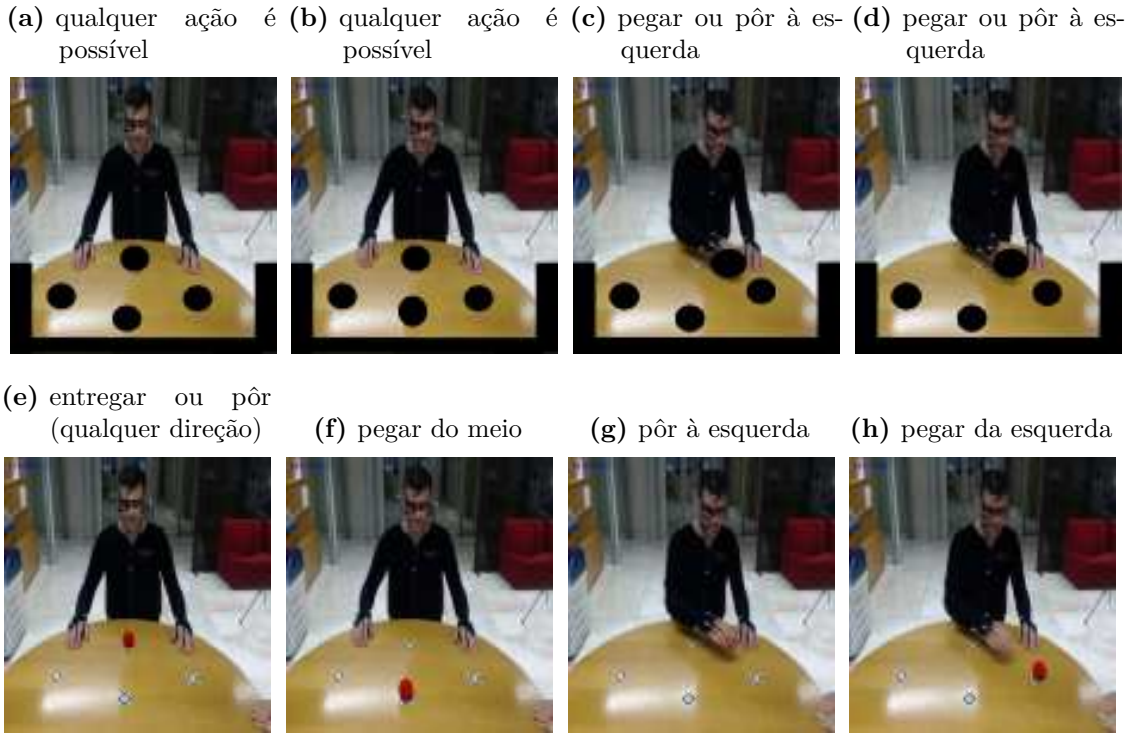
4.3.3 Antecipação

Embora seja possível antecipar ações no conjunto de dados estendido, em alguns casos há problemas sobre a antecipação que devem ser levados em consideração. Até as pessoas podem ser confundidas pelo excesso de confiança em sua capacidade preditiva. Em um caso específico, como apresentado na Figura 33, a voluntária antecipou erroneamente uma ação após observar um movimento semelhante a outro. Sua confiança em sua predição a enganou. Assim, mesmo pessoas, em algumas situações, precisam ter mais certeza sobre a ação antes de tomar uma decisão. Se uma pessoa pode ser enganada por seu excesso de confiança, esse problema é possivelmente maior em um modelo computacional.

O excesso de confiança em uma predição pode levar o modelo a tomar decisões equivocadas em uma aplicação de tempo real. Uma possível solução para mitigar esse problema é fornecer ao modelo a capacidade de estimar a incerteza sobre sua predição. Em um modelo determinístico, mesmo com um alto valor de limiar de probabilidade $p > 0{,}9$, uma ação pode ser antecipada erroneamente mesmo que ele esteja excessivamente confiante em sua predição. Esse excesso de confiança pode ser provocado pela falta de dados para ajudá-lo a diferenciar classes ambíguas.

Uma possível solução para aumentar a certeza do modelo é levá-lo a fazer mais

Figura 32 – Situações no conjunto de dados estendido com grandes ambiguidades ao analisar apenas a direção do olhar e o movimento. Em (a) e (b), qualquer ação é possível. É necessário aguardar o movimento para poder inferir a direção, mas, mesmo sabendo a direção, é necessário observar a ação quase que completamente para distinguir entre as ações *entregar/receber* e *pôr/pegar*. Nos dois casos (c) e (d), o movimento começa no lado esquerdo, enquanto o olhar é direcionado para a mesa. Com isso, a ação possível é *pôr* ou *pegar* na direção esquerda. Para as ações mostradas em (e)-(h), como a posição do objeto é levada em consideração, as ambiguidades podem ser reduzidas ou mesmo eliminadas. Em (e) e (f), o número de ações possíveis é reduzido após conhecer a posição do objeto. Em (e), as ações *pegar* e *receber* não são possíveis. Por outro lado, em (f) somente a ação *pegar da frente* é possível. O mesmo ocorre em (g) e (h). Em (g) a ação mais provável é a *pôr à esquerda* e em (h) a única possibilidade é *pegar da esquerda*. Observe-se que em (f) e (h) a ação pode ser antecipada observando-se apenas uma imagem.



Fonte: Montado pelo próprio autor.

z previsões antes de tomar a decisão sobre a classe de ação correta. Dessa forma, se nas próximas z observações a classe prevista permanecer a mesma, a confiança do modelo sobre a classe correta se comprovará e o mesmo poderá antecipar a ação. No entanto, mesmo que pareça uma boa solução, qual seria o melhor valor para z? Uma escolha imprecisa desse novo parâmetro pode adiar a antecipação de ações que não têm problemas de ambiguidade em no mínimo z observações. Além disso, o valor escolhido para z pode não ser suficiente para ações muito ambíguas.

Uma solução mais razoável é usar a incerteza estimada na predição como valor de entrada para uma função de limiar, ao invés do valor da probabilidade estimada.

Figura 33 – Duas amostras de ações do conjunto de dados *Acticipate*. Em (a), a voluntária antecipou erroneamente a ação, pensando que seria uma ação de *entregar à esquerda* (mostrada em (b)) em vez de uma *pôr à esquerda*.

(a) Pôr à esquerda

(b) Entregar à esquerda

Fonte: Montado pelo próprio autor.

Dessa maneira, o modelo pode antecipar uma ação quando estiver mais certo sobre sua predição. Com isso, ações ambíguas, que provavelmente fornecem mais incerteza ao modelo, precisariam de mais observações para serem antecipadas adequadamente enquanto aquelas com menos ambiguidades poderiam ser antecipadas anteriormente. Essa solução pode ser tomada como um valor z personalizado para cada ação e determinado pelo próprio modelo durante o seu treinamento.

4.4 Proposta 1: o papel do contexto e da incerteza na antecipação de ações

Nesta seção será descrita a solução proposta, a qual está dividida em três etapas principais: (*i*) extração e seleção de características, (*ii*) incorporação (*embedding*) de caracterísicas e (*iii*) o modelo de classificação. Os próximos tópicos abordarão cada uma dessas etapas em detalhes.

Como visto antes, muitos trabalhos, como aqueles baseados em fluxo duplo ((KWON et al., 2018; SIMONYAN; ZISSERMAN, 2014a; CARREIRA; ZISSERMAN, 2017b)), possuem uma fonte de contexto baseada numa CNN que extrai características de uma imagem RGB contendo toda a cena observada. Apesar de alcançar o estado da arte em muitas tarefas de reconhecimento de ações, não são indicadas para problemas representados por pequenos conjuntos de dados. Sendo assim, um dos objetivos desta proposta é mostrar a importância de se analisar o problema a ser resolvido a fim de determinar quais tipos de informações são mais relevantes para serem utilizados como fonte contexto, ao invés de

deixar essa tarefa por conta do próprio modelo.

Dito isso, após a análise realizada na Seção 4.3, aqui, o *gaze* e o objeto representarão a informação de contexto de cada ação. O *gaze* foi escolhido pois, como apresentado em Schydlo et al. (2018) e Duarte et al. (2018), é uma importante fonte de informação para a antecipação de intenções. Quanto ao abjeto, percebeu-se na referida análise que ele representa uma importante fonte de informação para a distinção entre algumas classes de ações do conjunto de dados a ser utilizado. Dessa forma, propõe-se um modelo baseado em redes neurais artificiais que antecipe ações representadas por sequências de observações de comprimento variável. O modelo proposto possui duas versões: uma determinística e uma estocástica.

Primeiro, duas versões do modelo determinístico serão aplicadas ao conjunto de dados original com seis ações e serão comparados com os resultados apresentados em Schydlo et al. (2018) e com os de outros cinco modelos de *baselines*. Em seguida, dentre todos os modelos (propostos e *baselines*) aquele com o melhor resultado tornar-se-á a nova referência para a comparação com outras seis versões do modelo determinístico aplicadas ao conjunto de dados estendido (doze ações). Por fim, o modelo com melhor resultado na base estendida será utilizado como *baseline* para três diferentes tipos de modelos estocásticos.

Com essa abordagem, será possível discutir qual modelo é melhor para a antecipação de ações e suas principais vantagens e desvantagens. O objetivo de treinar diferentes tipos e versões de modelos com diferentes versões do conjunto de dados (original e estendida) é fornecer resultados que levem a conclusões mais concretas.

4.4.1 Extração e seleção de características

Nesta proposta, a direção do olhar e a posição do objeto representarão informações de contexto que complementarão a informação de movimento representada pela evolução das articulações do corpo. No entanto, como o objetivo é usar apenas imagens RGB, as informações sobre a direção do olhar e das articulações do esqueleto não estão diretamente disponíveis. Sendo assim, decidiu-se alimentar o modelo *Openpose* (Cao et al., 2019) com cada imagem RGB presente no conjunto de dados *Acticipate*, com o objetivo de extrair tais informações. Para isso, utilizou-se a versão *Openpose* treinada para o conjunto de dados COCO que fornece 18 articulações 2D para o corpo e 25 pontos 2D para cada mão. A escolha do *Openpose* foi devido ao mesmo ser o *framework* de detecção mais utilizado para extração de esqueleto nos últimos anos em imagens 2D. Isso se deve a sua característica de rápida execução sem prejudicar a qualidade das detecções. Mais informações sobre o mesmo podem ser vistas em Cao et al. (2019).

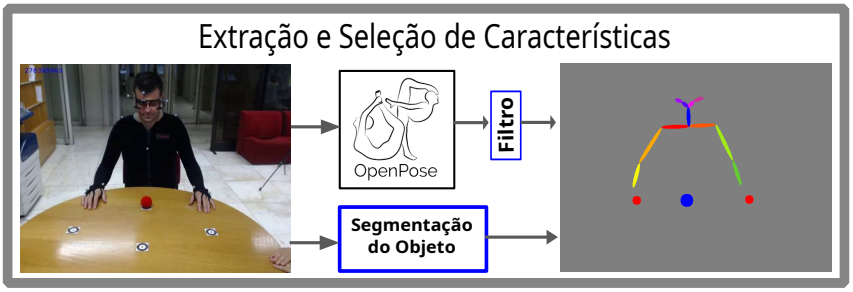
É importante mencionar que, em Schydlo et al. (2018), os autores usaram a informação do olhar e das articulações do corpo em 3D. Isso foi possível porque eles

utilizaram o ponto de fixação do olhar fornecido pelo *eye tracker* e o esqueleto em 3D do ator fornecido pelo sistema de MoCap. Diferentemente, como a proposta aqui é usar apenas as imagens RGB, têm-se apenas articulações em 2D fornecidas pelo *Openpose* para serem usadas como dados de entrada para o modelo. Além disso, como o ator usava óculos durante a aquisição de dados, os principais algoritmos para a estimativa da direção do olhar não funcionaram razoavelmente (BALTRUSAITIS et al., 2018). Por esse motivo, decidiu-se fazer uso dos pontos da cabeça, fornecidos pelo *Openpose*, como informações que possivelmente representarão a direção da cabeça ou até mesmo a direção do olhar. Tal representação será uma tarefa a ser assumida pelo próprio modelo de maneira não supervisionada. Além disso, a fim de reduzir a dimensionalidade, calculou-se o ponto central de cada mão ao invés de usar diretamente seus 25 pontos 2D.

Para as informações do objeto, foi extraído o ponto central da bola vermelha para cada *frame* usando uma função de segmentação por cor. Todo esse procedimento de pré-processamento está resumido na Figura 34 e descrito a seguir:

1. O modelo *Openpose* recebe uma imagem RGB representando uma observação. Esta operação resulta em 18 juntas 2D do corpo e 25 pontos 2D das mãos de cada usuário presente na imagem.
2. Sabendo que o ator está sempre no meio da imagem, é aplicado um filtro para remover os falsos-positivos para os esqueletos dos outros participantes do experimento ou de outras pessoas que passaram na cena no momento da captura do conjunto de dados.
3. Selecionam-se as articulações mais importantes (braços, ombros e cabeça) do ator.
4. Usam-se os pontos das mãos para calcular o ponto central de cada uma delas.
5. Fornece-se a mesma imagem RGB como entrada para a função de segmentação para que seja extraído o ponto central do objeto.

Figura 34 – Resumo da etapa de extração e seleção de características.



Fonte: Próprio autor.

4.4.2 Incorporação das Características

Após a etapa de pré-processamento, as informações principais são representadas por: (*i*) cinco pontos 2D ($\mathbf{v}_h \in \mathbb{R}^{10\times1}$) da cabeça: 2 das orelhas, 2 dos olhos e 1 do nariz; (*ii*) informações do objeto por um ponto 2D ($\mathbf{v}_o \in \mathbb{R}^{2\times1}$); e (*iii*) pose do usuário (movimento) por nove pontos 2D ($\mathbf{v}_m \in \mathbb{R}^{18\times1}$), onde os sete primeiros pontos representam braços e ombros e os dois últimos os pontos centrais das mãos. Observe-se que o movimento, a cabeça e o objeto têm um número diferente de pontos, o que gera um vetor de características desbalanceado. Por esse motivo, o modelo pode considerar a informação do movimento mais importante que as outras. Assim, propõe-se equilibrar os dados de entrada do modelo por meio de uma estrutura de incorporação (*embedding*), de maneira que as características de movimento e contexto tenham a mesma dimensão. Além disso, para representar o contexto, as características que representam a cabeça e o objeto também foram definidas com a mesma dimensão e, portanto, possivelmente terão a mesma importância. Os processos de incorporação podem ser vistos na Figura 35 e são explicados abaixo.

1. Incorporando a informação do objeto $\mathbf{v}_o$: $\mathbf{e}_o = f(\mathbf{W}_o^T\mathbf{v}_o + \mathbf{b}_o)$, onde $\mathbf{e}_o \in \mathbb{R}^{16\times1}$, $\mathbf{W}_o \in \mathbb{R}^{2\times16}$ e $\mathbf{b}_o \in \mathbb{R}^{16\times1}$.

2. Incorporando a informação da cabeça $\mathbf{v}_h$: $\mathbf{e}_h = f(\mathbf{W}_h^T\mathbf{v}_h + \mathbf{b}_h)$, onde $\mathbf{e}_h \in \mathbb{R}^{16\times1}$, $\mathbf{W}_h \in \mathbb{R}^{10\times16}$ e $\mathbf{b}_h \in \mathbb{R}^{16\times1}$.

3. Incorporando a informação do movimento $\mathbf{v}_m$: $\mathbf{e}_m = f(\mathbf{W}_m^T\mathbf{v}_m + \mathbf{b}_m)$, onde $\mathbf{e}_m \in \mathbb{R}^{16\times1}$, $\mathbf{W}_m \in \mathbb{R}^{18\times16}$ e $\mathbf{b}_m \in \mathbb{R}^{16\times1}$.

4. Incorporando a informação de contexto ($\mathbf{e}_o$, $\mathbf{e}_h$): $\mathbf{e}_c = f(\mathbf{W}_c^T[\mathbf{e}_o,\mathbf{e}_h] + \mathbf{b}_c)$, onde $\mathbf{e}_c \in \mathbb{R}^{16\times1}$, $\mathbf{W}_c \in \mathbb{R}^{32\times16}$, e $\mathbf{b}_c \in \mathbb{R}^{16\times1}$ e $[\bullet]$ é um operador de concatenação de vetores.

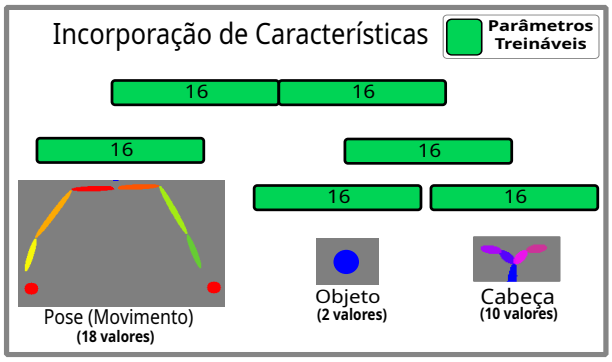
5. Criando o vetor de entrada contendo as informações incorporadas: $\mathbf{e}_{cm} = [\mathbf{e}_c, \mathbf{e}_m]$, onde $\mathbf{e}_{cm} \in \mathbb{R}^{32\times1}$.

Cada $\mathbf{W}_*$ e $\mathbf{b}_*$ representa pesos treinados pelo modelo, já $f(\bullet)$ representa a função de ativação ReLU. Assim, durante a fase de treinamento, o modelo aprende simultaneamente a classificar ações e a incorporar diferentes características de cada observação.

4.4.3 Modelo de classificação

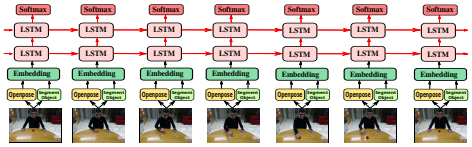
Ações distintas podem ter sequências de tamanhos distintos. Além disso, pode existir uma forte dependência entre observações de diferentes instantes de tempo. Sendo assim, como discutido na Seção 2.7, modelos que assumem a condição markoviana podem

Figura 35 – Processo de incorporação das características para cada observação (*embedding*). As conexões entre as articulações e a forma do objeto são mostradas apenas para fins de visualização. No entanto, apenas seus pontos são utilizados.



não capturar as longas dependências em uma sequência que representa a execução de uma ação. Portanto, decidiu-se por usar um modelo baseado numa RNN do tipo LSTM. A escolha do LSTM se deu não só por sua capacidade de capturar dependências em longas sequências, mas também por ter obtido bons resultados com o *star* $iRGB_{lstm}$ em relação à RNN padrão e à GRU (veja-se a Seção 3.4). Além disso, possibilitará uma comparação direta com os resultados apresentados em Schydlo et al. (2018), que também a utiliza como modelo de classificação.

Figura 36 – Arquitetura proposta para o modelo de classificação de ações.



A Figura 36 apresenta o modelo proposto, o qual é composto de duas camadas LSTM seguidas por um classificador *softmax*. Aqui, a primeira camada recebe como entrada no instante t: (i) o vetor resultante do processo de *embedding* $\mathbf{e}_{cm}^{(t)}$; (ii) o estado oculto $\mathbf{h}_1^{(t)} \in \mathbb{R}^{64\times1}$; e (iii) a célula $\mathbf{c}_1^{(t)} \in \mathbb{R}^{64\times1}$. A segunda camada recebe como entrada: (i) o estado oculto $\mathbf{h}_1^{(t)}$ resultante da primeira camada; (ii) o estado oculto $\mathbf{h}_2^{(t)} \in \mathbb{R}^{64\times1}$; e (iii) a célula $\mathbf{c}_2^{(t)} \in \mathbb{R}^{64\times1}$. Em seguida, uma camada totalmente conectada recebe como entrada o estado $\mathbf{h}_2^{(t)}$, aplica sobre o mesmo uma transformação linear usando uma matriz $\mathbf{p} \in \mathbb{R}^{64\times c}$ (onde c é o número de ações) e o normaliza usando a função *softmax*. Dessa forma, a saída do modelo é o vetor $\hat{\mathbf{s}}^{(t)}$ composto por c valores normalizados entre 0 e 1, onde cada valor está relacionado a uma possível ação. Com esse resultado e escolhendo um valor de limiar p para a probabilidade, ao utilizar a Equação (4.5), a antecipação pode

ser realizada seguindo o Algoritmo 1.

Result: *ação*
$ação \leftarrow -1$
$p \leftarrow limiar$
$M \leftarrow carregar\_modelo()$
$\mathbf{h}_1 \leftarrow zeros(64,1)$
$\mathbf{c}_1 \leftarrow zeros(64,1)$
$\mathbf{h}_2 \leftarrow zeros(64,1)$
$\mathbf{c}_2 \leftarrow zeros(64,1)$
while $ação = -1$ **do**
  $\mathbf{O} \leftarrow próxima\_imagem()$
  $\mathbf{f}_{cm} \leftarrow extrair\_características(\mathbf{O})$
  $\mathbf{e}_{cm} \leftarrow embedding(F_{cm})$
  $\hat{\mathbf{y}}, (\mathbf{h}_1,\mathbf{c}_1,\mathbf{h}_2,\mathbf{c}_2) \leftarrow M(\mathbf{e}_{cm}, (\mathbf{h}_1, \mathbf{c}_1, \mathbf{h}_2, \mathbf{c}_2))$
  $ação \leftarrow g(\hat{\mathbf{y}}, p)$
end

Algorithm 1: Algoritmo de antecipação de ação usado por modelos determinísticos durante a etapa de predição.

O modelo proposto é, por padrão, determinístico. No entanto, como mencionado anteriormente, este capítulo pretende mostrar como a antecipação pode ser efetivamente realizada usando a incerteza como entrada para uma função de limiar. Portanto, três versões do modelo proposto serão implementadas e testadas. Um LSTM Bayesiano utilizando *Bayes By BackProp* (BLSTM$_{BBB}$), um utilizando *MC Dropout* baseado em Gal e Ghahramani (2016b) (BLSTM$_{MC}$) e outro utilizando *Variational Dropout* (BLSTM$_{VD}$).

A incerteza é obtida executando-se a arquitetura da Figura 36 S vezes (simulação de Monte Carlo) usando a mesma observação $\mathbf{x}$ (que, no caso da Equação (4.1), é um exemplo $\mathbf{x}_t$). Portanto, como o modelo é estocástico, ele deve fornecer um valor diferente para cada uma das S predições. Assim, a informação mútua sobre as S predições fornece a incerteza epistêmica do modelo sobre a predição realizada para a observação $\mathbf{x}$. A qual pode ser calculada conforme apresentado na Equação (2.24). Essa métrica foi escolhida porque leva em consideração não apenas a entropia entre as classes (média sobre as S previsões), mas também a entropia média entre todas elas.

Para usar a informação mútua como limiar, a Equação (4.7) deve ser redefinida:

$$g(\hat{\mathbf{s}}^{(t)},u) = \begin{cases} arg\,max(m(\hat{\mathbf{s}}^{(t)})), & if\ h(\hat{\mathbf{s}}^{(t)}) < u \\ -1, & caso\ contrário \end{cases} \tag{4.15}$$

onde u é um valor de incerteza, h é a função de informação mútua (Equação (2.24)) e m é a média das previsões de S.

4.4.4 Experimentos

Todos os experimentos utilizaram o conjunto de dados *Acticipate*, apresentado na Seção 4.3. A partir dele, foram extraídos quatro tipos diferentes de dados usando o procedimento descrito na Seção 4.4: pontos da cabeça, posição do objeto, articulações dos braços e posições das mãos. Os pontos da cabeça e a posição do objeto formam a informação de contexto; articulações dos braços e posições das mãos formam uma pose, cuja evolução no tempo representa o movimento (também chamado aqui de entidade principal). Para melhor comparar os resultados e chegar a conclusões mais confiáveis sobre como cada fonte de informação influencia a antecipação da ação, decidiu-se realizar experimentos usando diferentes combinações de contexto (cabeça e objeto) e movimento, tanto para a versão original do conjunto de dados (6 ações), quanto para a sua versão estendida (12 ações).

4.4.4.1 Modelos e *baselines*

As principais abordagens apresentadas na Seção 4.1 precisam de um grande conjunto de dados para serem treinadas. Portanto, elas não são adequadas para serem usadas com o conjunto de dados Acticipate. Porém, para uma melhor comparação e discussão dos resultados da proposta, utilizou-se como principal *baseline* a proposta em Schydlo et al. (2018), que utiliza o conjunto de dados Acticipate. Além disso, mesmo tendo sido discutido que alguns modelos clássicos de natureza não sequencial não são ideais para a solução desse problema, uma vez que muitos dos modelos apresentados na Seção 4.1 possuem alta capacidade de aprendizado, para se obter resultados mais robustos sobre a qualidade da proposta, outros cinco modelos clássicos também serão utilizados como *baselines* adicionais para comparação: Naive Bayes (NB), Multilayer Perceptron (MLP), Rede Neural Convolucional 1D (CONV-1D), Máquina de Vetor de Suporte (SVM) e HMM. Para os primeiros quatro modelos, conforme mencionado anteriormente (Seção 4.4), utilizaram-se as sequências com um tamanho fixo (conforme ilustrado na Figura 7 da Seção 2.7), enquanto para o HMM, utilizaram-se as sequências com seus tamanhos originais.

Considerando a arquitetura dos modelos de *baselines*:

- O NB usa um modelo gaussiano para prever sua probabilidade condicional;
- O MLP é composto por apenas uma camada oculta;
- O CONV-1D tem três camadas convolucionais unidimensionais empilhadas, seguidas por uma MLP como classificador;
- O SVM implementa sua versão não linear por meio de um *kernel* RBF; e

Tabela 14 – Lista de todos os experimentos realizados. Os nomes $(BD)LSTM_*$ representam as versões determinísticas (D) e bayesianas (B) dos modelos LTSM propostos. $*$ representa a combinação de três tipos distintos de informações (movimento (m), cabeça (c) e objeto (o)), ou a técnica utilizada na implementação do modelo bayesiano (*MC dropout* (MC), *Bayes By Backprop* (BBB) e *Variational Dropout* (VD)).

Id	Nome do Modelo	Tipo	Movimento	Contexto		Conjunto de dados	
				Cabeça	Objeto	6 Ações	12 Ações
1	NB	*baseline*	✓	✓		✓	
2	CONV-1D	*baseline*	✓	✓		✓	
3	MLP	*baseline*	✓	✓		✓	
4	SVM	*baseline*	✓	✓		✓	
5	HMM	*baseline*	✓	✓		✓	
6	$DLSTM_{6m}$	Determinístico	✓			✓	
7	$DLSTM_{6mc}$	Determinístico	✓	✓		✓	
8	$DLSTM_{12m}$	Determinístico	✓				✓
9	$DLSTM_{12c}$	Determinístico		✓			✓
10	$DLSTM_{12o}$	Determinístico			✓		✓
11	$DLSTM_{12mc}$	Determinístico	✓	✓			✓
12	$DLSTM_{12mo}$	Determinístico	✓		✓		✓
13	$DLSTM_{12mco}$	Determinístico	✓	✓	✓		✓
14	$BLSTM_{MC}$	*MC Dropout*	✓	✓	✓		✓
15	$BLSTM_{VD}$	*Variational Dropout*	✓	✓	✓		✓
16	$BLSTM_{BBB}$	*Bayes By BackProp*	✓	✓	✓		✓

Fonte: Próprio autor.

- O HMM tem a sua probabilidade de emissão amostrada de uma distribuição gaussiana, a qual permite observações contínuas.

Os cinco modelos foram treinados com o conjunto de dados original (6 ações) e a sua versão estendida (12 ações). Para os experimentos com esses cinco modelos, não foi aplicada a técnica de *embedding* proposta. Assim, cada observação foi representada por um vetor com 18 valores correspondentes aos braços, dois valores para o objeto e 10 para os pontos da cabeça (veja-se a Seção 4.4.2). Cada modelo foi treinado e testado utilizando a combinação dos diferentes tipos de características (movimento, objeto e cabeça), totalizando-se assim 45 experimentos de *baselines* distintos. Porém, a título de esclarecimento, serão apresentados apenas os cinco mais conclusivos, um para cada modelo. Além disso, foram realizados mais 11 experimentos com diferentes versões do modelo proposto, onde os experimentos indicados como $DLSTM_*$ e $BLSTM_*$ correspondem respectivamente às variações dos modelos determinísticos e bayesianos realizados com as diferentes combinações dos dados de entrada. Os detalhes desses experimentos serão explicados mais adiante no texto. A Tabela 14 resume os 16 experimentos que serão apresentados.

Os experimentos 6 e 7 (conjunto de dados original) na Tabela 14 fornecem resultados que podem comparar a solução proposta com Schydlo et al. (2018) e com os cinco primeiros experimentos de *baseline*. Os experimentos de 8 a 13 fornecem resultados para mostrar o problema da ambiguidade entre as ações e a importância do contexto para a sua antecipação. Os últimos três experimentos fornecem resultados que mostram a importância da incerteza

para um modelo de antecipação. Por esse motivo, foram implementados três modelos bayesianos: *MC Dropout*, *Variational Dropout* e *Bayes By Backprop*. Para o *Variational Dropout*, como mencionado anteriormente, optou-se por usar α (Equação (2.19)) como um parâmetro treinável. Sendo assim, com esses três modelos, poder-se-á identificar qual aquele que melhor resolve o problema de antecipação de ações aqui abordado.

4.4.4.2 Configurações dos experimentos

Para cada imagem RGB em um vídeo, foi aplicado o procedimento de pré-processamento descrito na Seção 4.4. Todos os dados ausentes relacionados às cordenadas das articulações, mãos e posição do objeto foram definidos como sendo -1. Para avaliar a qualidade do modelo, o conjunto de dados foi estratificadamente dividido em 80% para treinamento e 20% para teste. Em cada experimento, uma validação cruzada de *10-folds* (rodadas) sobre o conjunto de treinamento foi realizada, onde nove *folds* foram usadas para treinar e uma para validar. Para cada *fold*, o processo de treinamento foi concluído quando a acurácia do reconhecimento (a acurácia alcançada avaliando apenas a predição da última observação da sequência) ao longo dos nove *folds* de treinamento nas cinco épocas anteriores foi superior a 98% (*earling stop*) ou quando as iterações excederam o número máximo de épocas.

Os 16 experimentos na Tabela 14 podem ser divididos em quatro categorias principais: *baselines* com 6 ações (de 1 a 5), determinísticos com 6 ações (6 e 7), determinísticos com 12 ações (de 8 a 13) e estocásticos com 12 ações (14, 15 e 16). Nos experimentos com os modelos determinísticos, diferentes configurações dos dados de entrada (movimento, cabeça e objeto) foram obtidas atribuindo -1 a $\mathbf{v}_m$, $\mathbf{v}_o$, e (ou) $\mathbf{v}_h$ em todo o conjunto de dados. Por exemplo, ao atribuir -1 a $\mathbf{v}_o$, o modelo considera apenas as informações de movimento ($\mathbf{v}_m$) e da cabeça ($\mathbf{v}_h$). Para cada um dos 16 experimentos, os hiperparâmetros foram escolhidos usando um processo de Otimização Bayesiana. Tais hiperparâmetros estão descritos nas Tabelas 15 e 16.

Tabela 15 – Lista de hiperparâmetros de cada experimento com os modelos de *baseline*.

Hiperparâmetro	NB	MLP	SVM	CONV-1D	HMM
Arquitetura	Gaussiana	Camada escondida (48)	*Kernel* (RBF)	Camadas Convolutionais (1x7x32, 1x5x32, 1x5x32) Camada escondida (64)	Estados (5)
Tamanho da sequência	72	64	64	96	64
Quantidade máxima de épocas	-	50	-	50	-
Tamanho do *batch*	-	32		32	-
Taxa de aprendizado (TA)	-	1e-3	-	1e-3	-
Otimizador	-	Adam	-	Adam	-

Fonte: Próprio autor.

Tabela 16 – Lista dos hiperparâmetros utilizados em cada experimento com os modelos propostos.

Hiperparâmetros	Configurações dos Modelos			
	$DLSTM_*$	$BLSTM_{MC}$	$BLSTM_{VD}$	$BLSTM_{BBB}$
Tamanho do *batch*	216	216	216	32
Truncamento da sequência	100	100	100	128
Tamanho da sequência	100	100	100	64
Quantidade máxima de épocas	100	100	100	200
Taxa de aprendizado (TA)	$1e-2$	$1e-2$	$1e-2$	$1e-2$
Decaimento da TA (por época)	1%	1%	1%	1%
Taxa de decaimento dos pesos	$1e-5$	$1e-5$	–	–
Truncamento do gradiente (Norma)	5,0	5,0	5,0	5,0
Dropout (Prob. de permanecer)	0,7	0,2	–	–
Otimizador	Adam	Adam	Adam	Adam

Fonte: Próprio autor.

4.4.5 Resultados e discussões

Para cada um dos 16 experimentos, após executar a Otimização Bayesiana sobre a validação cruzada de 10-*folds*, a melhor configuração de hiperparâmetros encontrada foi usada para treinar o modelo com o conjunto de treinamento completo. Em seguida, cada modelo foi testado com o conjunto de teste, que nunca foi visto pelo modelo durante o treinamento completo ou da validação cruzada. Os resultados do conjunto de teste foram plotados em gráficos e analisados cuidadosamente. Alguns desses gráficos são apresentados e discutidos nesta seção.

4.4.5.1 Modelos de *baseline* para o conjunto de dados original

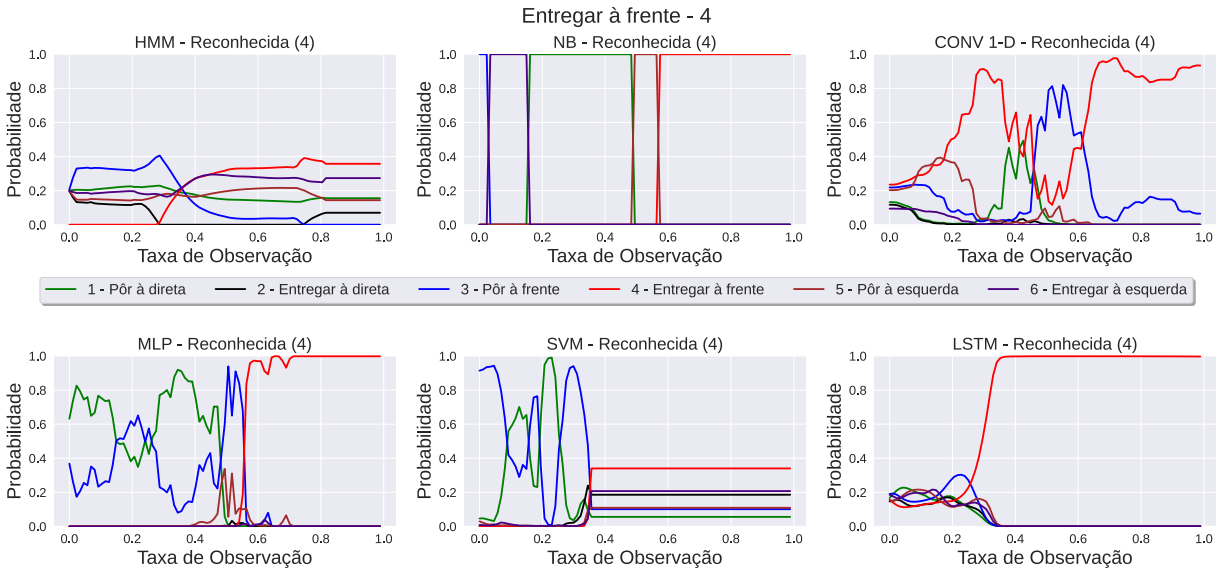
Tabela 17 – Comparação dos resultados obtidos pelos modelos (*baselines* e DLSTM) em diferentes taxas de observação. Os resultados para Schydlo et al. (2018) ([Schydlo]) foram fornecidos pelos autores. A subscrição $6mc$ indica que o modelo correspondente foi treinado com o conjunto de dados original (6 ações) usando movimento (m) e cabeça (c) como fonte de informação.

Nome dos Modelos	Percentual de observação (acurácia em %)									
	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
HMM	**19,05**	19,05	38,10	57,14	90,48	95,24	**100,0**	**100,0**	**100,0**	**100,0**
NB	**19,05**	09,52	09,52	09,52	19,05	33,33	71,43	95,24	95,24	95,24
$CONV$	09,52	14,29	19,05	19,05	33,33	47,62	42,86	61,90	61,90	80,95
MLP	**19,05**	19,05	28,57	28,57	33,33	47,62	71,43	90,48	85,71	95,24
SVM	**19,05**	19,05	19,05	19,05	28,57	71,43	95,24	95,24	95,24	95,24
$DLSTM_{6m}$	14,29	28,57	38,10	71,43	85,71	95,24	95,24	95,24	95,24	95,24
$DLSTM_{6mc}$	14,29	**38,10**	**47,62**	**90,48**	**95,24**	**100,0**	**100,0**	**100,0**	**100,0**	**100,0**
[*Schydlo*] (Pose 3D)	16,25	21,25	28,75	51,25	76,25	85,00	86,25	86,25	86,25	85,00
[*Schydlo*] (Pose 3D + Gaze)	15,00	16,25	40,00	73,75	86,25	97,50	96,25	92,50	91,25	87,50

Fonte: Próprio autor.

Como pode ser visto na Tabela 17, os modelos clássicos, usados como *baselines*, oferecem um resultado satisfatório quando analisada apenas a acurácia obtida em cada taxa de observação. Porém, como mencionado anteriormente, esse tipo de análise traz conclusões precipitadas sobre a capacidade do modelo de antecipar as ações. Um bom modelo de antecipação deve aumentar a diferenciação entre as classes à media que o número de observações também aumenta. Nesse sentido, é importante analisar os gráficos da Figura 37, onde é apresentada a distribuição de probabilidades entre as classes de cada modelo para cada taxa de observação de uma ação *entregar à frente*. Observe-se que os modelos de *baseline* (NB, HMM, MLP, CONV-1D e SVM) falham em acumular conhecimento sobre as ações executadas, ou mesmo, respondem com excesso de confiança (NB) sobre a predição. No HMM, a distribuição é quase uniforme, tornando difícil determinar uma probabilidade adequada para ser usada como um limiar de tomada de decisão. O Naive Bayes responde com alto excesso de confiança, mesmo no início da ação. Portanto, não é um modelo confiável para ser usado em tarefas de antecipação. Os outros três modelos, MLP, SVM e CONV-1D, são muito ruidosos e não fornecem confiança sobre a ação correta. Em contrapartida, o DLSTM proposto foi capaz de representar a evolução da ação, de forma a diferenciar claramente a ação correta das demais, à proporção que mais observações eram fornecidas.

Figura 37 – Evolução de uma ação *entregar à frente* com sua respectiva predição para os cinco modelos de *baseline* e o DLSTM proposto. Todos os modelos foram treinados no conjunto de dados original (6 ações) com informações do movimento e da cabeça.



Fonte: Próprio autor.

Para uma análise mais completa, a Tabela 18 traz a acurácia da antecipação de cada modelo calculado pela Equação (4.12). Como é possível observar, os modelos de *baseline*, mesmo sendo eficazes na tarefa de reconhecimento, falham quando aplicados à

Tabela 18 – Acurácia máxima de antecipação (segunda coluna) obtida por cada modelo (primeira coluna) ao aplicar o limiar de probabilidade (última coluna). Para atingir a acurácia correspondente, o modelo precisava, em média, de uma proporção de observação como a especificada na terceira coluna. Como o modelo NB é extremamente confiante em suas previsões, ele não é adequado à antecipação de ações. Os limiares de probabilidades apresentados são aqueles correspondentes ao melhor valor de antecipação dos seus respectivos modelos.

Modelo	Acurácia da Antecipação	Taxa Média de Observação Necessária	Limiar de Probabilidade
HMM	76,19%	0,45	0,38
NB	Não foi possível antecipar		
CONV-1D	47,62%	0,69	0,91
SVM	57,14%	0,55	0,68
MLP	61,90%	0,60	0,94
DLSTM	95,25%	0,40	0,90

Fonte: Próprio autor.

tarefa de antecipação. A acurácia máxima alcançada pelo HMM foi de 76,19% ao usar um limiar de 0,38. Com um limiar tão baixo, é difícil confiar neste modelo, uma vez que pelo menos uma das classes restantes pode atingir valores próximos (por exemplo, 0,37). O melhor entre os outros modelos de *baseline*, o MLP, alcançou apenas 61,90% com um limiar de 0,94. Enquanto o Naive Bayes não pôde sequer antecipar uma ação devido ao seu excesso de confiança. Já o DLSTM foi capaz de antecipar 95,25% das ações ao aplicar um limiar de 0,90. Todas essas evidências mostram a superioridade do modelo proposto em relação aos *baselines*. É importante observar que, como esperado, o LSTM acumula intrinsecamente o conhecimento de qual é a ação correta sem a necessidade de utilização de mecanismos extras. Diferentemente, os outros cinco modelos de *baselines*, mesmo utilizando o mecanismo de acúmulo de conhecimento descrito em Montalvao, Canuto e Carvalho (2018), não conseguiram melhorar a sua acurácia na antecipação.

Outro resultado importante é que a proposta, mesmo usando as articulações 2D do esqueleto extraídas de imagens, supera os resultados apresentados em Schydlo et al. (2018), que usou a fixação do olhar (*gaze)* e a pose 3D (Tabela 17) como entrada para o modelo. Com a informação de *movimento + cabeça* alcançou-se 90% de acurácia com menos de 40% das observações. Enquanto que o modelo dos autores atingiu 90% de acurácia após receber mais de 50% das observações. Em adição, conforme discutido antes, um modelo de antecipação eficaz também deve ser efetivo na tarefa de reconhecimento. Nesse sentido, o modelo $DLSTM_{6mc}$ reconheceu todas as ações na última observação (100% de acurácia média); enquanto o modelo dos autores alcançou um máximo de 97,5% utilizando 60% das observações e diminuiu para 87% com 100% das observações. Portanto, conclui-se que o modelo deles não reconhece todas as ações do conjunto de dados.

Além dos resultados acima, $DLSTM_{6mc}$ pode antecipar uma ação em média 3 *frames* antes que $DLSTM_{6m}$. Como os vídeos têm uma taxa de amostragem de $30Hz$, essa antecipação corresponde a aproximadamente $100ms$, o que pode equiparar-se aos $92ms$ apresentados em Schydlo et al. (2018) quando comparados os modelos que utilizavam *pose* e *pose + olhar*. Dessa forma, além de superar o desempenho alcançado por Schydlo et al. (2018), esta proposta conseguiu resolver o problema de reconhecimento de ações no conjunto de dados *Acticipate* e melhorar os resultados de antecipação de ações. O que corrobora ainda mais para a efetividade do modelo proposto.

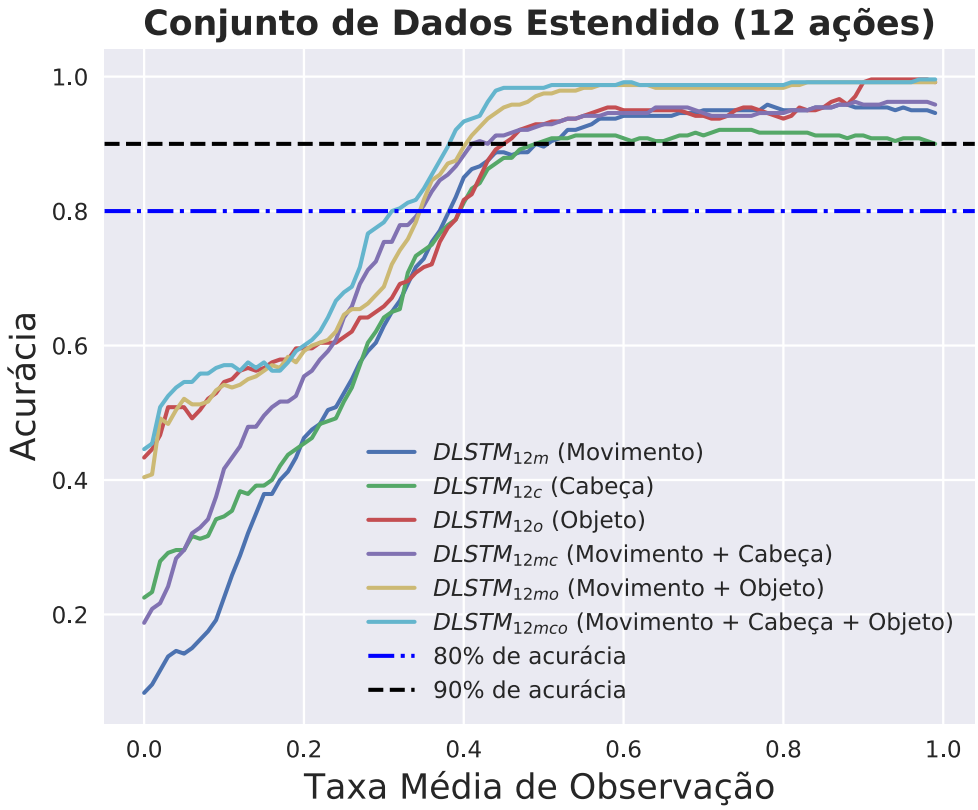
4.4.5.2 Modelos determinísticos aplicados ao conjunto de dados estendido

A Figura 38 mostra os resultados de 6 experimentos (de 8 a 13 da Tabela 14) usando a versão estendida do conjunto de dados. Pode-se observar que $DLSTM_{12m}$, $DLSTM_{12c}$ e $DLSTM_{12mc}$ não alcançaram 100% de acurácia na última observação. Este resultado mostra que eles não conseguiram separar as ações adequadamente mesmo ao usar as informações da cabeça. No entanto, as 6 novas ações são a única diferença entre $DLSTM_{6mc}$ e esses três modelos. Assim, quando o conjunto de dados foi dividido em mais ações, mais ambiguidades foram geradas entre elas. O que se alinha com a análise realizada na Seção 4.3.

Os modelos que usam a informação do objeto ($DLSTM_{12o}$, $DLSTM_{12mo}$ e $DLSTM_{12mco}$) foram capazes de reconhecer todas as ações. Além disso, eles alcançaram melhores resultados na tarefa de antecipação. Já na primeira observação, eles atingiram mais de 40% de acurácia e o modelo com a informação completa do contexto ($DLSTM_{12mco}$) atingiu em média 98% de acurácia após uma taxa média de observação de 0,44. Portanto, ações com movimentos semelhantes podem ser melhor distinguidas quando informações de contexto são utilizadas. Além disso, percebe-se como esse último modelo conseguiu extrair informações relevantes da posição do objeto e dos pontos da cabeça. Embora o objeto seja representado por apenas dois valores (um ponto bidimensional), assim como foi suposto, ele forneceu informações significativas sobre as ações ao modelo. Isso mostra a eficácia do processo de incorporação de características proposto na arquitetura do modelo.

Ao utilizar as informações do objeto, o modelo antecipou algumas ações após apenas algumas observações. Para uma melhor visualização, a Figura 39 ilustra a acurácia obtida pelos 6 modelos anteriores quando considerando apenas as 6 novas ações adicionadas (*receber* e *pegar* (*esquerda*, *meio*, *direita*)). Os modelos que usaram a informação do objeto começaram com uma acurácia média de classificação maior que 65% e atingiram os 90% após observar uma média de 10% dos *frames*. Sendo que o melhor modelo atingiu 95% de acurácia com menos de 5% das observações, em média. Em termos de *frames*, para o conjunto de dados *Acticipate*, isso corresponde a uma média de 4 *frames*. Esses resultados corroboram a afirmação anterior sobre a importância das informações do objeto para essas

Figura 38 – Resultados para modelos determinísticos no conjunto de dados estendido (12 ações). Cada experimento utilizou o mesmo modelo, porém, treinado com diferentes dados de entrada.

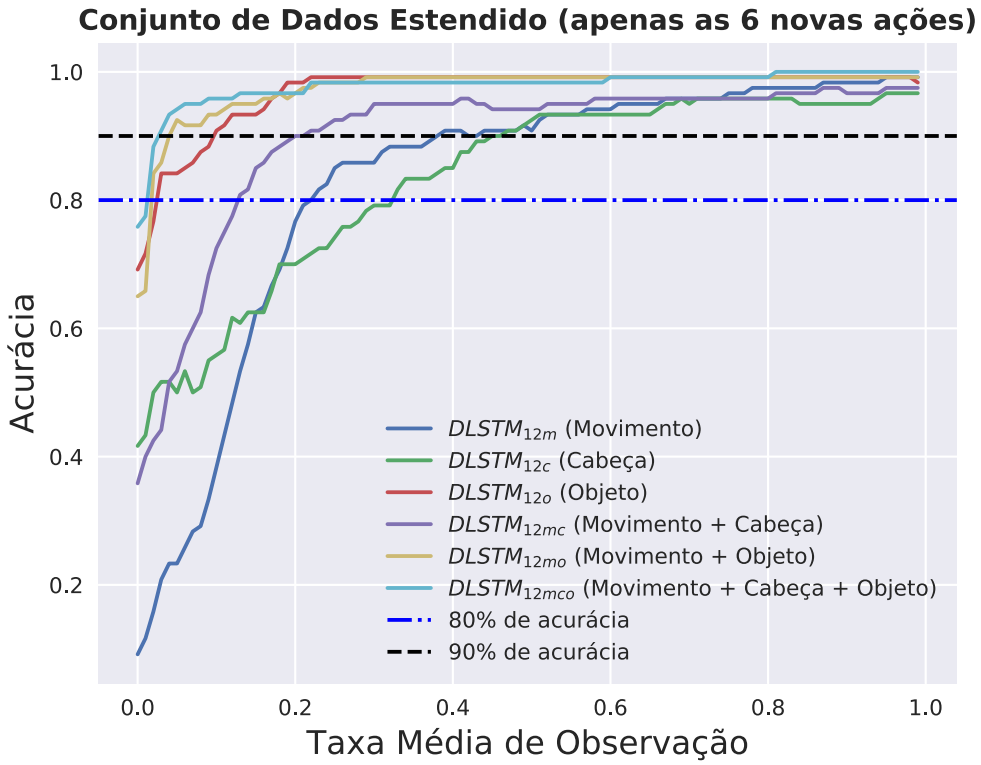


Fonte: Próprio autor.

6 novas ações.

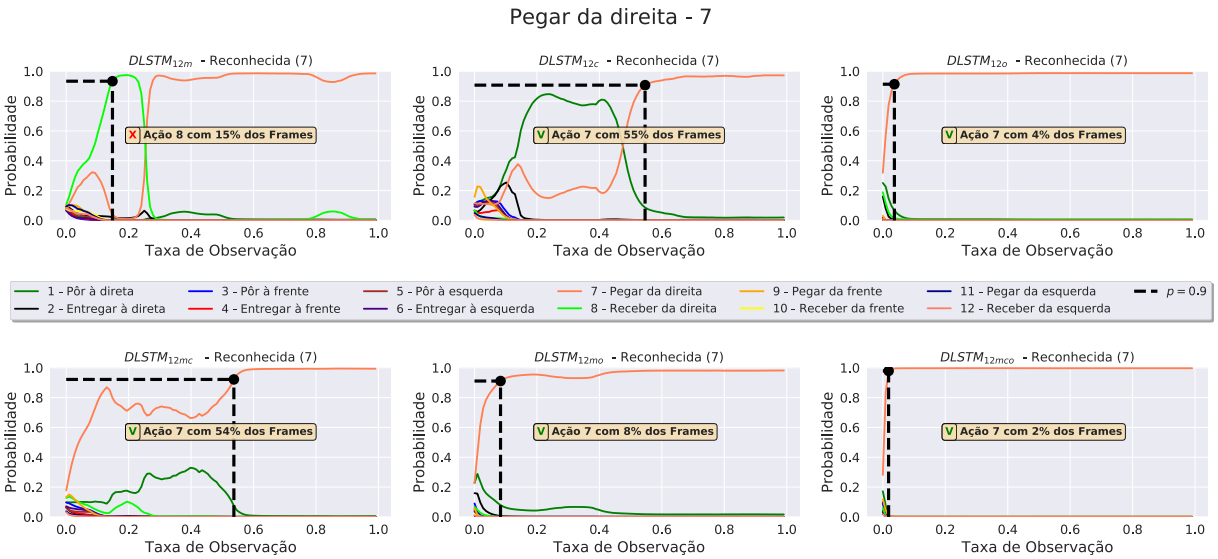
Para medir a acurácia da antecipação, utilizou-se a Equação (4.12) com o limiar $p = 0{,}9$ para os mesmos 6 modelos determinísticos. A Figura 40 apresenta a evolução de uma ação *pegar da direita* depois de passar pelos 6 modelos. Os gráficos ilustram como o modelo que usa apenas movimento ($DLSTM_{12m}$) equivocou-se em sua antecipação. Esse equívoco pode ter sido causado pelo excesso de confiança do modelo ao antecipar ações ambíguas. Outros modelos, aqueles que usam informações de contexto parciais/completas, anteciparam a ação corretamente. Observe-se que o modelo com contexto completo (*cabeça + objeto*) antecipou a ação depois de observar apenas 2% da sequência de dados (2 *frames* em seu respectivo vídeo). Outro resultado interessante é que os modelos confundiram as classes, assim como foi suposto na Seção 4.3. Analisando os vídeos, percebe-se que a ação *pegar da direita* tem um movimento semelhante às ações *pôr à direita, entregar à direita* e *receber da direita*; além de uma direção do olhar semelhante à ação *pôr à direita*. Portanto, $DLSTM_{12m}$ confundiu *pegar da direita* com *receber da direita* e $DLSTM_{12c}$ não tinha certeza sobre se a ação era *pegar da direita* ou *ou pôr à direita*. Essas características aparecem na quase totalidade das predições.

Figura 39 – Resultados para os modelos determinísticos no conjunto de dados estendido apenas para as 6 novas ações



Fonte: Próprio autor.

Figura 40 – Evolução da amostra de uma ação *pegar da direita* para os seis modelos determinísticos.

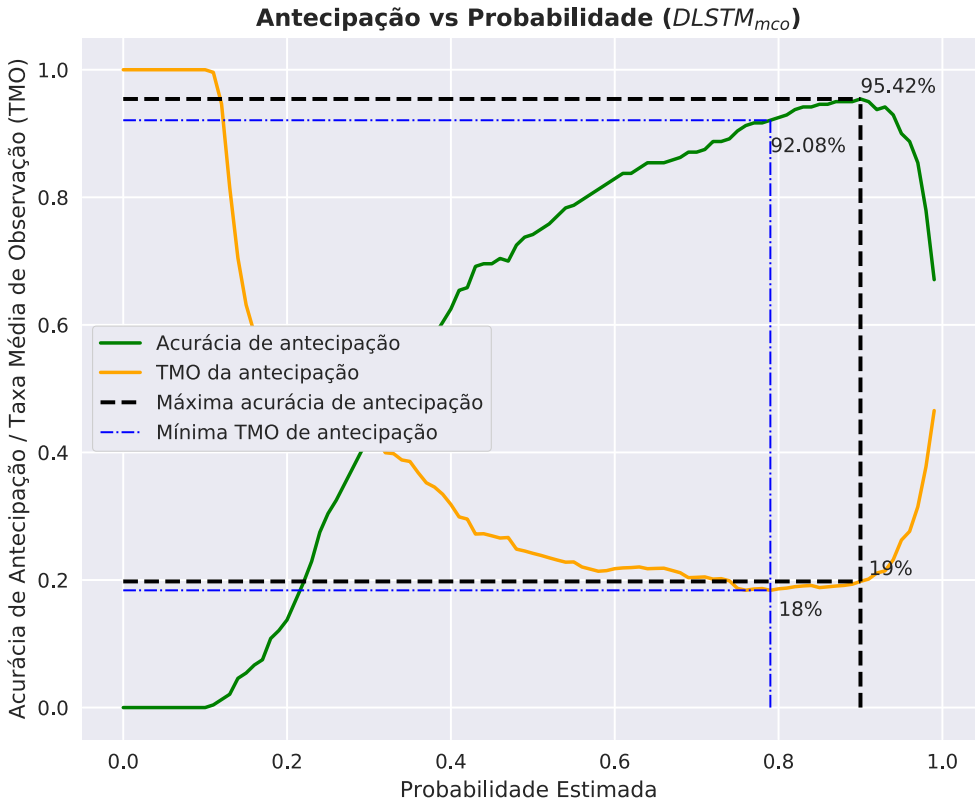


Fonte: Próprio autor.

Para destacar o compromisso entre a acurácia de um determinado limiar e a antecipação, o gráfico na Figura 41 apresenta a variação da acurácia da antecipação e

a porcentagem de observações quando da variação do limiar (p). No gráfico, vê-se que quando $p = 0{,}9$, $DLSTM_{12mco}$ pôde antecipar corretamente 95,42% das ações usando, em média, 19% das sequências dos vídeos. Como em média cada ação no conjunto de dados possui 79 imagens, essa taxa média de observação de 19% corresponde a uma média de 15 *frames* de um vídeo. Por outro lado, caso se queira determinar o número mínimo de observações necessárias para uma antecipação satisfatória, com $p = 0{,}8$, o modelo anteciparia corretamente 92,08% de ações, usando, em média, 18% da sequência de observação (14 *frames* em média).

Figura 41 – Variação da acurácia da antecipação e da taxa média de observação para o limiar que utiliza o valor da probabilidade estimada.

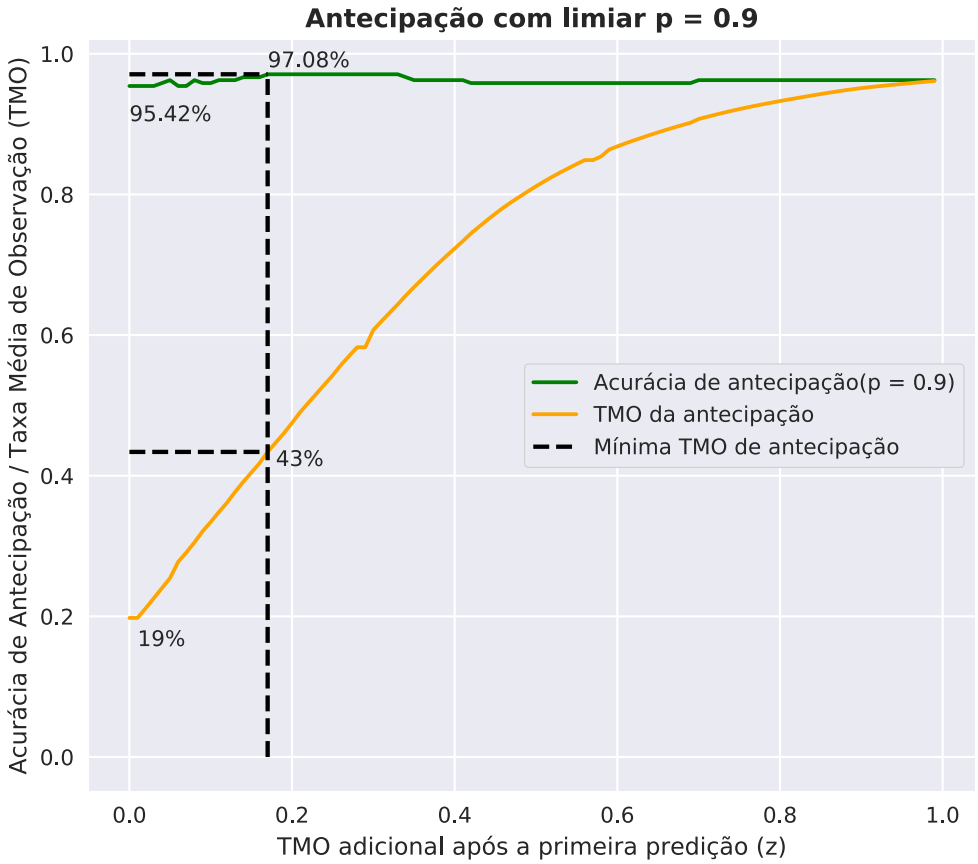


Fonte: Próprio autor.

Com $p = 0{,}9$, não é desejável que a função g (Equação (4.7)) tenha um excesso de confiança na predição do modelo, pois isso pode acabar por gerar muitos falsos-positivos na antecipação. Assim, conforme discutido na Seção 4.3, uma possível solução para reduzir o número de falsos-positivos é forçar o modelo a aguardar mais z observações para reafirmar sua predição. O problema dessa abordagem é que z é um novo hiperparâmetro que pode influenciar diretamente a efetividade da antecipação e deve ser escolhido cuidadosamente. Na Figura 42, é ilustrado como a acurácia da antecipação e a taxa média de observação variam em relação a z, onde z é a taxa média de observação adicional após a primeira antecipação usando o limiar $p = 0{,}9$. A melhor acurácia de antecipação (97,02%) é alcançada quando $z = 0{,}18$. Em outras palavras, o modelo precisa aguardar, em média,

mais 18% das observações para obter uma acurácia de antecipação de 97,02%. Comparando com os resultados anteriores, o ganho de menos de 2% em acurácia custa um aumento no tempo de antecipação maior que o dobro das observações necessárias (passando de 19% para 43%). Além disso, a taxa média de observação mínima necessária para antecipar qualquer ação, agora, é de 18%. Mesmo para ações menos ambíguas, como as apresentadas anteriormente na Figura 32 e aquelas usadas na Figura 40. Portanto, além do fato da escolha de z inserir um novo *trade-off* no projeto (taxa média de observação *vs* acurácia), ele não fornece uma maneira eficaz de melhorar a tarefa de antecipação de ação.

Figura 42 – Variação da acurácia da antecipação e taxa média de observação com taxa adicional de observação após a antecipação. Se, no tempo t, a probabilidade máxima exceder o valor de 0,9, o modelo deverá aguardar mais z observações a fim de confirmar a sua predição.



Fonte: Próprio autor.

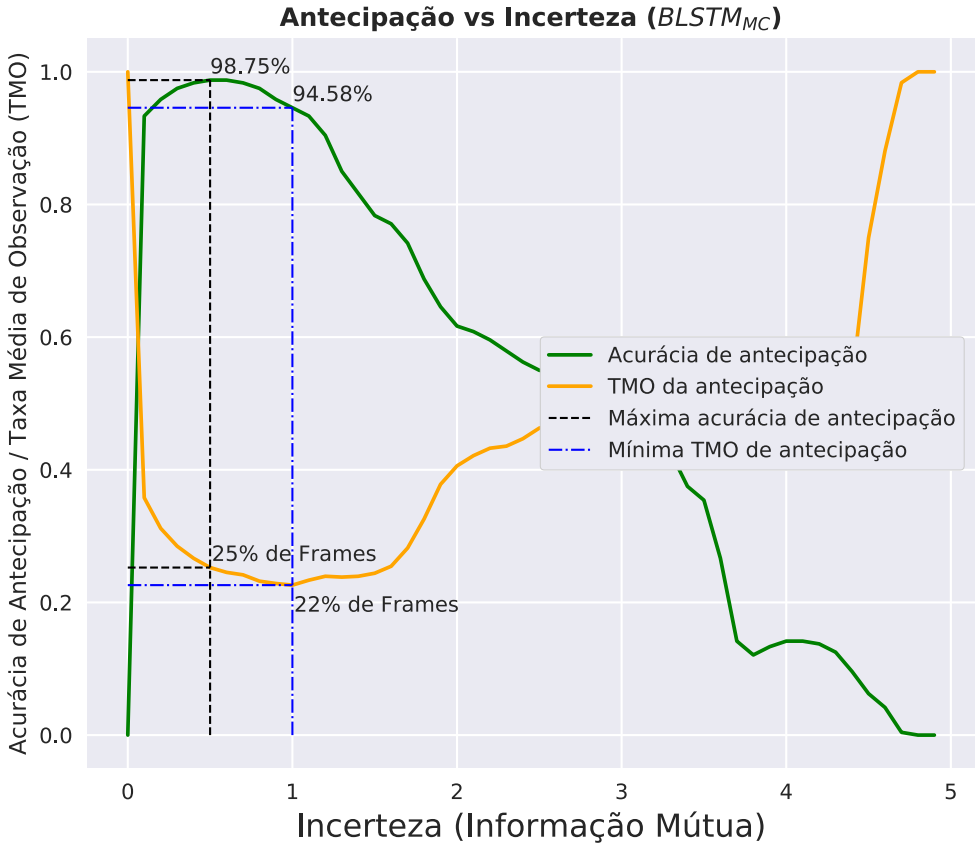
4.4.5.3 Modelos estocásticos

Os resultados dos modelos bayesianos $BLSTM_{MC}$, $BLSTM_{VD}$ e $BLSTM_{BBB}$ serão comparados com os resultados de $DLSTM_{12mco}$, o melhor modelo determinístico para o conjunto de dados estendido. Durante o tempo de predição, cada modelo bayesiano foi alimentado 50 vezes com a mesma observação $\mathbf{x}_t$, o que corresponde a uma simulação

de Monte Carlo com $S = 50$. Em seguida, aplicando a Equação (2.24) sobre as S predições, mediu-se a incerteza epistêmica de cada predição dos modelos em relação à observação $\mathbf{x}_t$. Por fim, por meio da aplicação da Equação (4.15), as ações foram antecipadas utilizando-se apenas os valores calculados da incerteza.

Os três modelos bayesianos também reconheceram todas as ações no conjunto de dados estendido. Além disso, eles obtiveram melhores resultados que o $DLSTM_{12mco}$ na acurácia da antecipação, mesmo quando este utilizava uma taxa média de observação extra de $z = 0{,}18$. Aplicando o mesmo procedimento da Figura 41, pode-se escolher um valor de limiar a ser usado em cada modelo. Assim, para cada um deles, o limiar de antecipação foi escolhido analisando a variação da acurácia da antecipação e o tempo médio de observação com a variação do valor de incerteza. A Figura 43 mostra essa comparação para $BLSTM_{MC}$.

Figura 43 – Variação da acurácia da antecipação e da taxa média de observação média utilizando um limiar de incerteza para o modelo baseado no *MC dropout*



Fonte: Próprio autor.

A Tabela 19 compara os resultados dos modelos bayesianos com o melhor modelo determinístico ($DLSTM_{12mco}$). Observe-se que $BLSTM_{MC}$ atinge a melhor acurácia na tarefa de antecipação (98,75%) usando o limiar de incerteza $u = 0{,}5$. No entanto, $BLSTM_{VD}$ e $BLSTM_{BBB}$ também alcançam resultados satisfatórios: com $u = 0{,}5$,

$BLSTM_{VD}$ alcançou 98,33% de acurácia de antecipação e, com $u = 0{,}3$, $BLSTM_{BBB}$ atingiu 97,08%.

Tabela 19 – Resultados obtidos pelos modelos estocásticos em comparação ao melhor modelo determinístico.

Nome do Modelo	Parâmetro	Acurácia da Antecipação	Taxa Média de Observação
$DLSTM_{12mco}$	$p = 0{,}9$ / $z = 0{,}0$	95,42%	19%
$DLSTM_{12mco}$	$p = 0{,}79$ / $z = 0{,}0$	92,08%	**18%**
$DLSTM_{12mco}$	$p = 0{,}9$ / $z = 0{,}18$	97,08%	43%
$BLSTM_{MC}$	$u = 0{,}5$	**98,75%**	25%
$BLSTM_{MC}$	$u = 1{,}5$	94,58%	22%
$BLSTM_{VD}$	$u = 0{,}5$	98,33%	26%
$BLSTM_{VD}$	$u = 1{,}5$	85,42%	20%
$BLSTM_{BBB}$	$u = 0{,}3$	97,08%	25%
$BLSTM_{BBB}$	$u = 1{,}3$	93,33%	20%

Fonte: Próprio autor.

Considerando o número mínimo de observações necessárias para aumentar a acurácia de antecipação, $DLSTM_{12mco}$ forneceu o melhor resultado. Em média, ele precisa receber 18% de observações para obter uma acurácia de antecipação de 92,08%. No entanto, mesmo que precise de menos observações, ele sofre uma redução na acurácia da antecipação de 95,42% com $p = 0{,}9$ para 92,08% com $p = 0{,}79$. Em resumo, o melhor modelo alcançou 98,75% após observar em média 25% de uma ação ($BLSTM_{MC}$). Um aumento de 6,67% em acurácia, com um custo médio de apenas 7% em observações extras. Resultado superior que o obtido quando usando $z = 0{,}18$ no modelo determinístico, em que um aumento menor que 2% custou, em média, 24% em observações extras. Além disso, para realizar a antecipação, não foi necessário escolher mais um hiperparâmetro, apenas o limiar de incerteza u.

Como mencionado nas seções anteriores, para a interação humano-máquina, o modelo deve não apenas ter um curto tempo de antecipação, mas também ser preciso em sua predição. Para o $BLSTM_{MC}$ atingir seu melhor valor de predição, ele necessita de 25% de observações, o que, de fato, não representa uma grande quantidade de amostras. Por exemplo, em um sistema baseado em imagens amostradas a 30Hz (câmeras comuns), uma ação que dure $2s$ seria antecipada depois de decorrido, em média, $0{,}5s$ do seu primeiro *frame*. Em outras palavras, ele poderia antecipar uma ação depois que o sistema observasse, em média, 15 *frames*. Portanto, já que o modelo pode ser considerado preciso em sua predição, o sistema possui cerca de $1{,}5s$ para tomar uma decisão correta.

Finalmente, foi possível ver que o modelo proposto superou a referência utilizada, Schydlo et al. (2018), mesmo usando informações menos precisas (pose 2D vs 3D e articulações da cabeça vs olhar). Em complemento, embora os resultados apresentados

tenham sido adquiridos em um pequeno conjunto de dados colaborativo, a proposta apresentada pode ser usada para resolver tarefas de antecipação mais complexas. Para isso é necessário analisar quais as informações de contexto seriam mais adequadas para conjuntos de dados mais gerais. Dessa forma, a antecipação de gestos no conjunto de dados Montalbano é uma aplicação que pode validar a proposta para um problema significativamente mais complexo que o abordado aqui com o Acticipate.

4.5 Proposta 2: aplicação no problema de antecipação de gestos

Como pôde ser visto no capítulo 3, mesmo que a proposta do *Strar* $iRGB_{lstm}$ possa ser utilizada no reconhecimento iterativo de gestos, ela não consegue antecipá-los. Outra observação feita é que muitos dos gestos do conjunto de dados Montalbano possuem parte significativa de informação contida no formato que cada uma das mãos assume durante a execução dos mesmos. Dessa maneira, a fim de aplicar os conhecimentos adquiridos neste capítulo ao problema de antecipação de gestos, propõe-se o seguinte:

- **Movimento:** Utilizar o *Strar iRGB* como forma de representação iterativa do movimento contido em vídeos do conjunto de dados Montalbano.
- **Contexto:** Utilizar a informação das mãos como informação de contexto.
- **Fusão:** Utilizar o *soft-attention* para fundir as informações de movimento e de contexto.
- **Classificação:** Utilizar uma LSTM bayesiana para poder classificar cada entrada e fornecer a incerteza da predição.

4.5.1 Descrição da proposta

4.5.1.1 Extração da informação de movimento

Este processo se dará da mesma forma que na Seção 3.4. Inclusive, serão utilizados os mesmos hiperparâmetros $\alpha = 0.6$ e $N = 5$. Dessa forma, assim como para o *Strar* $iRGB_{lstm}$, o extrator utilizado será uma *Resnet 50* seguida de um operador de convolução unidimensional com dimensões $\mathbb{R}^{3\times1}$. Dessa forma, a saída deste processo será um vetor $\mathbf{e}_m \in \mathbb{R}^{1023\times1}$.

4.5.1.2 Extração da informação de contexto

Após a análise realizada e apresentada nos comentários do Capítulo 3, Seção 3.5, percebeu-se a importância da informação das mãos para o reconhecimento de gestos utilizando o conjunto de dados Montalbano. Sendo assim, a partir do que foi visto na proposta da Seção 4.4, pode-se considerar que a forma das mãos é uma informação de contexto que possivelmente melhorará os resultados dos modelos de reconhecimento sobre tal conjunto de dados. Dessa maneira, a proposta aqui é utilizar as juntas do esqueleto 2D de cada indivíduo para poder extrair as regiões na imagem que contenham as suas mãos direita e esquerda. Apesar do Montalbano disponibilizar a informação de esqueleto capturada com o sensor *Kinect*, como uma das premissas deste trabalho é utilizar apenas informações extraídas de imagens RGB, propõe-se utilizar o *Openpose* para extrair as informações do esqueleto do indivíduo em cada imagem do vídeo. Como no Montalbano cada imagem contém apenas uma pessoa, não será necessário realizar uma etapa de filtragem, como feito na Seção 4.4.2.

Após extrair o esqueleto de uma imagem, utilizam-se as juntas que representam os punhos de ambas as mãos para poder extrair duas regiões de 100×100 píxeis. Dessa forma, as duas regiões recortadas são concatenadas horizontalmente, resultando em uma imagem 100×200 que representa a informação de contexto para o gesto em questão.

Assim como para o movimento, o extrator utilizado na imagem contendo o recorte das mãos será uma *Resnet 50*, seguida também de um operador de convolução unidimensional com dimensões $\mathbb{R}^{3\times1}$. Dessa forma, a saída deste processo será um vetor $\mathbf{e}_c \in \mathbb{R}^{1023\times1}$

4.5.1.3 Fusão das informações de movimento e de contexto

Note-se que tanto o movimento quanto o contexto, ambos são representados por vetores de características com as mesmas dimensões. Sendo assim, devido à sua efetividade apresentada nos resultados do modelo *Star* $RGB_{SoftAtt}$, propõe-se utilizar um operador *soft-attention* (veja-se Seção 3.3.2.2) a fim de fundir as características extraídas pelas duas CNNs (características das mãos e da imagem *Star iRGB*). Com isso, a cada observação, será possível ponderar a contribuição de cada fonte de informação (movimento e contexto) para a antecipação dos gestos. O resultado desse operador será um vetor $\mathbf{e}_{soft} \in \mathbb{R}^{1023\times1}$ contendo a média ponderada entre $\mathbf{e}_m$ e $\mathbf{e}_c$.

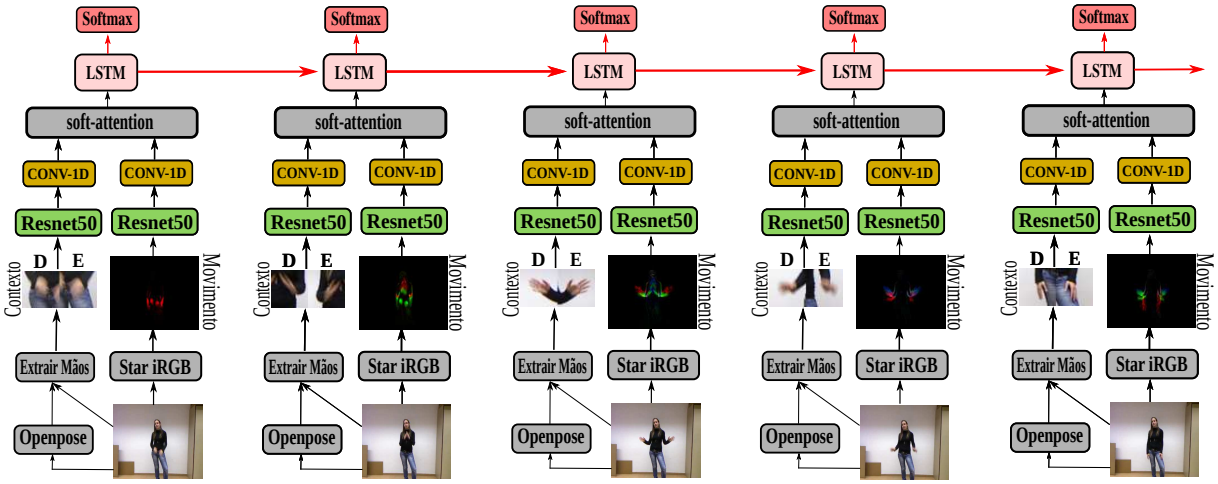
4.5.1.4 Modelo de classificação

Devido aos resultados alcançados em experimentos anteriores, propõe-se utilizar como classificador uma rede recorrente do tipo LSTM formada por apenas uma camada

com estados ocultos de 1024 neurônios. A classificação se dará por uma rede totalmente conectada com 20 neurônios de saídas que ativam uma função *softmax*. Para poder capturar a incerteza da predição, a LSTM será bayesiana. Isso se dará por meio da técnica de *MC Dropout*.

Um dos objetivos desta proposta é melhorar o reconhecimento de gestos por meio da informação de contexto extraída das mãos. Além disso, objetiva-se a possibilidade de antecipação dos gestos do conjunto e dados Montalbano. Uma vez que, nesse sentido, apenas dois resultados foram apresentados até o momento ((GUPTA et al., 2019),(MOLCHANOV et al., 2016)). Na Figura 44, pode ser vista uma representação completa do processo de reconhecimento de gesto proposto nesta seção, o qual é nomeado *BStar iRGB*$_{hand}$, onde o *B* em *BSTAR* corresponde à sua versão bayesiana e *hand* à informação de contexto utilizada, que, neste caso, é a informação das mãos.

Figura 44 – Representação do processo completo do *BStar iRGB*$_{hand}$ para vários instantes de tempo.



Como principal *baseline* serão utilizados os resultados do *Star iRGB*$_{lstm}$. Além disso, será implementada uma versão determinística do *BStar iRGB*$_{hand}$, o qual será chamado de *DStar iRGB*$_{hand}$. Em complemento, como essa proposta pode ser vista como uma arquitetura de fluxo duplo (movimento e contexto), para poder demonstrar a efetividade da priori assumida sobre a informação de contexto, outro *baseline* será um modelo que utiliza a imagem RGB completa como informação de contexto, ao invés apenas dos recortes das mãos. Esse modelo também será estocástico e terá o nome de *BStar iRGB*$_{image}$.

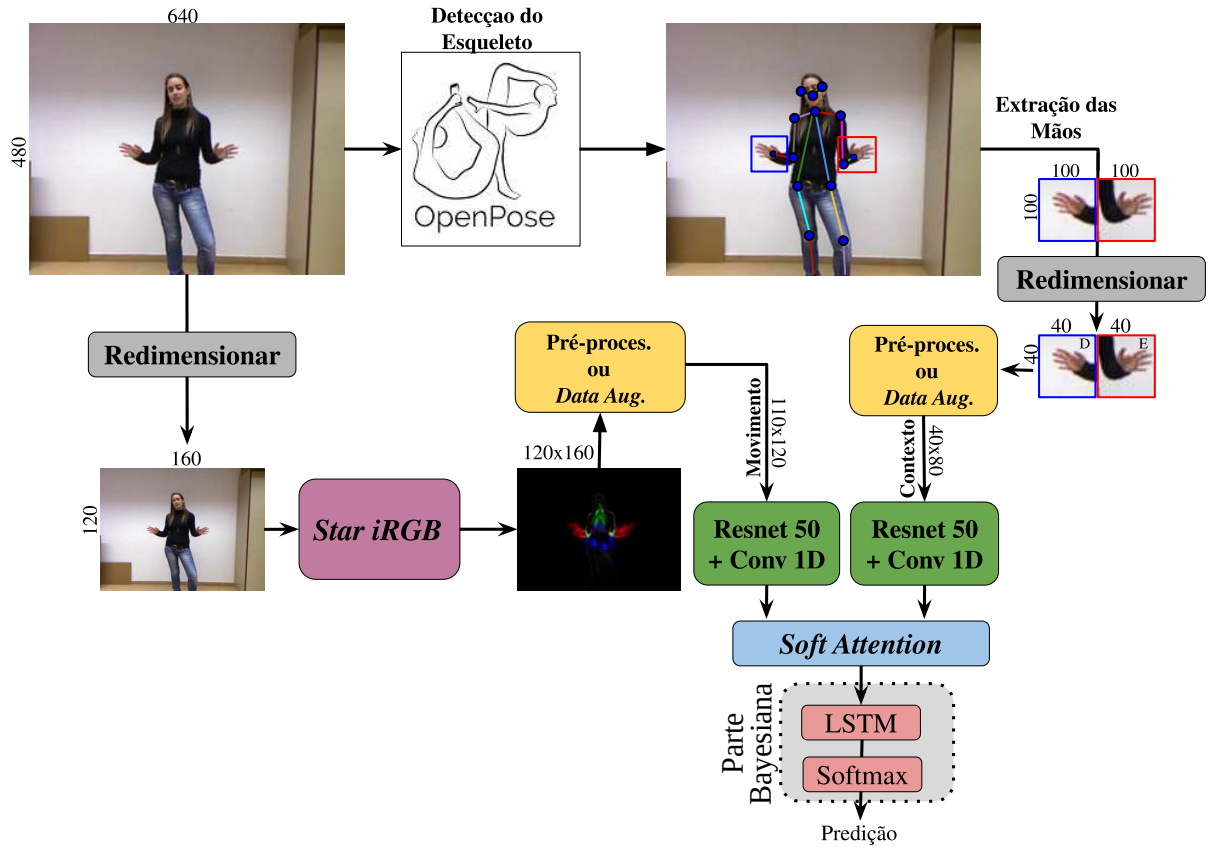
4.5.2 Experimentos

Como mencionado anteriormente, a base de dados utilizada será o Montalbano. Dessa forma, para o modelo proposto (*BStar iRGB*$_{hand}$) cada imagem presente em um clipe de vídeo que representa um gesto gerará uma representação *Star iRGB* e uma imagem

contendo as duas mãos do indivíduo. Para a obtenção dessa última, a imagem original alimentará o modelo *Openpose*; o esqueleto será extraído e as subimagens contendo as duas mãos serão recortadas e concatenadas. Apesar da imagem resultante ter dimensões 100×200 píxeis, para diminuir o tempo de processamento do modelo, a mesma foi redimensionada para 40×80. A representação *Star iRGB* continuou com dimensões 120×160. Para o modelo *Star* $iRGB_{image}$, a imagem original foi redimensionada para 120×160. Dessa forma, o mesmo processo de *data augmentation* aplicado na imagem *Star iRGB* pôde ser aplicado à imagem original, o qual é o mesmo aplicado para o treinamento do *Star* $iRGB_{lstm}$ (Seção 3.4).

Para o *BStar* $iRGB_{hand}$, o processo de *data augmentation* aplicado sobre a imagem *Star iRGB* também foi o mesmo. Porém, para não perder informações relevantes, nas imagens das mãos, apenas a operação de recorte aleatório não foi realizada. Assim, enquanto o processo de *data augmentation* resulta em uma imagem *Star iRGB* com dimensões 110×120 píxeis, a imagem das mãos continua com as mesmas dimensões, 40×80. A Figura 45 ilustra o processo completo do *BStar* $iRGB_{hand}$ em um instante de tempo qualquer.

Figura 45 – Representação do processo completo do *BStar* $iRGB_{hand}$ em um instante de tempo qualquer. Note-se que a caixa amarela representa tanto o pré-processamento durante a etapa de predição, quanto a operação de *data agmentation* durante o treinamento.



Os procedimentos para encontrar os hiperparâmetros e os ambientes de *software* e de *hardware*, utilizados no treinamento, foram os mesmos utilizados para o *Star* $iRGB_{lstm}$ (Seção 3.4). A Tabela 20 apresenta os hiperparâmetros utilizados no treinamento dos modelos que obtiveram os melhores resultados.

Tabela 20 – Lista dos hiperparâmetros utilizados para o treinamento dos modelos. Para a taxa de aprendizado TA são apresentados [TA1, TA2] correspondentes às TAs utilizadas para treinar os extratores e o classificador, respectivamente.

Hiperparâmetros	Modelos		
	BStar $iRGB_{hand}$	*DBStar* $iRGB_{hand}$	*BStar* $iRGB_{image}$
Geração do *Star iRGB*			
Termo de amortização (α)	0,6	0,6	0,6
Tamanho da janela (N)	5	5	5
Treinamento			
Tamanho do *batch*	32	32	32
Truncamento da sequência	24	24	24
Tamanho da sequência	12	12	12
Quantidade máxima de épocas	100	100	100
Taxa de aprendizado TA ([TA1, TA2])	[1e-4, 1e-3]	[1e-4, 5e-3]	[1e-4, 1e-3]
Decaimento da TA (por época)	1%	1%	1%
Taxa de decaimento dos pesos	1e-5	1e-5	1e-5
Truncamento do gradiente (Norma)	5,0	5,0	5,0
Dropout	0,30	0,20	0,30
Otimizador	Adam	Adam	Adam

Fonte: Próprio autor.

4.5.3 Resultados e discussões

4.5.3.1 Reconhecimento

Após o treinamento utilizando o conjunto de treino do Montalbano, o modelo *BStar* $iRGB_{hand}$ obteve uma acurácia média de reconhecimento na última observação de 97,46% sobre o conjunto de teste. Esse valor é 2,57% maior que o obtido em *Star* $iRGB_{lstm}$ e 2,88% maior que o do *Star* $RGB_{SoftAtt}$, quando treinados e testados utilizando o mesmo conjunto de dados. A Tabela 21 apresenta, além desses, os resultados dos outros dois modelos de *baseline*, referentes a uma versão determinística da proposta (*DStar* $iRGB_{hand}$) e uma versão da proposta utilizando a imagem completa como fonte de informação contextual (*BStar* $iRGB_{image}$). Note-se que o modelo proposto, bem como a sua versão determinística, alcançou uma acurácia média entre classes maior que os outros dois *baselines*. O que mostra a efetiviade da proposta. Note-se ainda que o *BStar* $iRGB_{image}$ foi o que alcançou o pior resultado (94,18%), até pior que o do *Star* $RGB_{SoftAtt}$ (94,58%). Esse resultado mostra a importância da utilização do conhecimento a priori sobre as informações de

contextos mais relevantes para o problema. E que a proposta aqui apresentada, mesmo utilizando menos informação, pode ser tão efetiva quanto as de fluxo duplo.

Tabela 21 – Acurácia obtida pelo modelo proposto e os quatro modelos de *baseline* sobre o conjunto de testes do Montalbano.

Modelo	Acurácia média
Star $RGB_{SoftAtt}$	94,58%
Star $iRGB_{lstm}$	94,89%
BStar $iRGB_{image}$	94,18%
DStar $iRGB_{hand}$	96,95%
BStar $\boldsymbol{iRGB_{hand}}$	**97,46%**

Fonte: Próprio autor.

Para uma visão mais geral sobre os resultados alcançados no conjunto de dados Montalbano até o momento, a Tabela 22 apresenta a acurácia obtida pelos modelos *BStar* $iRGB_{hand}$, *Star* $iRGB_{lstm}$ e *Star* $RGB_{SoftAtt}$ para cada classe de gestos, bem como as diferenças entre os resultados do *BStar* $iRGB_{hand}$ em realção a ambos os outros modelos. Como pode ser visto, em comparação ao *Star* $iRGB_{lstm}$, o modelo proposto melhorou a acurácia de reconhecimento para quase a totalidade das classes, sendo que apenas para a classe *fame* os dois modelos obtiveram o mesmo resultados. Em relação ao *Star* $RGB_{SoftAtt}$, o modelo proposto foi ligeiramente pior nos resultados de apenas três classes: *tantotempo* (−0,58%), *fame* (−1,08%) e *perfetto* (−1,69%). No entanto, obteve um aumento significativo nos resultados de todas as demais classes. Observe-se que 14 classes de gestos alcançaram mais de 97,00% de acurácia de reconhecimento, sendo que três delas alcançaram a acurácia máxima de 100,00% (*cheduepalle*, *daccordo* e *sonostufo*). Mesmo o pior resultado, o da classe *vatenne*, conseguiu ainda ultrapassar os 92,00%. Na Figura 46 pode ser vista a matriz de confusão com todos os resultados obtidos pelo modelo proposto.

Como apresentado na Seção 3.5, com o modelo *Star* $RGB_{SoftAtt}$, alguns gestos foram confundidos entre si (recorde-se a matriz de confusão da Figura 59). Após analisar os vídeos originais, tal confusão se deu, possivelmente, por eles terem movimentos semelhantes. Eis os principais gestos confundidos: *noncenepiu*, *ok*, *freganiente*, *prendere*, *cosatifarei*, *seipazzo*, *buonissimo*, *vattene*, *vieniqui*, *chevuoi* e *messidaccordo*. Agora, perceba-se que o *BStar* $iRGB_{hand}$ obteve as melhoras mais significativas de acurácia justamente para os referidos gestos. Tais melhoras foram de 6,97%, 8,62%, 6,48%, 3,26%, 2,66%, 5,95%, 3,37%, 3,94%, 3,84%, 4,55% e 2,78%, respectivamente. Esses resultados não foram uma coincidência, na verdade, eles são consequência do uso da imagem das mãos como informação de contexto que complementa a informação de movimento extraída pelo *Star iRGB*.

É importante observar que não foi qualquer informação de contexto que melhorou os resultados, pois o uso da imagem completa em *BStar* $iRGB_{image}$ não alcançou resultados satisfatórios. Nesse caso, foi a informação de contexto (imagem das mãos) escolhida por

Tabela 22 – Comparação dos resultados obtidos pelos modelos *BStar iRGB*$_{hand}$, *Star iRGB*$_{lstm}$ e *Star RGB*$_{SoftAtt}$ para cada classe de gestos do conjunto de dados Montalbano.

Gesto	Modelos (%)			Diferenças (%)	
	BStar iRGB$_{hand}$	*Star iRGB*$_{lstm}$	*Star iRGB*$_{SoftAtt}$	(hand - lstm)	(hand - SoftAtt)
vattene	92,70	91,01	88,76	1,69	3,94
vieniqui	96,15	95,05	92,31	1,10	3,84
perfetto	93,82	93,26	95,51	0,56	-1,69
furbo	97,75	89,33	96,07	8,42	1,68
cheduepalle	100,00	98,84	99,42	1,16	0,58
chevuoi	96,97	93,94	92,42	3,03	4,55
daccordo	100,00	98,77	98,77	1,23	1,23
seipazzo	98,92	96,76	92,97	2,16	5,95
combinato	99,46	98,91	98,91	0,55	0,55
freganiente	98,24	95,29	91,76	2,95	6,48
ok	95,98	91,38	87,36	4,60	8,62
cosatifarei	97,34	94,15	94,68	3,19	2,66
basta	98,90	97,79	97,24	1,11	1,66
prendere	97,28	95,11	94,02	2,17	3,26
noncenepiu	93,60	85,47	86,63	8,13	6,97
fame	98,38	98,38	99,46	0,00	-1,08
tantotempo	97,69	97,11	98,27	0,58	-0,58
buonissimo	97,19	94,38	93,82	2,81	3,37
messidaccordo	98,89	95,56	96,11	3,33	2,78
sonostufo	100,00	97,14	97,14	2,86	2,86

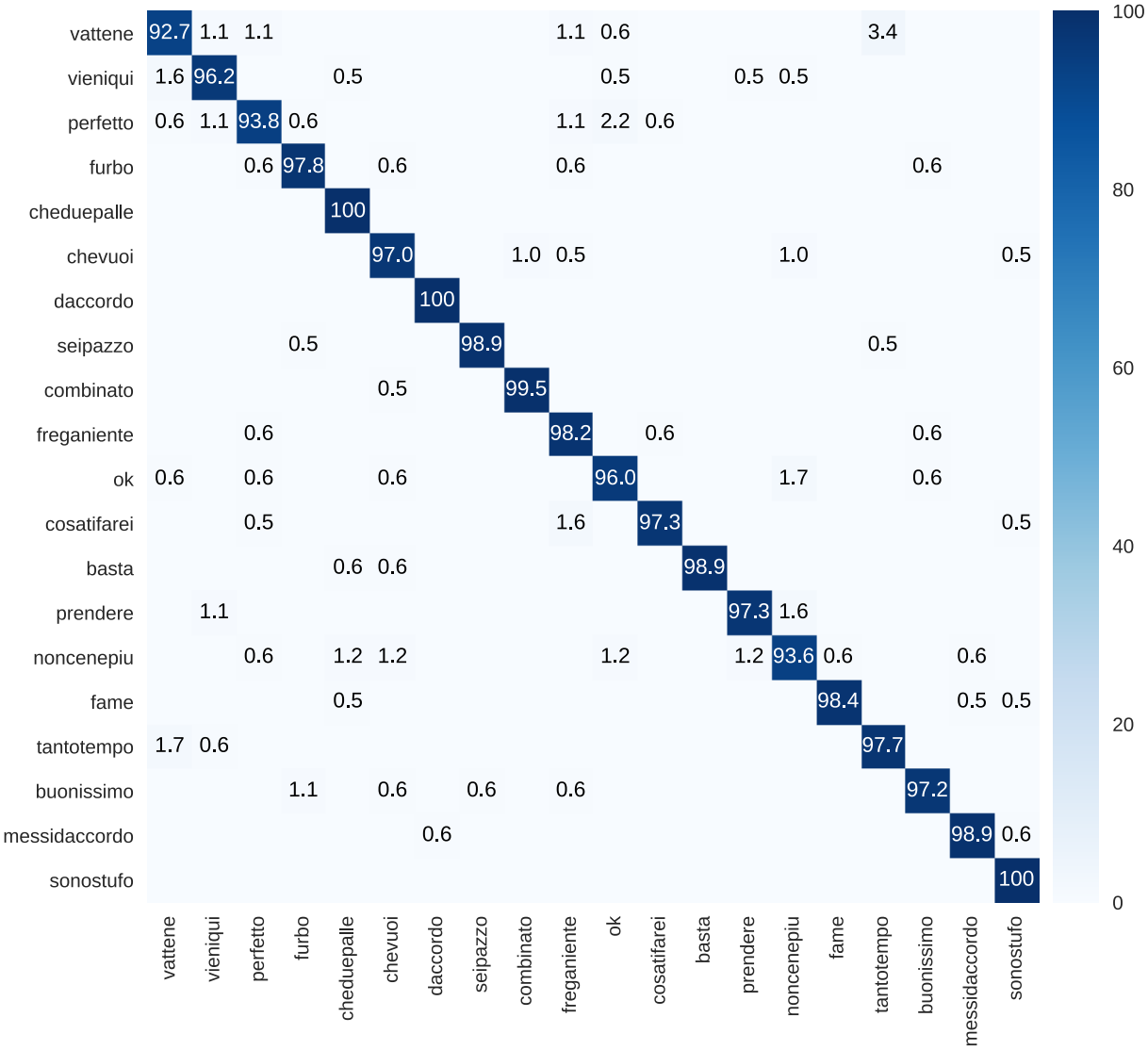
Fonte: Próprio autor.

meio da análise empírica (priori) do problema e do conjunto de dados, assim como foi apresentado em uma das hipóteses levantadas na Seção 1.3.

Mesmo consciente que a informação de contexto trouxe melhoras significativas para os resultados, é importante saber qual a sua contribuição efetiva em relação à de movimento. Nesse sentido, pode-se aproveitar das propriedades do operador de *soft-attention*. Pois ele é o mecanismo responsável por estabelecer tal contribuição, ao determinar os pesos que ponderam e fundem as características extraídas pelas duas CNNs. Assim, para poder realizar tal análise, a Figura 47 apresenta alguns gráficos correspondentes aos pesos atribuídos pelo *soft-attention* à informação de movimento (*Star iRGB*) e à de contexto (mãos). Note-se que para quase a totalidade das amostras de cada um dos gestos apresentados, o peso associado à informação de contexto foi maior que aquele associado à informação de movimento. Perceba-se ainda que, em algumas amostras, a exemplo do gesto 17, o peso atribuído ao movimento foi quase nulo. Isso mostra que a informação de contexto é realmente relevante para o problema, e que, comprovadamente, o mecanismo de *soft-attention* é eficaz na tarefa de fundir características de diferentes fontes que contribuem de maneira distinta para a solução do problema.

Para poder avaliar o tempo de resposta da solução proposta, foi realizado um experimento semelhante ao apresentado na Seção 3.4.3. Aqui, cada nova imagem extraída do conjunto de testes do Montalbano foi dada como entrada para o *Openpose* e para a Equação (3.7). Dessa forma, utilizou-se a informação do esqueleto para recortar as áreas nas imagens contendo as mãos. Em seguida, a representação *Star iRGB*, juntamente com a imagem contendo as mãos, foi enviada para um servidor onde o modelo *BStar iRGB*$_{hand}$,

Figura 46 – Matriz de confusão das predições feitas pelo modelo *BStar* $iRGB_{hand}$ treinado e testado com o conjunto de dados Montalbano.

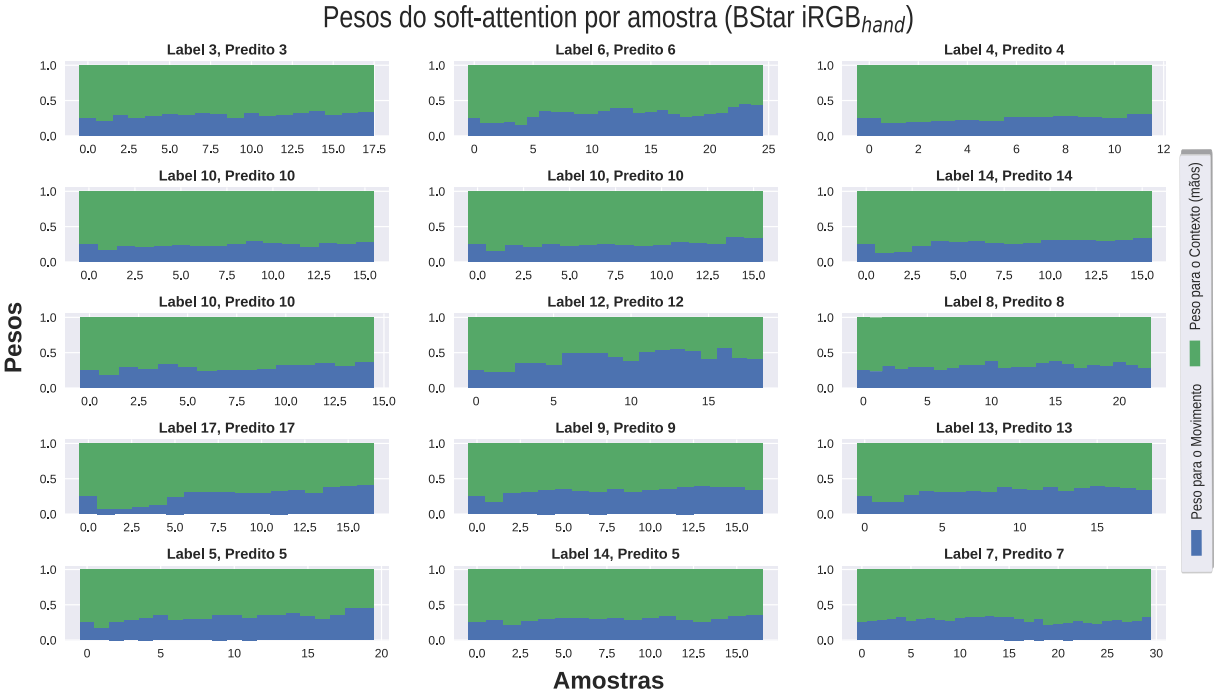


Fonte: Próprio autor.

previamente treinado, realizava a predição por meio de uma simulação de Monte Carlo de 20 amostras. A Tabela 23 traz um resumo dos tempos médios de resposta do sistema.

Como pode ser visto, para cada *frame* do vídeo, o tempo de resposta do *BStar* $iRGB_{hand}$ é em média de 85ms quando o modelo e o *Openpose* são executados em GPU e 1247ms quando todo o processo é executado em CPU. Dessa forma, o modelo proposto pode ser executado em GPU, sem prejudicar a resposta do sistema a cada novo *frame* recebido uma vez que esse tempo médio de resposta é menor que o tempo médio entre dois *frames* consecutivos (100ms). Por outro lado, devido ao uso do *Openpose* para a extração das mãos, o processo, quando executado em CPU, leva mais de 1s para processar cada *frame*, o que inviabiliza a sua utilização em CPU para a predição de gestos em tempo real

Figura 47 – Gráficos com os pesos calculados pelo operador de *soft attention* para cada amostra (observação) de um gesto dado como entrada ao modelo *BStar iRGB*$_{hand}$. Cada amostra representa um *frame* de uma sequência. A predição é feita por meio do argumento com maior probabilidade estimada na última amostra.



Fonte: Próprio autor.

Tabela 23 – Tempos médios de resposta do sistema calculados para a predição dos gestos no conjunto de dados de teste do Montalbano utilizando o *BStar iRGB*$_{hand}$. Observe-se que apenas o modelo e o *Openpose* podem ser executados em CPU ou GPU. O processo de cálculo do *Star iRGB*, da extração das mãoes, os serviços de comunicação sempre são executados na CPU. Todos os valores estão arredondados para o maior inteiro mais próximo.

Dispositivo	Tempo médio (ms)					
	Modelo	*Openpose*	Mãos	*Star iRGB*	Comunicação	Resposta
CPU	137	1097	1	1	11	1247
GPU	17	63	-	-	-	85

Fonte: Próprio autor.

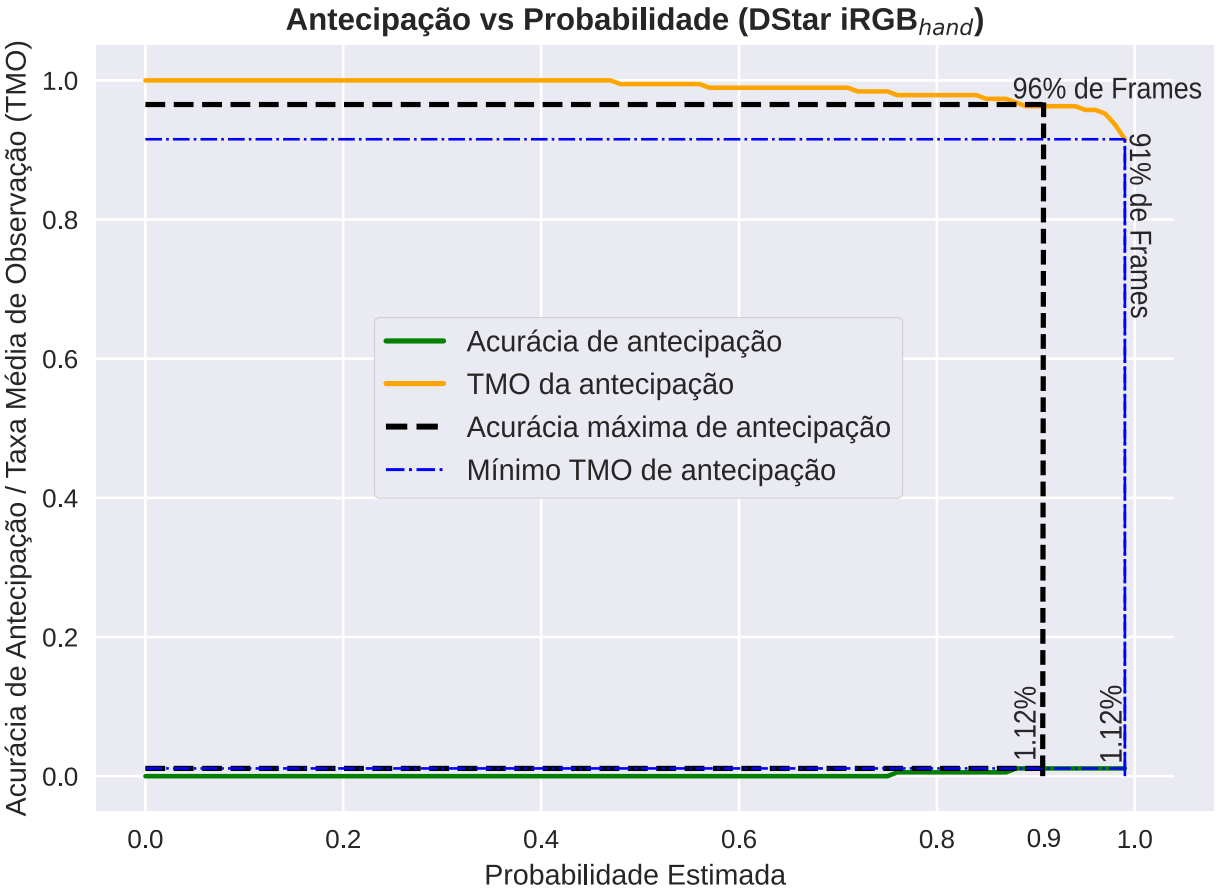
no Montalbano.

4.5.3.2 Antecipação

Assim como na Seção 4.4.5, aqui será avaliada a capacidade do *BStar iRGB*$_{hand}$ de antecipar os gestos do conjunto de testes do Montalbano utilizando a incerteza estimada como limiar de tomada de decisão. Antecipar gestos no Montalbano é um problema

significativamente mais complexo do que antecipar as ações do conjunto de dados Acticipate. Assim, como esperado, utilizando o modelo determinístico *DStar* $iRGB_{hand}$ e um limiar $p = 0{,}90$ sobre a probabilidade por ele estimada, conseguiu-se uma acurácia de antecipação de apenas 1,12%, observando, em média, 96% dos *frames* de cada vídeo. Perceba-se que, mesmo com uma acurácia média de reconhecimento de 96,95 (vide Tabela 21), esse péssimo resultado descarta a possibilidade de utilização de tal modelo para a antecipação dos gestos para os quais ele foi treinado a reconhecer. O valor do limiar p foi obtido por meio do mesmo procedimento utilizado na Seção 4.4.5, onde, para cada limiar entre [0,1], analisou-se o resultado da acurácia média de antecipação alcançada pelo modelo. Na Figura 48, pode ser visto o gráfico com os resultados desse processo.

Figura 48 – Variação da acurácia de antecipação e da taxa média de observação para o limiar que utiliza o valor da probabilidade estimada pelo modelo determinístico, *DStar* $iRGB_{hand}$.

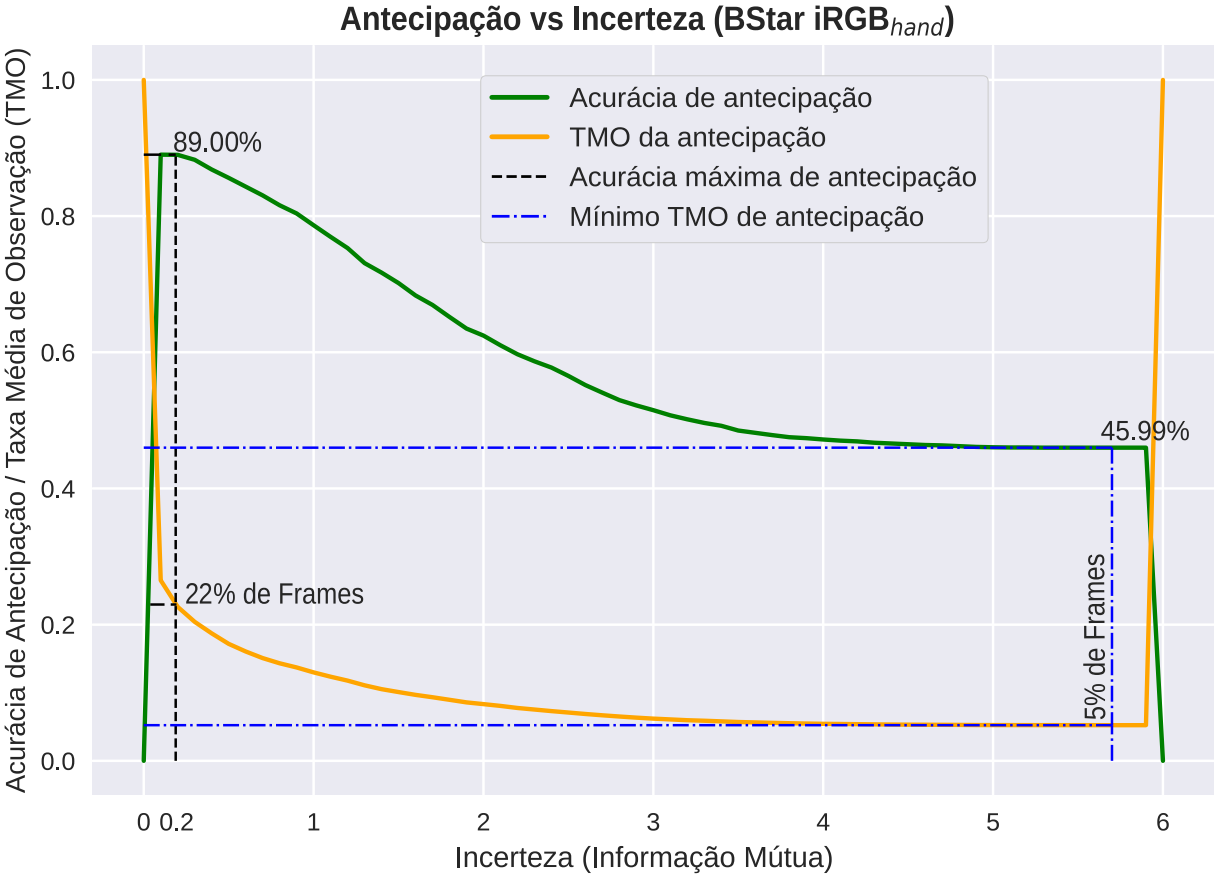


Fonte: Próprio autor.

Por outro lado, utilizando o modelo estocástico *BStar* $iRGB_{hand}$, e um limiar $u = 0{,}2$ sobre a incerteza por ele estimada, obteve-se uma acurácia média de antecipação de 89%, utilizando uma média de apenas 22% dos *frames* de um vídeo contendo um gesto. Mesmo sendo 8,46% menor que a acurácia média de reconhecimento alcançada pelo mesmo modelo, esse valor de acurácia de antecipação pode ser considerado satisfatório, devido

à complexidade do problema e da incapacidade de antecipação demonstrada pela sua versão determinística. Aqui, o valor do limiar u foi determinado da mesma forma como apresentado para o modelo determinístico. Os resultados do processo podem ser vistos no gráfico da Figura 49. A Tabela 24 traz um resumo dos principais resultados obtidos por ambos os modelos para diferentes valores de limiares. Assim, considerando que essa versão reamostrada do Montalbano (10 FPS) possui uma média de 20 *frames* por gesto, esse resultado significa que, em média, é possível antecipar corretamente 89% dos gestos desse conjunto de dados observando menos de 5 *frames*.

Figura 49 – Variação da acurácia da antecipação e da taxa média de observação utilizando um limiar sobre a incerteza estimada pelo modelo estocástico *BStar iRGB*$_{hand}$.



Fonte: Próprio autor.

Mesmo com o modelo estocástico, a utilização da probabilidade média como tomada de decisão não é efetiva para a antecipação. Perceba-se na Figura 50 que ao antecipar o gesto utilizando um limiar $p = 0{,}90$ sobre a probabilidade média estimada, o modelo se comporta com excesso de confiança e antecipa os gestos de maneira equivocada. Pode-se ainda aproveitar os mesmo gráficos para mostrar como o modelo proposto é capaz de acumular o conhecimento necessário para a separação entre as classes. O mesmo inicia com uma alta probabilidade para algumas poucas classes, e com probabilidade próxima de zero para as demais. Esse comportamento já era esperado, pois, como a forma das

Tabela 24 – Resultados obtidos pelo modelo estocástico em comparação aos obtidos por sua versão determinística.

Nome do Modelo	Parâmetro	Acurácia da Antecipação	Taxa Média de Observação
DStar iRGB$_{hand}$	$p = 0{,}99$	01,12%	91,00%
DStar iRGB$_{hand}$	$p = 0{,}90$	01,12%	96,00%
BStar iRGB$_{hand}$	$u = 5{,}70$	45,99%	**05,00**%
BStar iRGB$_{hand}$	$u = 0{,}20$	**89,00**%	22,00%

Fonte: Próprio autor.

mãos é levada em consideração, já nos primeiros *frames*, várias classes podem ser quase que totalmente desconsideradas, uma vez que elas são representadas por formas de mãos diferentes.

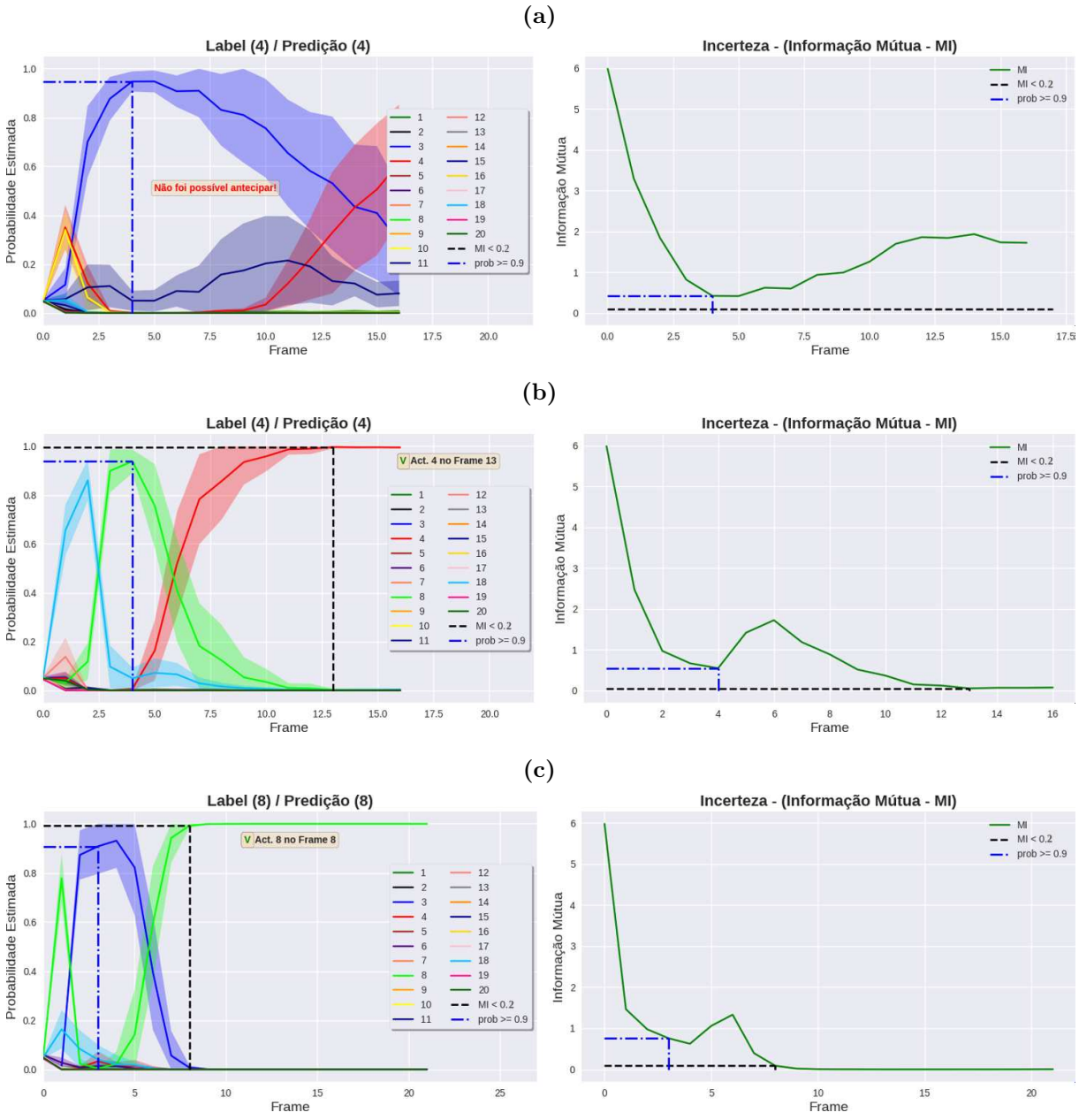
A fim de apresentar em quais classes o modelo cometeu mais erros, a Figura 51 traz a matriz de confusão com os resultados da acurácia de antecipação obtida pelo modelo em cada uma das classes de gestos. Note-se que alguns dos gestos que mais dependem da informação das mãos alcançaram os piores resultados: *perfetto* (71,90%), *ok* (82,80%), *noncenepiu* (83,10%), *freganiente* (84,10%) e *furbo* (88,80%). Após observar as imagens das mãos dadas como entrada ao modelo, percebeu-se que muitas delas possuíam majoritariamente as regiões dos braços, do fundo, ou de outras partes do corpo. Em muitos casos, mesmo quando a mão estava presente, os dedos não apareciam, o que, sem dúvida, prejudica a determinação da sua forma. Essa falta de representabilidade das mãos na informação de entrada para o modelo possivelmente provocou tais erros na antecipação dos gestos.

Para poder comparar com resultados da literatura, a antecipação foi medida em termos de TPR[4] (*True-Positive Rate*) e FPR[5] (*False-Positive Rate*). Dessa forma, o modelo alcançou uma $TPR = 93{,}52\%$ e uma $FPR = 0{,}34\%$ utilizando em média 22% das observações. Por outro lado, o modelo proposto por Gupta et al. (2019) alcançou uma $TPR = 83{,}00\%$ e uma FPR = 5,6% utilizando em média 20% das observações. Já o modelo proposto em Molchanov et al. (2016) obteve uma $TPR = 94{,}80\%$ e uma FPR = 0,8% utilizando em média 41% das observações. Note-se que o modelo proposto obteve resultados bem superiores que o de Gupta et al. (2019) tanto na TPR quanto na FPR com um taxa extra de observação de apenas 2%. Já os resultados em Molchanov et al. (2016), mesmo com uma TPR levemente menor (93,52% *vs* 94,80%), o modelo aqui proposto obteve uma FPR menor (0,3% *vs* 0,8%), e, o que é mais importante, com este modelo a antecipação

[4] A métrica TPR, na tarefa de antecipação, mede qual a taxa de acertos do modelo em relação ao número de gestos antecipados.

[5] A métrica FPR, na tarefa de antecipação, mede a taxa de não antecipação do modelo, ou seja, o número de gestos não antecipados em relação à quantidade total de gestos. O termo *false-positive* advém do fato de que,quando modelo não antecipa o gesto, significa que ele assume que o gesto está fora da sequência, o que resulta em um *falso-positivo*.

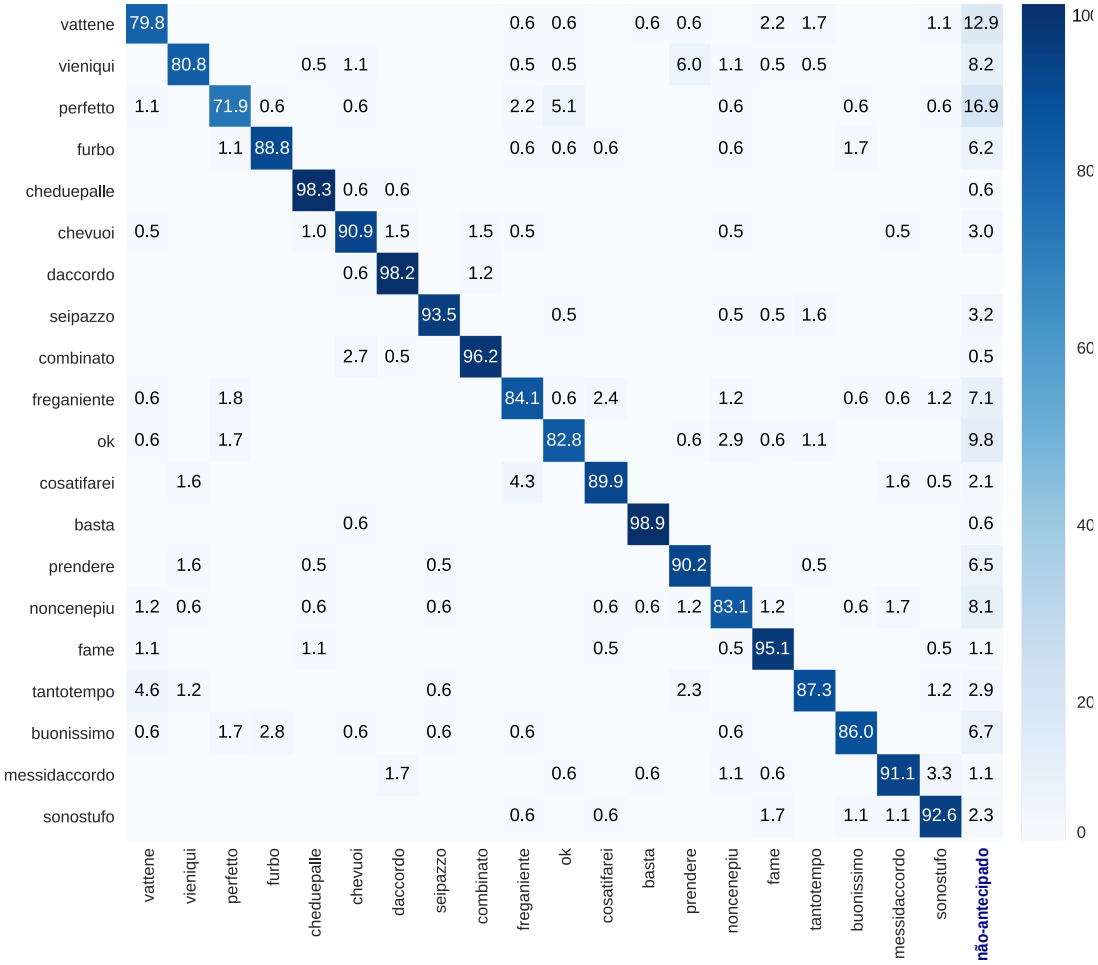
Figura 50 – Exemplo de algumas predições para gestos presentes no conjunto de dados Montalbano. No título de cada gráfico, o termo predição representa a tarefa de reconhecimento (predição realizada no último *frame* da sequência). A tarefa de antecipação é realizada utilizando um limiar $p = 0{,}9$ sobre a probabilidade média estimada com as 20 simulações (*MC dropout*), ou por meio de um limiar $u = 0{,}2$ sobre a incerteza estimada (informação mútua).



Fonte: Próprio autor.

pode ser realizada utilizando quase 50% menos observações (22% *vs* 41%). Importante salientar que ambas as propostas comparadas utilizam dados multimodais. Além disso, suas arquiteturas são baseadas em CNN 3D que extraem características de mais de uma fonte de informação, fundem-nas e passam-nas para uma RNN. Assim, possivelmente, elas demandam uma quantidade de processamento que inviabiliza a sua utilização em uma

Figura 51 – Matriz de confusão com os resultados de antecipação feita pelo modelo *BStar* $iRGB_{hand}$ treinado com o conjunto de dados Montalbano. Note-se que uma nova classe foi inserida (*não-antecipado*) a fim de apresentar os resultados referentes aos gestos não antecipados.



Fonte: Próprio autor.

aplicação real *online*. Ou seja, o modelo de antecipação aqui apresentado mostrou-se ser superior aos das propostas comparadas em vários aspectos.

4.6 Comentários Gerais

Como esperado, os modelos bayesianos aqui propostos (Propostas 1 e 2) forneceram melhores resultados de antecipação que sua respectivas versões determinísticas, com um pequeno custo em observações adicionais. O excesso de confiança na predição dos modelos diminui ao aguardar mais observações. No entanto, como foi possível observar na Seção 4.4.5, para modelos determinísticos, essa melhoria na antecipação é representada por um novo hiperparâmetro a ser escolhido (z) e que não fornece resultados satisfatórios. Por outro lado, usando a incerteza como valor de limiar, não se aumenta o número de

hiperparâmetros a serem escolhidos (apenas troca-se a probabilidade pela incerteza), e o modelo é capaz de obter melhores resultados em termos de acurácia, com um pequeno custo na taxa extra de observação.

Sendo assim, voltando-se à indagação levantada por Jain et al. (2015) que diz: *dado que a probabilidade de uma das classes do modelo tenha ultrapassado o limiar estabelecido, a ação deve ser predita imediatamente ou é necessário esperar por mais observações a fim de aumentar a confiança na predição?* Após as análises dos resultados aqui obtidos pode-se responder que: *nem a primeira nem a segunda opção estão corretas, pois a probabilidade estimada por um modelo determinístico não é uma boa métrica para se estabelecer um limiar de antecipação.* O melhor é utilizar um limiar sobre a incerteza epistêmica estimada pelo modelo para realizar a antecipação da ação correspondente à classe com maior probabilidade média no momento em que esse limiar for ultrapassado. Com relação ao questionamento aqui levantado de: *quando a saída de um modelo é confiável o suficiente para antecipar uma ação?* A resposta é: *quando o modelo for capaz de realmente representar a incerteza epistêmica sobre a predição do modelo e o valor da incerteza ultrapassar um limiar pré-estabelecido.*

Na visão deste trabalho, o *MC Dropout* (GAL; GHAHRAMANI, 2016a; GAL; GHAHRAMANI, 2016b) e o *Variational Dropout* (KINGMA; SALIMANS; WELLING, 2015) foram os melhores modelos implementados. Uma vez que o *dropout* e a reparametrização local podem fornecer uma amostra diferente para cada observação, um lote com S observações corresponde a uma simulação de MC de tamanho S, que ajuda a modelar a inferência de distribuição à posteriori. Além disso, para a predição, é necessário apenas criar um lote de tamanho S, repetindo a mesma observação, o que favorece à predição paralela em GPUs. Por outro lado, como o truque de reparametrização não tira proveito do processamento em lote para realizar a amostragem dos pesos, todas as observações no lote usam o mesmo peso amostrado.

Consequentemente, nos experimentos realizados na Seção 4.4.4, os modelos baseados em *BBB* levaram mais tempo para serem treinados que os baseados em *MC Dropout* e, durante a predição, ele precisa executar o modelo S vezes com a mesma observação, o que não permite paralelizar a predição nas GPUs. No entanto, parece que uma vantagem significativa do *BBB* é a possibilidade de podar o modelo analisando cada parâmetro. Pois, como são distribuições gaussianas, a relação média-variância pode indicar se um parâmetro é necessário ou se ele pode ser descartado durante a etapa de predição (GRAVES, 2011; BLUNDELL et al., 2015).

Como apresentado, o modelo de antecipação de gestos não foi tão efetivo quanto o reconhecedor. Uma explicação para tal resultado é que, uma vez que a informação das mãos é importante para a separação entre as classes, a representação de tal informação foi prejudicada devido ao nível de ruído oferecido pelo *Openpose*. Assim, as juntas dos punhos

e dos cotovelos nem sempre estão bem posicionadas. O outro problema está na maneira como as mãos são extraídas. Apenas recortar uma área de tamanho fixo ao redor das juntas que representam os punhos nem sempre retorna as imagens correspondentes às mãos. A depender da distância das mãos para a câmera, algumas de suas partes podem ser perdidas. Nesse sentido, uma possível solução seria utilizar um modelo de detecção de mãos, ao invés do *Openpose*. Sistemas de detecção de objetos poderiam ser treinados para reconhecer as mãos e seriam bem mais rápidos que o *Openpose*. Nesse sentido, seria possível obter uma representação mais significativa das mãos, ao passo que o tempo de processamento do modelo pode ser consideravelmente diminuído quando comparado ao do *Openpose* e possivelmente forneceriam melhores resultados. Os principais modelos de detecção de objetos disponíveis, como Yolo V4 (BOCHKOVSKIY; WANG; LIAO, 2020), SSD (LIU et al., 2016), RetinaNet (LIN et al., 2017) e Faster R-CNN (REN et al., 2015) podem ser executados em menos de *20ms* em GPU. Isso implica em uma melhora considerável no tempo de processamento em relação ao *Openpose*.

A fim de mostrar a contribuição dada por este trabalho no reconhecimento e antecipação de gestos, a Tabela 25 resume os principais resultados obtidos sobre o conjunto de dados Montalbano. Observe-se que a maioria das propostas apresentadas neste trabalho alcançaram resultados bem competitivos para a tarefa de reconhecimento. Sendo que o *BStar* $iRGB_{hand}$, ao obter uma acurácia média de reconhecimento de 97,46%, é, neste momento, o segundo melhor resultado para reconhecimento de gestos para o conjunto de dados Montalbano, ficando atrás, por menos de 1%, dos resultados apresentados por Molchanov et al. (2016) (98,20%) e Gupta et al. (2019) (97,70%), os quais são propostas multimodais. Por isso, é importante frisar mais uma vez que, aqui, foi usada apenas a informação de cor dos vídeos segmentados, e mesmo assim, esse resultado ultrapassa quase todos os outros, independente do tipo de dados que utilizam, sejam multimodais ou não.

Por fim, mas não menos importante, avaliando os resultados na tarefa de antecipação (TPR), mesmo que em termos quantitativos esta proposta tenha ficado levemente atrás dos resultados apresentados por Molchanov et al. (2016) (também menos de 1%), em termos qualitativos, a proposta aqui apresentada pode ser considerada superior às demais. Além disso, conforme já comentado, como os autores utilizaram dados multimodais, o seu modelo é baseado em arquiteturas de alta complexidade (CNN 3D) e, portanto, demandam mais poder de processamento que a o *BStar* $iRGB_{hand}$. Vendo de um outro ponto de vista, mesmo considerando a acurácia de antecipação, e sabendo que ela é menor que a de reconhecimento, ela consegue ser tão boa quando a acurácia de reconhecimento de muitos métodos multimodais.

É importante ainda lembrar que, apesar dos ótimos resultados obtidos, como discutido no experimento de tempo real apresentado na Seção 3.3.5, para que o reconhecimento ou a antecipação possa ser realizada de modo *online*, onde as imagens são fornecidas

Tabela 25 – Comparando os resultados obtidos pelas propostas apresentadas com os dos principais trabalhos que objetivaram reconhecer os gestos do conjunto de dados Montalbano utilizando os vídeos segmentados. TPR corresponde à métrica *True-Positive Rate* e ACC à acurácia.

Trabalho	Tipo de dados	Resultado	Problema
Cao, Zhang e Lu (2015a)	RGB	60,07%	Reconhecimento
Fernando et al. (2015)	Esqueleto e Áudio	80,29%	Reconhecimento
Wu et al. (2016)	Profundidade	82,62%	Reconhecimento
Escobedo e Camara (2015)	Esqueleto e RGB-D	88,38%	Reconhecimento
Wu e Shao (2014)	RGB-D	90,30%	Reconhecimento
Efthimiou et al. (2016)	Áudio e RGB	93,00%	Reconhecimento
Liu et al. (2020)	Esqueleto	93,80%	Reconhecimento
Pigou et al. (2014)	RGB-D	95,68%	Reconhecimento
Neverova et al. (2016)	Áudio, RGB-D e Esqueleto	96,81%	Reconhecimento
Pigou et al. (2018)	Esqueleto e RGB-D	97,23%	Reconhecimento
Molchanov et al. (2016)	**RGB-D e Fluxo óptico**	**98,20%**	**Reconhecimento**
Gupta et al. (2019)	**RGB-D e Fluxo óptico**	97,70%	Reconhecimento
Gupta et al. (2019)	**RGB-D e Fluxo óptico**	83,00%	Antecipação (TPR)
Molchanov et al. (2016)	**RGB-D e Fluxo óptico**	**94,80%**	**Antecipação (TPR)**
Proposta	Tipo de dados	Resultado	Problema
Star $iRGB_{rnn}$	RGB	91,11%	Reconhecimento
BStar $iRGB_{image}$	RGB	94,18%	Reconhecimento
Star $RGB_{SoftAtt}$	RGB	94,58%	Reconhecimento
Star $iRGB_{lstm}$	RGB	94,89%	Reconhecimento
Star $iRGB_{gru}$	RGB	94,73%	Reconhecimento
DStar $iRGB_{hand}$	RGB	96,95%	Reconhecimento
BStar iRGB$_{hand}$	**RGB**	**97,46%**	**Reconhecimento**
BStar iRGB$_{hand}$	**RGB**	**93,52%**	**Antecipação (TPR)**
BStar iRGB$_{hand}$	**RGB**	**89,00%**	**Antecipação (ACC)**

Fonte: Próprio autor.

em um fluxo contínuo, além dos modelos reconhecedores e antecipadores apresentados, é necessário que os vídeos sejam segmentados, ou que, pelo menos, seja indicado em que instante um gesto ou uma ação se inicia.

5 RECONHECIMENTO E ANTECIPAÇÃO ONLINE DE GESTOS

> To me programming is more than an important practical art. It is also a gigantic undertaking in the foundations of knowledge.
>
> Grace Hopper

O principal objetivo deste capítulo é aplicar os conhecimentos adquiridos nos capítulos anteriores para a resolução de um problema de reconhecimento e antecipação de gestos *online*. Dessa forma, será possível demonstrar que as propostas anteriores são factíveis de serem implementadas em um ambiente interacional real. Nesse sentido, primeiro, será proposto um modelo de reconhecimento e antecipação para um ambiente monocâmera representado pelo conjunto de dados Montalbano. Depois, será proposto um outro modelo que atuará na antecipação e reconhecimento de gestos em uma situação bem mais desafiadora: um espaço inteligente multicâmeras.

Um espaço inteligente pode ser entendido como um ambiente capaz de tomar decisões por meio de dados coletados de vários sensores e dispositivos interconectados (LEE; ANDO; HASHIMOTO, 1999). Em um espaço inteligente, são diversos os possíveis agentes coletores de dados, por exemplo, robôs, computadores, sensores de presença, câmeras, dentre outros. Alguns desses agentes, como é o caso dos robôs, também são atuadores e, por isso, recebem instruções do espaço sobre as tarefas a serem executadas. Dentro do ambiente monitorado, um usuário pode interagir diretamente com os dispositivos, bem como com o próprio ambiente. Dessa forma, como as decisões são tomadas pelo espaço, ele deve ser capaz de prover a infraestrutura necessária para que a interação ocorra efetivamente, independente de para quem ela foi dirigida. Dessa maneira, para facilitar a utilização das aplicações em diferentes locais, um ambiente interacional deve satisfazer dois requisitos principais: *(i)* utilizar uma interface de interação intuitiva e *(ii)* utilizar sensores comuns aos mais diversos ambientes. Foi nesse sentido que este trabalho escolheu os gestos dinâmicos como interface de interação e as câmeras RGB como sensores de captura de dados.

Embora os capítulos anteriores tenham apresentado as dificuldades no reconhecimento dos gestos dinâmicos, esse problema é ainda maior dentro do escopo dos espaços inteligentes multicâmeras. Mesmo que as várias câmeras possam ajudar na diminuição de características oclusas, elas aumentam significativamente a necessidade de processamento do modelo e ainda geram um problema referente à decisão de que câmera deve ser utilizada

no reconhecimento de gestos. Além disso, as imagens são capturadas de diversos ângulos distintos, o que pode dificultar muito o aprendizado do modelo. Sendo que, em uma situação de reconhecimento *online*, esses problemas são ainda mais presentes, uma vez que não se tem a informação do tamanho de cada sequência que representa um gesto. Ademais, um mesmo gesto pode ser realizado com mais ou menos imagens a depender do usuário. Ou seja, por um lado, mais câmeras aumenta a área monitorada pelo espaço ao mesmo tempo que reduz os problemas com oclusão, por outro lado, quanto mais câmeras, mais complexo pode se tornar o processo de reconhecimento.

Considerando-se o uso do *Star iRGB* em um ambiente multicâmeras, tem-se logo um problema: ele forneceria uma representação de movimento para cada câmera, e, caso fossem seguidas as mesmas linhas de raciocínio da propostas anteriores, utilizaria-se de uma CNN para cada câmera presente no espaço. Com isso, a necessidade de processamento aumentaria proporcionalmente ao número de câmeras, o que pode dificultar o uso desse modelo em espaços com muitas câmeras. Dessa forma, objetivando o uso de ambientes multicâmeras para a interação natural, Queiroz et al. (2018) propôs um método que usa as imagens de N câmeras de um espaço inteligente calibrado para reconstruir o esqueleto tridimensional (3D) de cada indivíduo presente no mesmo. Com essa solução, é possível utilizar o potencial dos ambientes com múltiplas câmeras para o reconhecimento (antecipação) de gestos utilizando o movimento fornecido pela evolução do esqueleto tridimensional do usuário no tempo. Note-se que o esqueleto, mesmo sendo bidimensional, mostrou-se eficaz na representação de movimento na proposta de antecipação de ações apresentada na Seção 4.4. Sendo assim, a utilização do esqueleto 3D como fonte de informação de movimento não fugiria ao escopo deste trabalho.

Mesmo com a possibilidade de utilização de esqueletos 3D, alguns trabalhos utilizam modelos baseados em sequências fixas de imagens (ESCOBEDO-CARDENAS; CAMARA-CHAVEZ, 2015; LIU et al., 2019), o que acaba gerando uma dependência de como os gestos devem ser executados, e, por conseguinte, limita a naturalidade da sua execução. Um problema similar ocorre com as abordagens baseadas em regras (SANTOS; NETO; SALEME, 2015), que apresentam uma forte dependência da percepção do projetista. Uma ligeira mudança na realização de um gesto poderia prejudicar o seu reconhecimento. Logo, uma abordagem baseada em um modelo de aprendizado automático, orientado à tarefa de classificação de gestos e que não limitasse o tamanho da sequência de entrada, provavelmente superaria as restrições dos modelos de sequências fixas e os baseados em regras.

Dessa maneira, neste capítulo, propõe-se um reconhecedor de gestos *online* que possa ser utilizado em um espaço interacional multicâmeras. No entanto, para garantir a coerência com os capítulos anteriores e poder fechar o escopo das soluções anteriormente propostas, primeiro, será proposto um modelo que possa ser utilizado no reconhecimento e

antecipação de gestos *online* com o conjunto de dados Montalbano (capturado com uma única câmera). Sendo assim, a proposta consiste em modificar apenas a última camada do modelo *BStar* $iRGB_{hand}$ (anteriormente treinado para reconhecimento e antecipação de gestos na Seção 3.4), acrescentando um novo ramo para que o mesmo possa classificar uma sequência de imagens como sendo *gesto* ou *não-gesto*. Isto possibilitará a segmentação temporal das sequências de imagens que representarão apenas um gesto; as quais poderão alimentar o o ramo original do *BStar* $iRGB_{hand}$, que foi treinado conforme a proposta da Seção 4.5.

Os resultados obtidos nessa primeira proposta poderão servir de parâmetro para se medir as dificuldades encontradas na segunda proposta, que é destinada ao reconhecimento e à antecipação de gestos *online* em um ambiente real multicâmeras. Nesse sentido, utilizar-se-á o método apresentado em Queiroz et al. (2018) para extrair o esqueleto 3D de um usuário presente na cena juntamente com a informação das suas mãos. Assim, as duas informações alimentarão um modelo que também segmentará cada gesto presente nas sequências de imagens e o reconhecerá (antecipará) como pertencente a uma das classes de gestos possível.

Em resumo, as principais contribuições deste capítulo são:

- **Ambiente monocâmera:**
  - um modelo capaz de segmentar temporalmente um fluxo de imagens como sendo *gesto* ou *não-gesto*;
  - um modelo capaz de reconhecer e antecipar gestos de forma *online*.
- **Ambiente multicâmeras:**
  - um modelo capaz de segmentar temporalmente uma sequência contínua de esqueletos 3D como sendo *gestos* ou *não-gestos*;
  - um modelo que permite classificar uma sequência de movimentos de esqueletos 3D como pertencentes a uma das classes de gestos possível; e
  - um sistema que une os modelos de segmentação e classificação, de maneira a possibilitar o reconhecimento e antecipação *online* de gestos em um ambiente interacional real.

5.1 Trabalhos relacionados

Dentre os trabalhos que abordam o problema de reconhecimento de gestos dinâmicos, a maioria deles foca no reconhecimento *offline*, ou seja, reconhecer um gesto presente em uma sequência de observações previamente segmentadas. Em Cao, Zhang e Lu (2015b), os

autores utilizaram um HMM a fim de reconhecer gestos em uma sequência de esqueletos 3D. Já Wang e Wang (2017b) dividiram cada esqueleto 3D de uma sequência em cinco partes principais: pernas, braços e tronco. Em seguida, a sequência correspondente a cada uma dessas partes era fornecida a uma rede neural recorrente do tipo LSTM. As saídas das cinco LSTMs eram então concatenadas e dadas como entrada para uma outra LSTM que classificava a sequência de esqueletos como pertencente a uma das classes de gestos. O problema da abordagem é que a sequência deve conter apenas o gesto, ou seja, deve ter sido segmentada previamente.

Chai et al. (2016) propuseram uma abordagem capaz de reconhecer gestos dinâmicos de modo *online*. Primeiro, realizava-se uma segmentação temporal no vídeo contendo os gestos. Para isso, foram extraídas características das mãos das pessoas por meio de uma Fast-RCNN (GIRSHICK, 2015), rede neural profunda utilizada no reconhecimento de objetos, e do descritor HOG (*Histogram of Oriented Gradients*). Essas características alimentavam uma LSTM responsável por classificar cada *frame* de entrada como sendo *gesto* ou *não-gesto*. Dessa maneira, as sequências marcadas como sendo gesto alimentam um classificador baseado também numa LSTM. Nessa abordagem, os autores utilizaram tanto imagens RGB quanto imagens de profundidade.

Em Neverova et al. (2015) foi apresentada uma abordagem similar a de Chai et al. (2016), no entanto, eles utilizavam todas as informações disponíveis no conjunto de dados Montalbano v2 (imagens RGB e de profundidade, esqueleto 3D e silhueta), e complementavam com a informação de áudio disponível no Montalbano v1. Os autores extraiam características das configurações das juntas de um usuário (velocidades, ângulos e acelerações) e as forneciam para um segmentador temporal baseado numa MLP com apenas uma camada escondida. Similarmente ao trabalho anterior, as sequências segmentadas eram então passadas para uma arquitetura que utilizava, além do esqueleto, imagens RGB, de profundidade e áudio para classificar o gesto presente na sequência. Apesar das duas abordagens oferecerem soluções para o reconhecimento de gestos *online*, as mesmas utilizam dados multimodais e necessitam de grandes conjuntos de dados para serem treinadas. Sendo assim, caso fossem utilizadas em um ambiente multicâmeras, a complexidade de seus modelos cresceria consideravelmente, uma vez que, mesmo sendo multimodais, as informações utilizadas são adquiridas de um único sensor, mais especificamente, um *Kinect* 360.

Em Molchanov et al. (2016) é proposto um modelo que utiliza uma CNN 3D para extrair características de uma sequência de oito imagens. Os mapas de características da CNN passam por um operador de *pooling* temporal e são dados como entrada para uma RNN do tipo GRU que classifica a última imagem da sequência como pertencente a uma das possíveis classes de gestos. Como função de custo os autores utilizaram a chamada CTC (*Connectionist Temporal Classification*) (GRAVES et al., 2006), que tem a tarefa de

encontrar a sequência de rótulos mais prováveis a partir de uma sequência de predições. Utilizando um limiar sobre a probabilidade de cada predição, eles forneceram o primeiro resultado para a antecipação de gestos no Montalbano. Além disso, alcançaram o estado da arte nas métricas de acurácia de reconhecimento dos gestos segmentados (98,2%) e no índice de Jaccard da segmentação e classificação (0,98). Apesar dos resultados, a proposta utiliza três CNN 3D distintas para extrair características de cada uma das três fontes de dados, RGB, profundidade e fluxo óptico. Dessa forma, além de ser uma abordagem para dados multimodais, possivelmente não será factível de utilização em uma aplicação real que exija do sistema um baixo tempo de resposta. Uma abordagem similar foi apresentada em Gupta et al. (2019). Nela, os autores também utilizaram três CNNs 3D como extratores de características para as informações de cor, profundidade e fluxo óptico; um HMM para a segmentação das sequências e uma LSTM como o emissor de probabilidades do HMM. Os autores também apresentaram resultados para a antecipação de gestos no Montalbano, mas não conseguiram ultrapassar nehum dos resultados apresentados por Molchanov et al. (2016). Essa proposta sofre dos mesmos problemas da anterior.

Considerando-se agora os ambientes interacionais, são escassos os trabalhos que abordam o reconhecimento *online* de gestos para esse tipo de ambiente. Em Santos et al. (2017), os autores modelaram um comportamento interacional capaz de interagir com um robô móvel por meio de gestos dinâmicos. Apesar dos gestos serem reconhecidos em tempo real, o reconhecimento era baseado em um conjunto de regras pré-definidas que se baseavam na localização de determinadas juntas do usuário. Esse tipo de reconhecedor é altamente propício a falsos-negativos quando da existência de pequenas variações na maneira como os gestos são executados por diferentes usuários. Assim, gestos que possuem uma pequena variação em sua execução não são detectados nem reconhecidos. Além disso, depende exclusivamente da experiência do(a) projetista, pois não é comumente aprendido por meio da análise automática de dados. Dessa forma, reconhecedores baseados em regras limitam a sua utilização a ambientes interacionais específicos e cuidadosamente projetados. Outras abordagens semelhantes também podem ser vistas em Zhao, Pan e Hu (2013), Santos, Neto e Saleme (2015) e Saleme, Celestrini e Santos (2017).

Mesmo com os problemas apresentados nos trabalhos anteriores, vê-se em alguns deles a importância do uso de esqueletos para o reconhecimento de gestos, ao mesmo tempo que o uso de sensores de profundidade pode limitar a escalabilidade da solução. Sendo assim, visando a interação em espaços inteligentes multicâmeras, Queiroz et al. (2018) propuseram um método capaz de reconstruir tridimensionalmente as juntas dos esqueletos detectados em imagens capturadas por um sistema multicâmeras calibrado. O método proposto pelos autores pode ser resumido em cinco principais etapas: primeiro, utiliza-se o *Openpose* para detectar as juntas dos usuários presentes em cada imagem; segundo, os esqueletos formados pelas juntas recebem um identificador único e são então associados às respectivas câmeras nas quais foram identificados; em seguida, utiliza-se de geometria

epipolar para encontrar a correspondência entre os esqueletos de câmeras diferentes; dessa forma, por meio de uma busca em grafo, agrupam-se as correspondências, a fim de evitar redundâncias; e, finalmente, a reconstrução tridimensional das juntas é feita utilizando suas coordenadas e os parâmetros de calibração das câmeras. Mais detalhes sobre essa proposta podem ser vistos tanto em Queiroz et al. (2018) quanto em Queiroz (2019).

Após essa análise, percebe-se que são poucos os trabalhos que oferecem soluções para o reconhecimento de gestos dinâmicos em ambientes interacionais, e os que assim o fazem, possuem um escopo limitado, o que impossibilita a sua utilização em ambientes mais diversificados. Uma outra percepção é que diferentes trabalhos utilizam o esqueleto como fonte de informação para o reconhecimento de gestos, enquanto que aqueles que pretendem reconhecer gestos em tempo real não são adequados para ambientes multicâmeras. Além, disso, não foram encontrados trabalhos que objetivassem a antecipação de gestos nesse tipo de ambiente.

Dessa maneira, neste capítulo, será proposto um sistema de reconhecimento de gestos *online* que possa ser utilizado em um espaço interacional multicâmeras. Para isso, primeiro, uma bordagem semelhante à de Neverova et al. (2015) e Chai et al. (2016) será implementada utilizando, no entanto, uma versão modificada do *BStar* $iRGB_{hand}$, e será aplicada sobre o conjunto de dados Montalbano. Em seguida, uma abordagem similar será utilizada para o ambiente multicâmeras, porém, utilizando apenas os esqueletos 3D fornecidos pela implementação da proposta apresentada em Queiroz et al. (2018) ao invés do *Star iRGB*. Ambas as proposta serão capazes não só de reconhecer os gestos, mas também de antecipá-los.

5.2 Ambiente monocâmera: Conjunto de dados Montalbano

Como comentado anteriormente, os modelos dessa proposta estarão baseados no *BStar* $iRGB_{hand}$. Assim, as informações de movimento e de contexto serão extraídas da mesma maneira como apresentadas na Seção 4.5.

5.2.1 Proposta

5.2.1.1 Segmentação de movimento

Em uma aplicação *online*, as imagens são fornecidas ao modelo uma a uma, seguindo a ordem de captura. Assim, para que o reconhecedor possa classificá-las como pertencente a uma das classes de gestos, é necessário utilizar-se de algum mecanismo de segmentação temporal que determine quando um possível gesto inicia e termina. O termo em inglês para modelos que desempenham esse tipo de tarefa é *spotting*, como descrito em Chai et al.

(2016). Nesse sentido, propõe-se adaptar o modelo *BStar iRGB*$_{hand}$ para que ele realize tal tarefa. Para isso, será adicionada a ele uma nova LSTM bayesiana de camada única também com 1024 neurônios; porém, seguida de um classificador *softmax* com apenas 2 saídas, as quais representarão as classes *gesto* (1) e *não-gesto* (0). Dessa forma, optou-se por fazer uso da transferência de aprendizado para que a parte da extração de características do *BStar iRGB*$_{hand}$ não necessitasse ser retreinada. Assim, como o segmentador, provavelmente, levará em consideração mais a informação de movimento do que a de contexto, também será treinado um mecanismo de *soft-attention* exclusivo para ele.

5.2.1.2 Classificador

O classificador deverá receber uma sequência de imagens e classificá-la como pertencente a uma das possíveis classes de gestos. Dessa forma, a proposta é utilizar de maneira integral o modelo *BStar iRGB*$_{hand}$ já treinado nos experimentos da Seção 4.5. Com isso, o modelo de segmentação e o de classificação compartilharão o mesmo extrator de características. O objetivo do compartilhamento de tais estruturas é reduzir tanto o tempo de treinamento do modelo segmentador, quanto diminuir o tempo de processamento quando os dois modelos estiverem trabalhando em conjunto na fase de predição. Para fins comparativos, essa proposta será chamada de *BStar iRGB*$_{online}$, e pode ser visualizada por meio da ilustração da Figura 52.

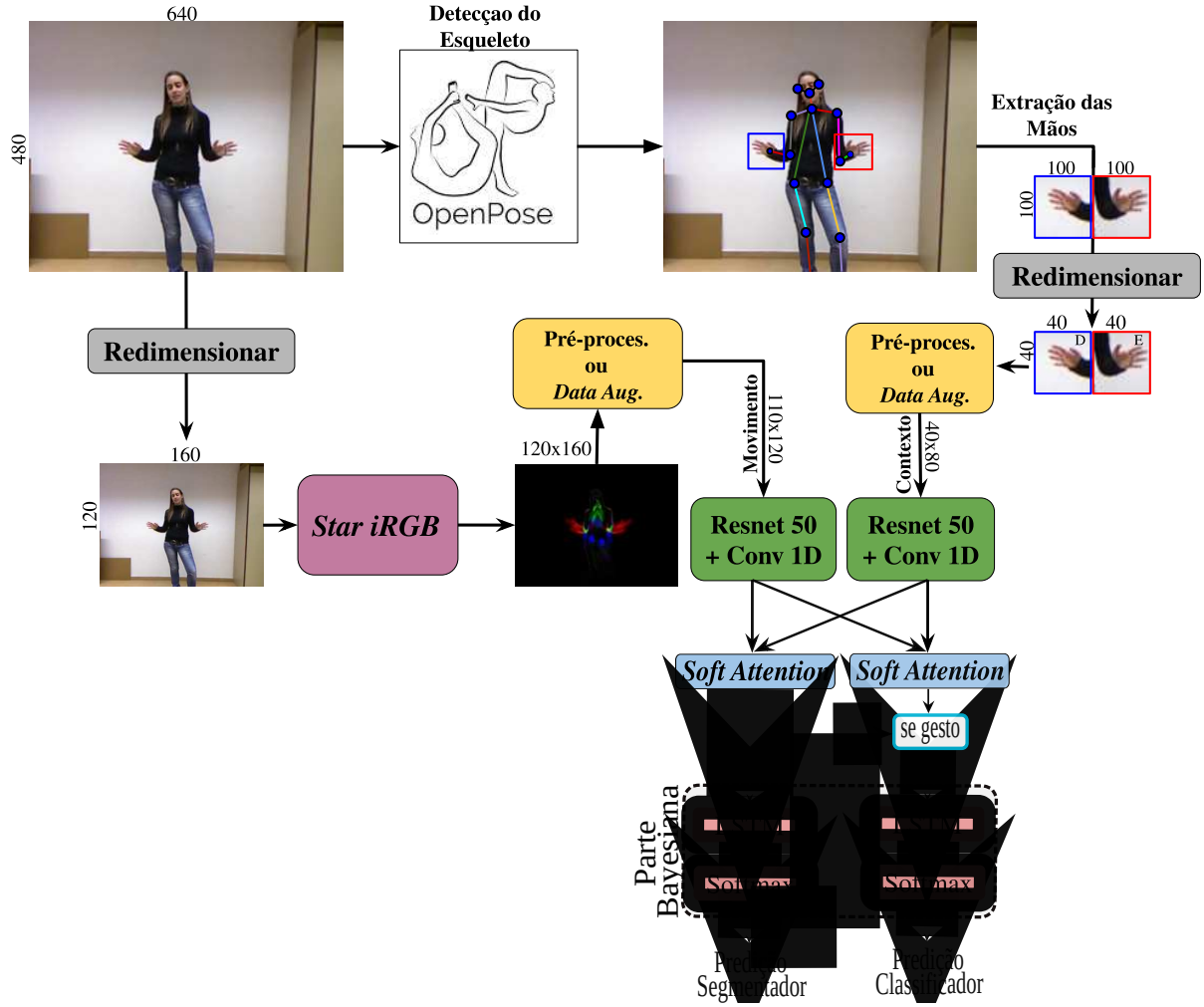
Note-se que o modelo classificador só será acionado caso a entrada seja classificada pelo segmentador como sendo um gesto. Em adição, como ambos os modelos compartilham a parte de extração de características, quando o classificador for acionado, ele reutilizará as características já extraídas para a utilização no segmentador. Assim, em relação ao *BStar iRGB*$_{hand}$, apenas uma nova camada LSTM será acrescentada a essa proposta: a do segmentador. Fazendo com que, supostamente, os tempos de respostas do *BStar iRGB*$_{online}$ e do *BStar iRGB*$_{hand}$ sejam praticamente os mesmos.

Saliente-se que, como tanto o segmentador quanto o classificador são estocásticos, pode-se realizar a antecipação dos gestos utilizando a incerteza como limiar de tomada de decisão.

5.2.2 Experimentos

Para treinar o segmentador, o conjunto de dados Montalbano foi utilizado completamente em sua configuração original (apenas com a reamostragem para 10 FPS, como feito para o *Star iRGB*$_{lstm}$ e o *Star iRGB*$_{hand}$). Assim, foram utilizados não apenas os *frames* contendo os gestos, mas também aqueles que não os contêm (*frames* de *não-gesto*). Dessa maneira, como o treinamento exige que as sequências sejam de mesmo tamanho, cada vídeo

Figura 52 – Ilustração do modelo *BStar iRGB$_{online}$* proposto para a segmentação temporal do movimento e para a classificação dos gestos.



Fonte: Próprio autor.

contendo uma gravação de um indivíduo foi separado em sequências aleatórias com 48 *frames*. Sendo que os *frames* das sequências que pertencem a um dos 20 gestos receberam o rótulo 1 (*gesto*) e os demais, o rótulo 0 (*não-gesto*). Esse processo foi feito tanto para o conjunto de treino, quanto para o de validação. Já para a o conjunto de teste, apenas a rotulação foi modificada. Dessa forma, a predição sobre o conjunto de teste foi realizado utilizando os vídeos completos (sem reamostragem), os quais continham, em média, 1400 *frames* a uma taxa de exibição de 20 FPS. O objetivo de não realizar a reamostragem no conjunto de teste é para garantir uma comparação justa com os resultados da literatura que objetivam o reconhecimento *online*.

Assim como nos capítulos anteriores, o valor da sequência de treinamento (48 *frames*), bem como os outros hiperparâmetros utilizados no treinamento do modelo que obteve melhor resultado, foi determinado por meio de uma Otimização Bayesiana. Um resumo de tais hiperparâmetros podem ser vistos na Tabela 26. Como apenas a LSTM e

o operador de *soft-attention* do modelo de *spotting* foram treinados, muitos dos hiperparâmetros apresentados são os mesmo do *BStar iRGB*$_{hand}$. Para esse treinamento, foram utilizadas as mesmas infraestruturas de *hardware* e *software* utilizadas nos experimentos da Seção 4.5.2.

Tabela 26 – Lista dos hiperparâmetros utilizados para o treinamento do *BStar iRGB*$_{online}$.

Hiperparâmetros	Valor
Geração do *Star iRGB*	
Termo de amortização (α)	0,6
Tamanho da janela (N)	5
Treinamento	
Tamanho do *batch*	32
Truncamento da sequência	48
Tamanho da sequência	24
Quantidade máxima de épocas	50
Taxa de aprendizado (TA)	1e-3
Decaimento da TA (por época)	1%
Taxa de decaimento dos pesos	1e-5
Truncamento do gradiente (Norma)	5,0
Dropout	0,30
Otimizador	Adam

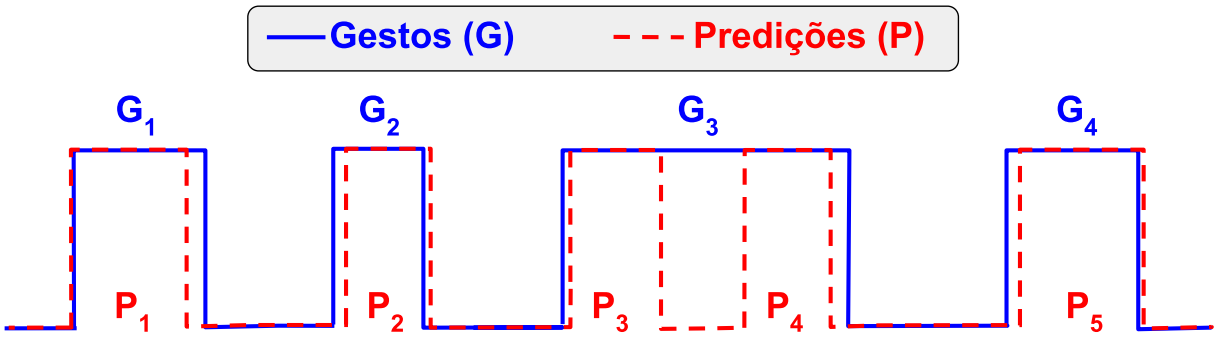
Fonte: Próprio autor.

5.2.2.1 Predição

Para a predição, utilizaram-se as sequências completas do conjunto de teste, possuindo *gestos* e *não-gestos*. Desse modo, para o problema de reconhecimento, sempre que o modelo de *spotting* marcava uma observação como sendo gesto, as próximas observações eram armazenadas até que a primeira predição de um *não-gesto* ocorresse. Nesse momento, a sequência armazenada era passada para o modelo de classificação que determinava a qual das 20 classes de gestos a sequência pertencia. Já para antecipação, o modelo iniciava a predição quando o modelo de *spotting* predizia um *frame* como sendo *gesto* e só parava quando o limiar de incerteza $u = 0{,}2$ era ultrapassado, ou quando um *frame* era classificado como *não-gesto*. Caso o limiar não fosse ultrapassado até o final da sequência, significava que a antecipação não pôde ser realizada.

É importante salientar que o modelo de *spotting* pode gerar vários ruídos na predição (variação entre *gestos* e *não-gestos*), o que poderia prejudicar a segmentação da sequência. Neste sentido, aplicou-se sobre a sua predição um filtro de média móvel com janela de tamanho 3 e sobreposição 2 sobre o resultado da predição. Assim, o problema pôde ser atenuado, uma vez que, com isso, o *spotting* muda a sua predição de *gesto* para *não-gesto* quando a média muda de 1 para 0, ou de *não-gesto* para *gesto* quando a média muda de 0 para 1.

Figura 53 – Exemplo do problema de avaliação. A sequência possui quatro gestos, mas foram detectados cinco.



Fonte: Próprio autor.

5.2.2.2 Método de avaliação

Os dois modelos serão avaliados conjunta e individualmente. Para o modelo segmentador, será utilizada a acurácia média da classificação de cada observação. Já para o modelo classificador, será considerada a acurácia média da classificação de cada sequência contendo um gesto a qual foi determinada pelo rótulo fornecido. Para o modelo completo, deve-se considerar que o segmentador pode detectar mais gestos que os existentes, ou simplesmente não detectar nenhum, o que acarretaria em problemas no uso da acurácia como métrica de avaliação.

A Figura 53 ilustra um exemplo onde a acurácia de classificação é falha. Nela, o segmentador dividiu o gesto G_3 em duas predições (P_3 e P_4). Dessa forma, a sequência possui quatro gestos e o *spotting* identificou cinco. No caso de o classificador errar um dos outros gestos (G_1, G_2 ou G_4) e acertar os dois gestos preditos em G_3, a acurácia da classificação seria 100%, mesmo que um dos gestos da sequência não tenha sido classificado corretamente. Algo ainda pior ocorreria caso o classificador acertasse todas as predições (P_1, P_2, P_3, P_4 e P_5), o que daria uma acurácia de 125%, situação inadmissível, uma vez que a acurácia máxima que um modelo de classificação pode alcançar é de 100%.

Portanto, para avaliar o sistema completo, será utilizado o Índice de *Jaccard* (JI) (Equação (5.1)) que mede a sobreposição entre duas sequências, uma desejada (G) e outra predita (P), mediante o cálculo da interseção sobre a união entre as duas. Essa métrica alcança o valor máximo de 1,0 quando todos os *frames* classificados como sendo um gesto estão corretos. Ou seja, quando a sequência da predição é igual à sequência do rótulo. Dessa forma, caso o modelo de *spotting* detecte um gesto onde não tem, ou caso uma observação pertencente a uma das classes de gestos for classificada erroneamente, o resultado do JI será penalizado.

$$J(G,P) = \frac{|G \cap P|}{|G \cup P|} \tag{5.1}$$

Observe-se que o JI não considera os acertos do segmentador para a classe *não-gesto*. Assim, será também medida a acurácia de todas as predições considerando todas as classes do modelo completo: as classes de gestos juntamente com a classe *não-gestos*. No caso do Montalbano serão 21 classes ao todo (20 gestos e 1 *não-gesto*).

5.2.3 Resultados e discussões

5.2.4 Reconhecimento

Após o treinamento, o modelo segmentador alcançou uma acurácia de reconhecimento de 93,12%. Considerando-se o modelo completo (segmentador e classificador) como tendo 21 classes, o *BStar* $iRGB_{online}$ atingiu uma acurácia de 88,57%. Agora, levando-se em consideração apenas a classificação realizada no último *frame* da sequência, o modelo alcançou uma acurácia média de reconhecimento de 89,44%. Como se sabe, o modelo de classificação alcançou 97,46% de acurácia média de reconhecimento sobre os gestos segmentados (vide Tabela 21). Sendo assim, o valor obtido pelo modelo completo não foi tão bom quanto ele. Isso se deve ao fato de que o modelo de *spotting* não conseguiu segmentar todas as sequências com 100% de acertos na predição de cada *frame*.

Mesmo assim, tendo-se em conta que foi usada apenas de informação de cor, esse é um resultado significativo. Ao analisar as predições realizadas para cada um dos vídeos, percebeu-se que os piores resultados foram alcançados por gestos executados continuamente, ou seja, aqueles cujos braços não retornavam à posição de descanso antes de iniciar um novo gesto. Dessa maneira, o segmentador juntava os dois gestos, dificultando assim a tarefa do classificador. Com isso, percebe-se que o modelo de segmentação é de extrema importância para uma melhora nos resultados. Por outro lado, em muitos dos vídeos, é notória a efetividade do modelo, uma vez que quase todos os gestos são reconhecidos corretamente. A execução do modelo de reconhecimento sobre alguns vídeos do conjunto de testes do Montalbano pode ser visualizada no seguinte *link*: <https://youtu.be/Qc-I73324vQ>.

Assim como mostrado no capítulo anterior, aqui, a informação das mãos também foi de grande importância para o reconhecimento dos gestos. Isso pode ser visto nos gráficos da Figura 54. Note-se como o peso dado pelos operadores de *soft-attention* de ambos os modelos para a informação das mãos manteve uma superioridade em relação ao peso dado para a informação de movimento. Note-se ainda como o operador de *soft-attention* do segmentador utilizou a informação de movimento de maneira mais uniforme que o classificador. Isso se deve ao fato de que o movimento é uma informação importante para a determinação de um *gesto* ou um *não-gesto* em todas as amostas da sequência. Para

o classificador, a depender classe mais provável, a informação de contexto tem muito mais importância que a de movimento, o que causa uma variação no grau de importância atribuído a cada uma delas.

Figura 54 – Pesos calculados pelos operadores de *soft-attention* do modelo de *Spotting* e do modelo de classificação do *BStar iRGB*$_{online}$. No gráfico do classificador, os pesos mostrados correspondem apenas às amostras de contém gestos.

Fonte: Próprio autor.

A fim de comparar com os resultados da literatura, todas as predições dentro de um intervalo determinado pelo segmentador receberam a classe da última predição. Dessa forma, o índice de *Jaccard* obtido foi de 0,807. Caso a competição, para a qual o Montalbano foi lançado, ocorresse hoje, esse valor estaria entre os quatro melhores, sendo o melhor dentre todos os que utilizaram apenas informação de cor.

A Tabela 27 traz um resumo dos principais resultados para a tarefa de localização e reconhecimento de gestos no Montalbano medida por meio do Índice de Jaccard; tanto os apresentados durante a competição quanto aqueles apresentados após a sua realização. Veja-se que algumas propostas alcançaram resultados acima dos 0,9, sendo que uma delas alcançou quase o valor máximo de 1,0. Na interpretação deste trabalho, esses resultados são devidos a uma maior qualidade dos modelos de segmentação utilizados. Porém, como dito antes, tais resultados trazem consigo o problema da inviabilidade de utilização dos respectivos métodos em um sistema em tempo real ou *online*.

5.2.4.1 Antecipação

O *BStar iRGB*$_{online}$ alcançou uma acurácia média de antecipação de 77,16%, utilizando um limiar $u = 0{,}2$. Considerando-se a complexidade do problema, mesmo

Tabela 27 – Comparando os resultados obtidos pela proposta apresentada com os dos trabalhos que objetivaram segmentar e reconhecer os gestos do conjunto de dados Montalbano. Todos os resultados estão dados em termos do índice de Jaccard.

Trabalho/Proposta	Tipo de dados	Resultado
Molchanov et al. (2016)	RGB-D e Fluxo óptico	0,980
Pigou et al. (2018)	Esqueleto e RGB-D	0,910
Gupta et al. (2019)	RGB-D e Fluxo óptico	0,910
Neverova et al. (2016)	Áudio, RGB-D e Esqueleto	0,881
Hosseini, Montagne e Hammer (2019)	Esqueleto	0,852
Neverova et al. (2014)	RGB-D e Esqueleto	0,850*
Monnier, German e Ost (2014)	RGB-D e Esqueleto	0,834*
Chang (2014)	RGB e Esqueleto	0,826*
BStar iRGB$_{hand}$	**RGB**	**0.807**
Peng et al. (2014)	RGB	0,791*
Pigou et al. (2014)	RGB-D	0,788*
Chen et al. (2014)	RGB-D	0,648*
Wu e Shao (2014)	RGB-D	0,628*

* Resultados apresentados na competição *Chalearn 2014: Looking At people (Track 3: Gesture Recognition)*(ESCALERA et al., 2014).

Fonte: Próprio autor.

o resultado não sendo tão alto quanto as acurácias de reconhecimento, ele pode ser considerado satisfatório. Além do mais, como visto na Tabela 25 da Seção 4.6, são poucos os trabalhos que apresentaram resultados para a antecipação de ações no referido conjunto de dados e, mesmo utilizando apenas informação de cor, estes resultados são bem competitivos.

Assim como na tarefa de reconhecimento, aqui, em muitos dos vídeos, também é notória a efetividade do modelo na antecipação de gestos. A execução do modelo de antecipação sobre alguns vídeos do conjunto de testes do Montalbano pode ser visualizada no seguinte *link*: <https://youtu.be/8dUt46LARlI>.

É importante destacar que, como a proposta apresentada possui um modelo de *spotting*, os resultados poderiam ser melhorados caso se optasse pelo reconhecimento (utilizar a predição na última observação) sempre que a antecipação não ocorresse até o final da sequência segmentada. Essa é uma decisão a ser tomada pelo projetista do sistema interacional e fica aqui apenas como uma observação (sugestão).

5.2.4.2 Tempo de resposta

Quanto ao tempo de resposta, para cada *frame* de entrada, o sistema completo (*spotting* e classificação) teve um tempo médio de resposta de $1.423ms$ quando executado em CPU e de $95ms$ quando executado em GPU. Tendo em mente que o conjunto de testes possui uma taxa de exibição de 20 FPS, mesmo em GPU, o sistema não conseguiria

Tabela 28 – Tempos médios de resposta do sistema calculados para a predição dos gestos no conjunto de dados de teste do Montalbano utilizando o *BStar iRGB*$_{online}$. Os tempos considerados foram somente dos instantes em que os dois modelos agiram em conjunto. Observe-se que apenas o modelo e o *Openpose* podem ser executados em CPU ou em GPU. O processo de cálculo do *Star iRGB*, extração das mãos e os serviços de comunicação sempre são executados em CPU. Todos os valores estão arredondados para o maior inteiro mais próximo.

Dispositivo	*Spotting e Classificação*	Tempo Médio (ms) *Openpose*	Mãos	*Star iRGB*	Comunicação	Resposta
CPU	148	1262	1	1	11	1423
GPU	19	63	-	-	-	95

Fonte: Próprio autor.

processar todas as imagens sem que houvesse descartes ou atrasos no tempo de resposta. No entanto, considerando-se as abordagens apresentadas nos capítulos anteriores, onde essa taxa era de 10 FPS, o mesmo seria, possivelmente, capaz de processar todos os *frames* sem perdas substanciais na qualidade do reconhecimento e antecipação dos gestos.

Assim como indicado antes, a mudança do *Openpose*, parte do sistema que demanda mais processamento, por um modelo de detecção de mãos mais simples, provavelmente traria uma melhora considerável para o tempo de resposta do sistema, fazendo-o responder adequadamente mesmo a uma *stream* de vídeo com uma taxa de 20 FPS. Por fim, perceba-se que essa proposta, como suposto, mesmo possuindo dois classificadores distintos, demandou quase o mesmo tempo de processamento do *BStar iRGB* ($93ms$). A Tabela 28 traz um resumo de todos esses tempos.

5.3 Aplicação em um ambiente inteligente multicâmeras

Nesta seção será proposto um modelo que aplicará muito do que foi proposto neste trabalho a fim de reconhecer e antecipar gestos em um ambiente multicâmeras. O intuito aqui é mostrar que as propostas já apresentadas podem ser utilizadas para a criação de um sistema interacional a ser executado em um ambiente real. Oferecendo-se, assim, uma interface intuitiva que possa ser utilizada na interação, por exemplo, entre humanos e robôs. Esta proposta utilizará partes das principais propostas apresentadas neste e nos capítulos anteriores, as quais serão indicadas à medida que forem sendo apresentadas.

5.3.1 Proposta

O reconhecedor de gestos desta proposta está dividido em três partes principais, as quais serão explicadas em seguida, a saber: informação de movimento e de contexto,

extração e seleção de características, segmentação temporal do movimento e classificação dos gestos presentes nas sequências segmentadas.

5.3.1.1 Informação de movimento e de contexto

Considerando-se um espaço inteligente com N câmeras, a utilização do *Star iRGB* resultaria em N representações *Star iRGB* a cada instante de tempo. Dessa forma, seguindo as propostas anteriores, ao usar uma CNN para extrair características de cada imagem tornar-se-ia muito custoso processar as N imagens. Por isso, baseando-se na proposta da Seção 4.4, a informação do esqueleto do indivíduo, por fornecer informações de movimento com menos informações que uma imagem, pode ser mais eficiente do ponto de vista computacional. Mesmo assim, um indivíduo poderia ter N representações de esqueletos bidimensionais (2D), um para cada câmera. Isso possivelmente geraria muito ruído para o modelo, uma vez que, a depender da posição e orientação da câmera, cada esqueleto possuiria uma orientação, uma posição e uma escala diferente dos demais. Dessa forma, uma solução seria transformar todos os esqueletos para o referencial de uma das imagens, ou obter uma representação única do esqueleto no referencial do mundo. Sendo assim, decidiu-se pela segunda opção, um vez que a primeira, além de ser custosa e levantar a questão de qual imagem escolher como referencial, ofereceria múltiplas entradas para o modelo. Com isso, a extração de esqueletos utilizada será a abordagem descrita por Queiroz et al. (2018). Desse modo, a cada instante de tempo, as imagens capturadas pelas N câmeras são passadas para um reconstrutor de esqueletos que fornece uma representação tridimensional (3D) do esqueleto do usuário presente na cena. Cada coordenada 3D das juntas estará representada no sistema de coordenadas no qual as câmeras foram calibradas. Assim como na antecipação de ações do Capítulo 4, o movimento será representado pela evolução das juntas do esqueleto no tempo.

Recorrendo-se aos modelos propostos *BStar iRGB*$_{hand}$ e *BStar iRGB*$_{online}$, a informação de contexto será representada por imagens contendo a área das mãos dos usuários. Um problema que surge aqui é que para cada câmera, têm-se duas imagens, uma para o recorte da mão esquerda e outra para o da mão direta. Dessa forma, para as N câmeras pode-se ter até N imagens representando a mão esquerda e N representando a direita. Nesse sentido, as mãos serão recortadas tal como feito para o *BStar iRGB*$_{hand}$ e o *BStar iRGB*$_{online}$, porém, ao invés de juntá-las em apenas uma imagem, a proposta aqui é dividi-las em duas imagens, uma contendo os recortes da mão direita e outra os da mão esquerda.

Um problema enfrentado é a impossibilidade de extração da região de ambas as mãos em todas as imagens devido à não detecção da junta do punho em algumas delas. A falta de detecção de algumas juntas pode dar-se devido às incertezas do modelo, ou mesmo pela oclusão das mãos em algumas câmeras. Assim, objetivando uma solução

rápida, propõe-se estabelecer que, sempre que a junta do punho não for detectada, ou quando ela for detectada com um grau de confiança menor que 0,5, ao invés da imagem da mão, o recorte terá todos os seus píxeis iguais a zero. Uma observação importante é que, no Montalbano, devido à posição da câmera, as mãos eram sempre vistas de frente, e nunca eram perdidas. No entanto, aqui, devido aos diferentes ângulos de visão das câmeras, em algumas imagens pelo menos uma das mãos estarão oclusas. Isso, claramente, incrementa ainda mais o grau de dificuldade do problema que a proposta deverá lidar.

5.3.1.2 Extração e seleção de características

A proposta de incorporação de características (*embedding*) apresentado na Seção 4.4 mostrou-se eficaz na tarefa de extrair e selecionar as informações de movimento e de contexto que melhor representassem o problema de antecipação de ações. Além disso, lá, foi utilizado apenas as sete juntas que representam a cabeça e os membros superiores do indivíduo. Dessa maneira, uma proposta similar será apresentada aqui.

A proposta consiste em aplicar um operador de vetorização sobre as 7 juntas do esqueleto (cababeça, ombros, cotovelos e punhos), obtendo-se uma entrada $\mathbf{k} \in \mathbb{R}^{21\times1}$. Assim, por meio de duas matrizes de pesos treináveis $\mathbf{W}_{m1} \in \mathbb{R}^{21\times256}$ e $\mathbf{W}_{m2} \in \mathbb{R}^{256\times256}$, e dois vetores de *bias* $\mathbf{b}_{m1}$ e $\mathbf{b}_{m2} \in \mathbb{R}^{256\times1}$, após a operação $f(\mathbf{W}_{m2}^T f(\mathbf{W}_{m1}^T \mathbf{k} + \mathbf{b}_{m1}) + \mathbf{b}_{m2})$, tem-se a representação de movimento $\mathbf{e}_m \in \mathbb{R}^{256\times1}$ do esqueleto, onde f é a função de ativação ReLU.

Para garantir o balanceamento entre as características (veja-se a discussão da Seção 4.4.2), a proposta de extrator de características das mãos também deverá resultar em um vetor $\mathbf{e}_c \in \mathbb{R}^{256\times1}$. Seguindo a linha de propostas anteriores, a *Resnet 50* juntamente com um filtro unidimensional $\mathbb{R}^{3\times1}$, forneceria a informação de contexto (mãos) codificada em um vetor $\mathbf{e}_c \in \mathbb{R}^{1023\times1}$, o que necessitaria de uma transformação $g : \mathbb{R}^{1023\times1} \mapsto \mathbb{R}^{256\times1}$ para ficar com a mesma dimensão do vetor $\mathbf{e}_m$. Para facilitar esse processo, decidiu-se utilizar a *Resnet 34* ao invés da *Resnet 50*. A *Resnet 34*, além de ser mais rápida e a ainda possuir um bom extrator de características, possui uma dimensionalidade menor na saída do seu último mapa características, que, após a aplicação do operador GAP, resulta no vetor $\mathbf{e}_{gap} \in \mathbb{R}^{512\times1}$. Dessa forma, utilizando uma camada de *embedding* $f(\mathbf{W}_c^T \mathbf{e}_{gap}^T + \mathbf{b}_c)$, com $\mathbf{W}_c \in \mathbb{R}^{512\times256}$ e $\mathbf{b}_c \in \mathbb{R}^{256\times1}$, tem-se um vetor de características $\mathbf{e}_c \in \mathbb{R}^{256\times1}$.

Porém, a informação de contexto é representada por meio de duas imagens, uma com os recortes das mãos esquerdas e outra com os recortes das mãos direitas. Dessa forma, propõe-se utilizar duas *Resnet 34* com pesos compartilhados para extrair as características das duas imagens. Isso resulta em dois vetores $\mathbf{e}_{gap1}$ e $\mathbf{e}_{gap2} \in \mathbb{R}^{512\times1}$. O compartilhamento dos pesos das duas CNNs, além de economizar tempo de treinamento, fará com que elas sejam capazes de diferenciar as características de cada imagem, de maneira a extrair

aquelas que sejam mais relevantes para o problema como um todo. Usar duas CNNs sem compartilhamento de pesos, faria com que cada uma extraísse as características que achasse mais relevante para sí, sem levar em consideração diretamente as características da outra. Isso poderia gerar mais dificuldades de treinamento para o SGD. A utilização de tal abordagem foi inspirada nas redes siamesas, amplamente utilizada no problema de reconhecimento facial. Uma leitura mais completa sobre tal arquitetura pode ser vista em Koch, Zemel e Salakhutdinov (2015).

Após extraídas as características, por meio de um operador *soft-attention*, os dois vetores ($\mathbf{e}_{gap1}$ e $\mathbf{e}_{gap2}$) serão fundidos e só então será obtido o vetor $\mathbf{e}_c \in \mathbb{R}^{256\times1}$. Desse modo, têm-se agora dois vetores com a mesma dimensionalidade, um que representa a informação de movimento ($\mathbf{e}_m$) e outro que representa a informação de contexto ($\mathbf{e}_c$). Assim, seguindo o proposto anteriormente na Seção 5.2, os dois vetores serão fundidos por um outro operador *soft-attention* que resultará num vetor $\mathbf{e}_{soft} \in \mathbb{R}^{256\times1}$. Note-se que o primeiro operador *soft-attention* será encarregado de fundir as informações das duas mãos, enquanto o segundo se encarregará de fundir as informações de contexto e de movimento.

5.3.1.3 Segmentação temporal do movimento e classificação

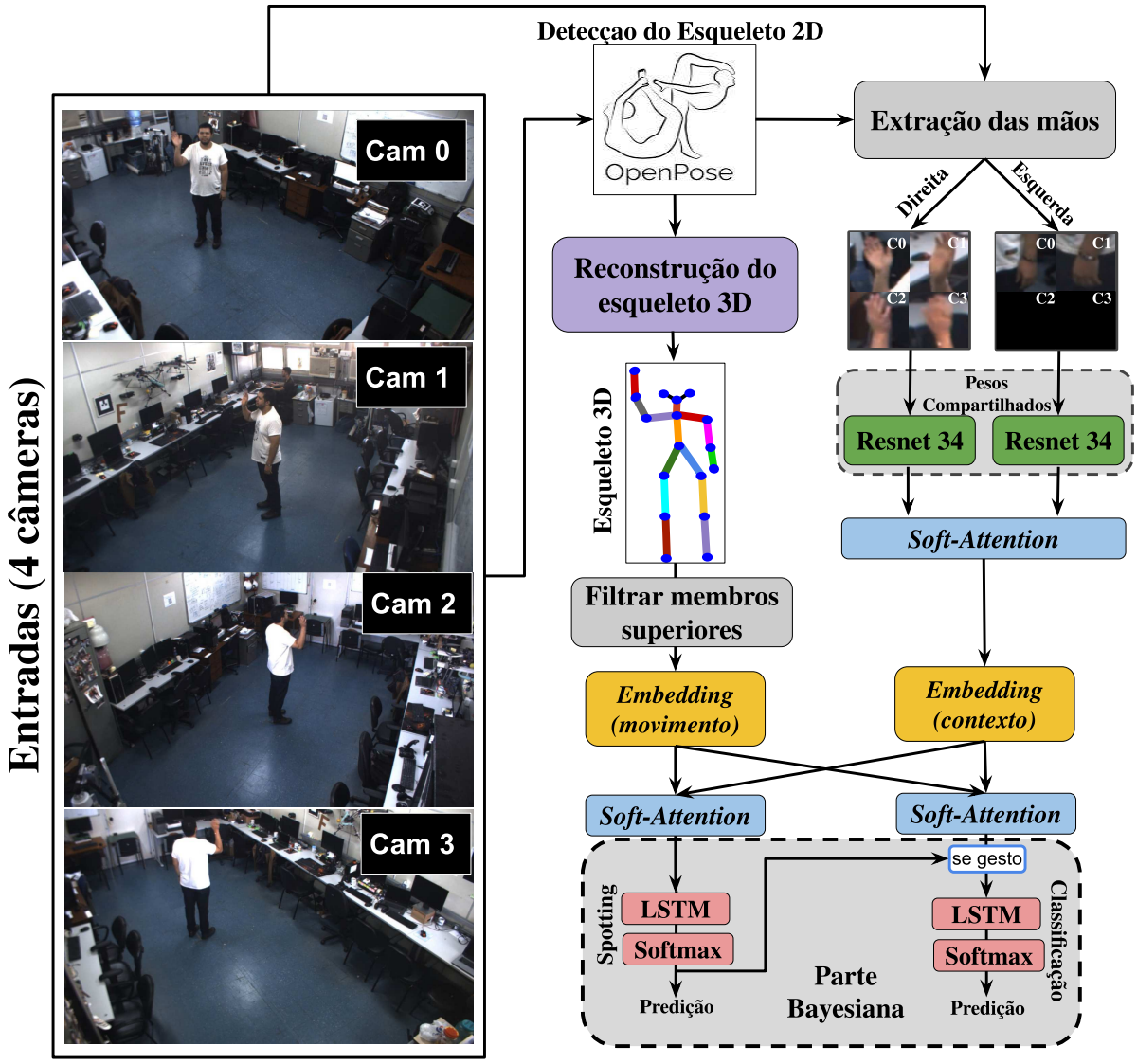
O modelo segmentador (*spotting*) e classificador serão baseados na proposta anterior (*BStar* $iRGB_{online}$). Dessa maneira, o vetor $\mathbf{e}_{soft}$ contendo a informação de movimento e de contexto será dado como entrada para uma LSTM de camada única com 128 neurônios escondidos, seguida de um classificador *softmax* com apenas duas saídas. Esse modelo realizará a tarefa de *spotting*. Para o modelo classificador, um modelo idêntico será utilizado, no entanto com d saídas, onde d representa a quantidade de classes de gestos do conjunto de dados utilizado. No entanto, assim como no *BStar* $iRGB_{online}$, os dois modelos compartilharão a parte de extração e seleção de características, mas terão os seus próprios operadores de *soft-attention* para decidir qual informação é a mais importante para a sua tarefa: o movimento ou o contexto. Perceba-se que até aqui são três os operadores de *soft-attention*: um para fundir as informações das mãos, e dois para fundir as informações de movimento e de contexto; sendo um para o modelo de *spotting* e outro para o modelo de classificação.

O funcionamento dos dois modelos se dará como anteriormente: o modelo classificador só será acionado quando o segmentador predisser um *gesto*, e será desativado quando o mesmo detectar um *não-gesto*. Para possibilitar que a antecipação ocorra por meio de um limiar sobre a incerteza da predição, as LSTMs dos dois modelos serão estocásticas. Para isso, também será utilizada a abordagem *MC dropout*.

Para melhor identificação, essa proposta será chamada de $BSKL_{hand}$, onde o B significa Bayesiano, o SKL é uma abreviação da palavra esqueleto em inglês (*skeleton*) e

hand significa que a informação de contexto é as mãos. Na Figura 55 pode ser vista uma ilustração completa do $BSKL_{hand}$ para um espaço inteligente com 4 câmeras.

Figura 55 – Arquitetura do modelo $BSKL_{hand}$ para um espaço inteligente com 4 câmeras.

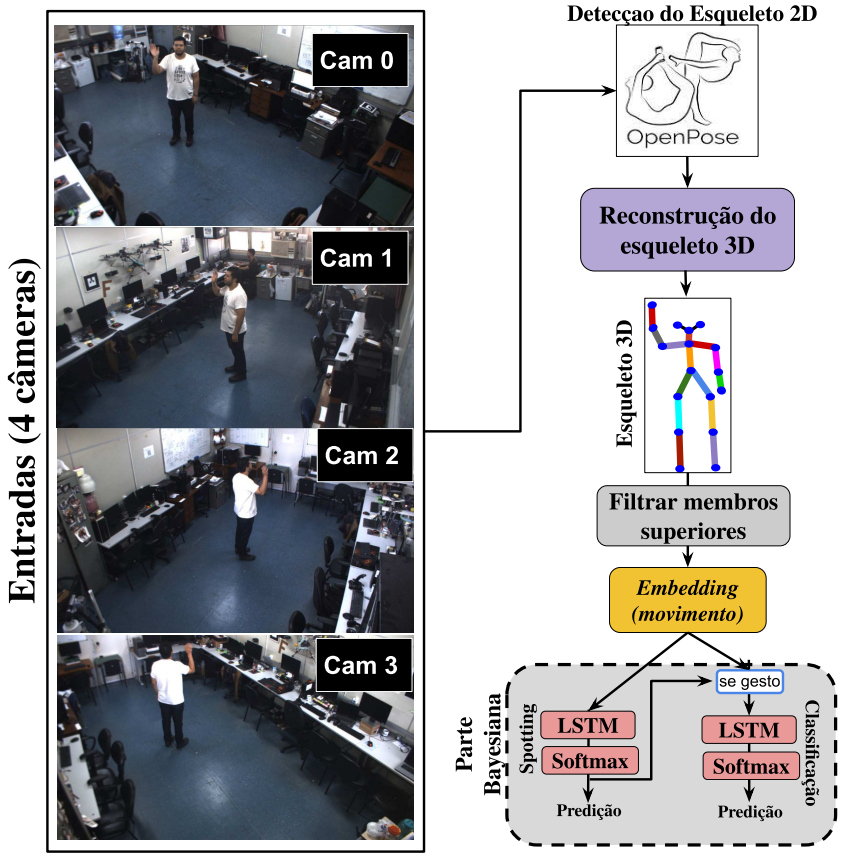


Fonte: Próprio autor.

Para poder avaliar o ganho do uso da informação de contexto nesse novo problema, propõe-se como *baseline* uma versão do modelo proposto, porém que utiliza apenas a informação de movimento extraída do esqueleto 3D. Esse modelo será chamado apenas de *BSKL* e possuirá a seguinte arquitetura: um *embedding* igual a do $BSKL_{hand}$ para a extração e seleção de características do esqueleto 3d, uma LSTM de camada única com 128 neurônios escondidos, seguida por uma camada totalmente conectada com 2 neurônios de saída que ativam uma função *softmax*. Esse será o modelo de *spotting*. A saída do *embedding* será compartilhada com uma estrutura similar que fará o papel do classificador. Ele será formado de uma LSTM de camada única com 128 neurônios escondidos, seguida

por uma camada totalmente conectada com d neurônios de saída que ativam uma função *softmax*. Aqui, d também representa o número de gestos do conjunto de dados utilizado. Assim como em $BSKL_{hand}$, o modelo classificador só será ativado quando o segmentador identificar um *gesto* e será inativado quando o segmentador identificar um *não-gesto*. Para melhor visualização, a Figura 56 trás uma ilustração desse modelo de *baseline*.

Figura 56 – Arquitetura do modelo de *baseline BSKL* para um espaço inteligente com 4 câmeras.



Fonte: Próprio autor.

5.3.2 Experimentos

O espaço inteligente utilizado está descrito em Almonfrey et al. (2018) e Carmo et al. (2019). Nele, existem quatro câmeras calibradas que se comunicam com o espaço através de uma infraestrutura física em rede e uma infraestrutura de *software* baseada em microsserviços. Como característica, o espaço inteligente fornece as imagens capturadas das suas câmeras em uma taxa de 10 FPS. Para fins dos experimentos, considera-se que todos os usuários estão num mesmo plano do espaço.

5.3.2.1 Conjunto de dados

Para a realização deste trabalho foi capturado um conjunto de dados composto por 1.080 vídeos (270 para cada câmera) que totalizam 2.160 gestos distribuídos entre 15 classes distintas. Cada vídeo tem um resolução de 1288×728 píxels e corresponde a um gesto realizado oito vezes por um mesmo voluntário. Como o espaço utilizado possui quatro câmeras, os voluntários foram instruídos a realizar cada um dos 15 gestos oito vezes: de frente para cada uma das quatro câmeras e de frente para um ponto entre elas. No total, foram realizadas aquisições com 18 voluntários distintos. Tais aquisições foram feitas no espaço inteligente localizado no Instituto Federal do Espírito Santo, campus Vitória (Ifes-Vitória), e serão utilizadas para o treinamento e validação dos modelos propostos. O conjunto de treino possuirá 1020 gestos capturados de 16 voluntários escolhidos aleatoriamente, já o de validação, possuirá 240 gestos correspondentes aos dois voluntários restantes. Deve-se salientar que antes de cada aquisição de dados os voluntários assistiram a um vídeo demonstrativo de como cada gesto deveria ser executado. No entanto, mesmo executando-os diferentemente do indicado, nenhuma das aquisições foi interrompida. É importe mencionar também que essa captura dos dados se deu por meio de termo de sigilo que não permite a divulgação do mesmo. Por esse motivo, a base ainda não está disponível publicamente.

Inicialmente, o principal objetivo era treinar o modelo proposto com conjunto de dados levantados no espaço inteligente do Ifes, e testá-lo por meio de uma aplicação em um segundo espaço com as mesmas características, localizado na Universidade Federal do Espírito Santo, campus Goiabeiras (Ufes). No entanto, por conta da pandemia do vírus Sars-Cov-2, os experimentos reais foram descartados, devido à impossibilidade de comparecimento físico de muitas pessoas ao laboratório. Dessa forma, para que fosse possível obter resultados os mais próximos possíveis dos que seriam obtidos em uma aplicação real, foram realizadas mais duas aquisições (240 gestos), com apenas dois usuários, utilizando a infraestrutura do espaço inteligente da Ufes. Assim, os dados levantados nessa aquisição serão utilizados como conjunto de teste.

Usar um conjunto de teste capturado em um outro espaço, com dimensões, ângulos de captura e qualidade de iluminação distinta do espaço onde foi capturado o conjunto de treino, ajudará na validação da capacidade de generalização da proposta. Sendo assim, o conjunto de dados completo (treino, teste e validação) é formado por 2.400 vídeos, correspondentes a 300 aquisições de 20 indivíduos distintos, e a ele será dado o nome de IS-Gesture.

Todos os vídeos do IS-Gesture foram rotulados de maneira que cada *frame* contivesse o rótulo do gesto ao qual ele pertencia: *não-gesto* (classe 0) ou *gesto* (classes de 1 a 15). Uma lista do vocabulário de gestos presentes no conjunto de dados é apresentada na

Tabela 29 e uma ilustração de cada gesto é apresentada na Figura 57.

Tabela 29 – Lista dos gestos presentes no conjunto de dados IS-Gesture. A classe 0 corresponde a um não-gesto, ou seja, quando nenhum gesto está sendo executado.

Classe	Gesto	Classe	Gesto
0	Não-gesto	8	Não (permissão)
1	Pedido de ajuda	9	Ruim (*feedback*)
2	Venha aqui	10	Dar passagem
3	Pode sair	11	Apontar
4	Siga-me	12	Dúvida
5	Pare	13	Mais alto (volume)
6	Abortar (missão/tarefa)	14	Mais baixo (volume)
7	Bom (*feedback*)	15	Silêncio

Fonte: Próprio autor.

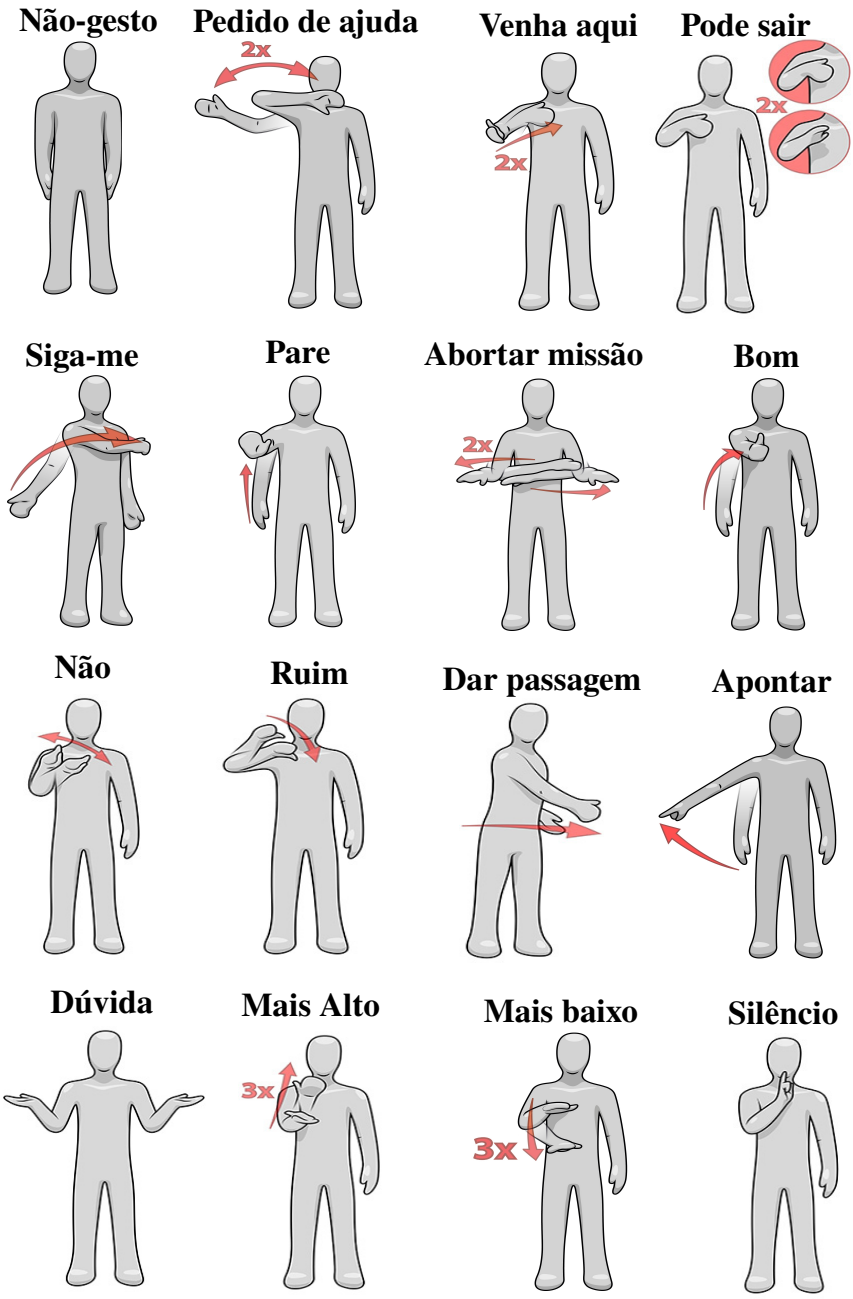
5.3.2.2 Pré-processamento dos dados

As imagens capturadas pelas quatro câmeras num mesmo instante de tempo foram dadas de entrada para um serviço do espaço inteligente que implementa o *Openpose*. Assim, as saídas desse serviço, os esqueletos 2D detectados em cada imagem, foram dadas como entrada para um outro serviço responsável pela recontrução do esqueleto 3D. Sendo assim, esse processo foi aplicado apenas uma vez sobre os conjuntos de treino e validação, e gerou 111.161 esqueletos 3D correspondentes aos 270 vídeos, e 111.161 pares de imagens, cada uma com quatro recortes contendo a informação das mãos extraídas de cada imagem. Considerando os ruídos que podem ser gerados na etapa de reconstrução do esqueleto 3D, foi aplicada uma rotina de supressão de não máximos a fim de eliminar os esqueletos com juntas menos prováveis. Na Figura 58, pode ser vista a reconstrução de um esqueleto tridimensional a partir de 4 imagens capturadas em um instante de tempo qualquer, e dos esqueletos bidimensionais nelas detectados.

Como o modelo de esqueleto do *Openpose* não possui a junta que representa o centro do torso (tratada aqui apenas como junta do torso), esta foi criada como sendo o ponto central do triângulo formado pelas juntas do tórax e dos quadris. Dessa forma, cada esqueleto foi centralizado em torno junta do torso e tiveram os comprimentos entre juntas normalizados de maneira a terem norma unitária, assim como proposto em Queiroz (2019).

Em seguida, como as juntas dos membros inferiores comumente não trazem informações relevantes para a diferenciação dos gestos utilizados, optou-se por selecionar apenas sete juntas correspondente aos punhos, cotovelos, ombros e cabeça. Por fim, após a vetorização dos pontos das juntas, o vetor resultante foi normalizado para uma distribuição

Figura 57 – Ilustração de cada um dos gestos do conjunto de dados IS-Gesture.



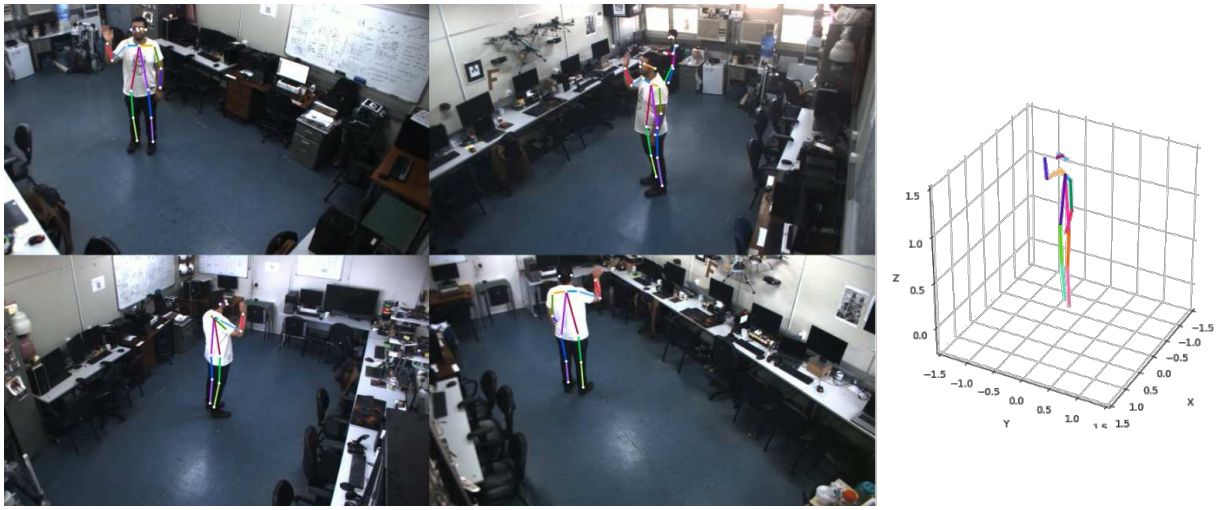
Fonte: Próprio autor.

Z-score (média zero e variância unitária). Dessa forma, a entrada para ambos os modelos propostos é um vetor normalizado $\mathbf{k} \in \mathbb{R}^{21\times1}$.

5.3.2.3 Treinamento

Primeiro, o modelo classificador foi treinado até atingir um valor satisfatório sobre o conjunto de validação, e depois, após congelar os pesos do segundo *soft-attention* e da *Resnet 34*, o modelo segmentador foi treinado; como proposto para o *BStar iRGB*$_{online}$.

Figura 58 – Exemplo da reconstrução de um esqueleto 3D a partir de quatro imagens adquiridas no espaço inteligente da Ufes.



Fonte: Próprio autor.

Dessa maneira, os dois modelos possuem estruturas distintas quanto ao *embedding* de características, mas possuem a mesma estrutura de extração das características que representam as informações de contexto. O mesmo processo foi utilizado no treinamento do *BSKL*.

Como cada proposta tem dois modelos de naturezas distintas, o treinamento de cada um também foi diferente, a saber: (*i*) para o treinamento do classificador, foram utilizadas apenas as sequências segmentadas (aquelas que contêm apenas os gestos) e os seus correspondentes rótulos de classe; (*ii*) já para o treinamento do segmentador (classificador binário), as observações com rótulo 0 foram consideradas *não-gestos* (classe 0) e as com rótulo maior que 0 foram consideras como *gestos* (classe 1).

Assim como na maioria das propostas anteriores, utilizou-se também um processo de Otimização Bayesiana para determinar a configuração do hiperparâmetros que maximizasse a acurácia obtida sobre o conjunto de validação do IS-Gesture. Como este é um conjunto de dados relativamente pequeno e muito complexo, após a obtenção do melhor conjunto de hiperparâmetros para cada modelo, eles foram treinados utilizando o conjunto de treino e validação juntos, até que convergissem. Nesse casso, a convergência foi representada por duas condições de *earling stopping*, onde o processo de treinamento parava quando: o modelo passava mais de cinco épocas sem variar pelo menos 1% na acurácia de treinamento ou quando o número máximo de épocas fosse atingido. Em complemento, como cada um dos 300 vídeos contém uma aquisição correspondente a oito ou mais execuções do mesmo gesto de um mesmo voluntário, para que o modelo não fosse direcionado a repetir a mesma predição, cada vídeo foi dividido em sequências aleatórias. O tamanho dessa sequência também era um hiperparâmetro, e pode ser visto, assim como os demais, na Tabela 30.

Tabela 30 – Hiperparâmetros utilizados no treinamento dos modelos propostos.

Hiperparâmetros	Modelos	
	Spotting	Classificação
Tamanho da Sequência	256	32
Tamanho da subsequência	128	16
Número de épocas	100	300
Tamanho do lote	128	96
Taxa de aprendizagem	1e-3	1e-2
Taxa de decaimento L2	1e-5	1e-5
Truncamento do gradiente (Norma)	5,0	5,0
Otimizador	adam	adam

Fonte: Próprio autor.

O processo de treinamento e busca pelos melhores hiperparâmetros durou 22 dias para o $BSKL_{hand}$ e 2 dias para o *BSKL*. E para isso, utilizaram-se novamente as mesmas infraestruturas de *hardware* e *software* dos experimentos da Seção 4.5.2.

5.3.2.4 Predição

A predição será feita como no *BStar* $iRGB_{online}$, explicada na Seção 5.2.2.1. No entanto, para isso, o modelo treinado será implementado na forma de serviço do espaço inteligente. Dessa forma, por meio de uma simulação do espaço inteligente em funcionamento, poder-se-á avaliar o desempenho da proposta. Para esse fim, serão utilizadas as mesmas métricas apresentadas na Seção 5.2.2.2.

5.3.3 Resultados e discussões

Após a realização dos experimentos, os resultados foram analisados e serão descritos na sequência. Primeiro serão analisados os resultados no modelo de *baseline BSKL* e depois os do $BSKL_{hand}$. Assim, será possível perceber os principais ganhos com a proposta que inclui a informação de contexto, representada pelas imagens das mãos.

5.3.3.1 Modelo de *baseline*

Após treinado, o segmentador do *BSKL* alcançou uma acurácia média de 82% sobre a base de testes do IS-Gesture. Considerando-se a média móvel utilizada, o segmentador pode estar realizando a predição três *frames* após o início e o fim do gesto, o que gera, em teoria, um erro de pelo menos seis observações por gesto. Dessa maneira, em um cálculo grosseiro, como o conjunto de teste possui 240 gestos e 18.932 observações (somadas as

dos *gestos* e as dos *não-gestos*), tem-se um erro esperado de pelo menos 7%. Fazendo com que os 82% de acurácia alcançado não seja de todo ruim.

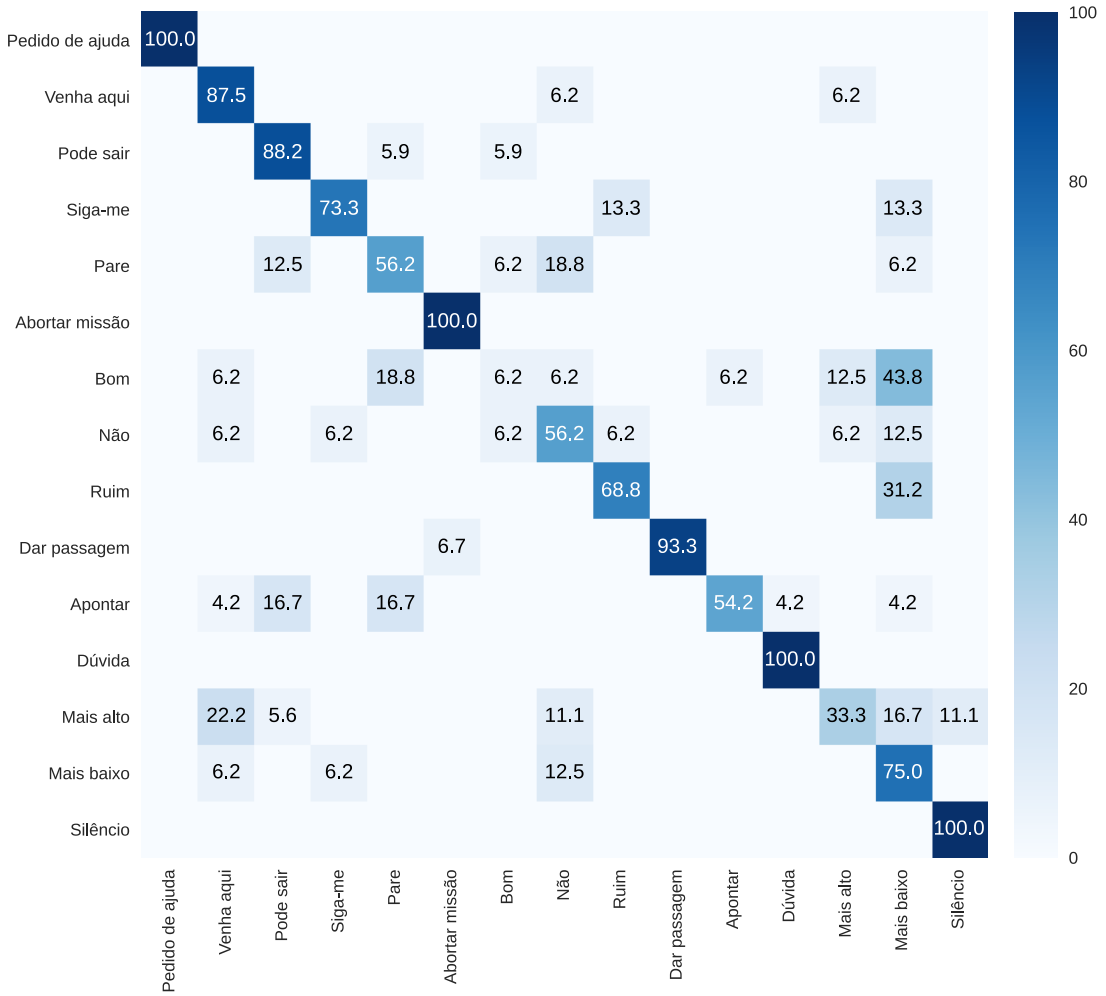
Na Figura 59, é possível ver a matriz de confusão com os resultados do classificador. A acurácia média obtida na classificação dos gestos foi de 72,40%. Note-se que os gestos *pedido de ajuda*, *abortar missão*, *dúvida* e *silêncio* alcançaram a acurácia máxima de 100%. Os gestos *venha aqui* e *pode ir* alcançaram quase 90%, enquanto que o gesto *dar passagem* ultrapassou os 93%. O pior resultado foi para o gesto *bom*, que alcançou apenas 6,25%. Assim, mesmo com altos valores de acurácia para algumas classes, no geral, pode-se considerar que a proposta não alcançou resultados satisfatórios. Isso se deve ao fato de que o vocabulário de gestos utilizado possui gestos ambíguos que dependem não só do movimento dos braços, mas também da forma da mão. Dessa maneira, os gestos que podem ser diferenciados apenas com o movimento alcançaram a acurácia máxima. Por outro lado, aqueles mais dependentes da forma da mão foram confundidos uns com os outros, como já era esperado. Como exemplo, o que diferencia o gesto *bom* dos gestos *mais baixo* e *pare* é a forma das mãos, uma vez que todos possuem movimentos muito semelhantes. No gesto *bom* (Figura 60 a)), a mão está fechada e com o polegar apontando para cima, enquanto que em uma variante do gesto *mais baixo* (Figura 60 b)) a mão está aberta com palma inicialmente para cima e depois para baixo. Já no gesto *pare* (Figura 60 c)), o movimento é o mesmo do gesto *bom*, no entanto, no *pare*, a mão fica com a palma aberta estendida para frente.

Considerando-se a operação conjunta de ambos os modelos, a acurácia de reconhecimento das 16 classes (15 gestos e 1 não-gesto) foi de 74,60%. Agora, em relação ao índice de Jaccard, o valor obtido foi de 0,251. Isso significa que dentre as predições do modelo de *spotting* pertencentes à classe *gesto*, apenas 25% foram classificadas corretamente pelo classificador. Apesar do resultado não ser muito alto, isso não quer dizer que o modelo não consegue predizer, uma vez que se as predições corretas estiverem na parte final de cada sequência segmentada, ainda é possível reconhecer o gesto na última observação. Isso vai depender da qualidade do modelo de *spotting* de segmentar as observações que contém o gesto corretamente. Isso pode ser visto quando avaliada a capacidade de antecipação. Nesse sentido, utilizando um limiar de incerteza de $u = 1{,}4$, o modelo foi capaz de antecipar corretamente 42% dos gestos utilizando em média 21% das observações. Perceba-se que as duas métricas (acurácia e índice de Jaccard) precisam ser avaliadas cuidadosamente, pois representam coisas distintas. Apenas uma definição precisa do que se quer da aplicação será capaz de determinar qual delas é a mais adequada.

5.3.3.2 Modelo proposto

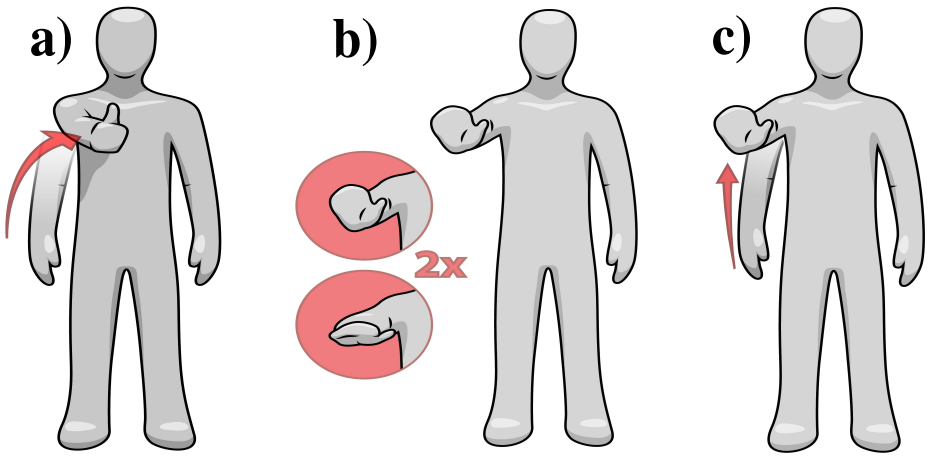
Após o treinamento, o segmentador do $BSKL_{hand}$ alcançou uma acurácia média de 92,68% sobre a base de testes do IS-Gesture. Esse valor é 13% maior que o obtido pelo

Figura 59 – Matriz de confusão com os resultados do classificador de gestos do modelo *BSKL*.



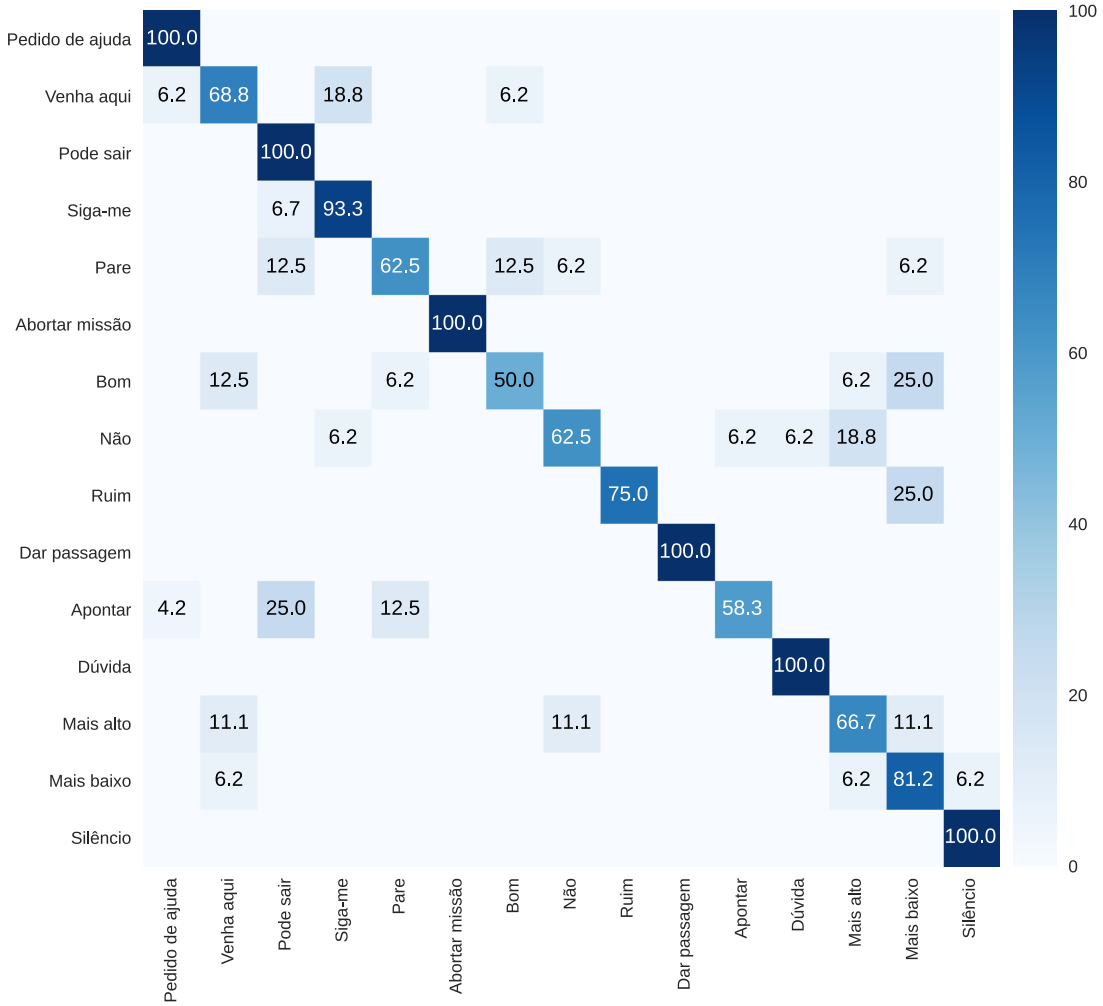
Fonte: Próprio autor.

Figura 60 – Exemplo de gestos que podem ser confundidos. a) gesto *bom*, b) gesto *mais baixo* e c) gesto *pare*. Nessas classes de gestos, o que as difere é a forma da mão.



Fonte: Próprio autor.

Figura 61 – Matriz de confusão com os resultados do classificador de gestos do modelo $BSKL_{hand}$.



Fonte: Próprio autor.

segmentador do BSKL (82%) que não utiliza informação contextual. Da mesma forma, o modelo classificador alcançou uma acurácia de reconhecimento sobre a predição da última observação de cada sequência de 80,04%, o qual é 10% maior que os 72,40% do classificador do *BSKL*. Na Figura 61, é possível ver a matriz de confusão com os resultados do classificador do $BSKL_{hand}$. Note-se que seis gestos alcançaram a acurácia máxima de reconhecimento de 100%: *pedido de ajuda*, *abortar missão*, *dúvida* e *silêncio*. Sendo que o pior resultado também foi o do gesto *bom*, com 50,00%. Porém, perceba-se que mesmo sendo o pior resultado, ele é 43,25% maior que o alcançado para o mesmo gesto com o *BSKL*.

Para uma análise mais completa, a Tabela 31 apresenta os resultados de ambos os modelos $BSKL_{hand}$ e *BSKL* para cada classe de gesto, bem como a diferença entre eles. Observe-se que apenas o gesto *venha aqui* obteve pior resultado com o $BSKL_{hand}$ (68,75%) que com o *BSKL* (87,50%). Os demais gestos obtiveram ou resultados iguais (*pedido de*

ajuda, *abortar missão*, *dúvida* e *silêncio*), ou resultados bem superiores. Sendo que os mais significativos foram os dos gestos *pode sair*, *siga-me*, *bom* e *mais alto*, um incremento de 21,76%, 20,00%, 43,75% e 33,34%, respectivamente. Frise-se que esses gestos são os mais dependentes da forma da mão. Dessa forma, o gesto *bom*, anteriormente discutido como dependente da informação da mão, obteve um aumento de 43,75%. Isso reafirma a importância da utilização da informação de contexto.

Tabela 31 – Comparação dos resultados obtidos pelos modelos $BLSKL_{hand}$ e $BLSKL$ para cada classe de gestos do conjunto de dados IS-Gesture.

Gesto	$BLSKL_{hand}$ (%)	$BLSKL$ (%)	Diferença (%)
Pedido de ajuda	100,00	100,00	00,00
Venha aqui	68,75	87,50	-18,75
Pode sair	100,00	88,24	21,76
Siga-me	93,33	73,33	20,00
Pare	62,50	56,25	06,25
Abortar missão	100,00	100,00	00,00
Bom	50,00	06,25	43,75
Não	62,50	56,25	06,25
Ruim	75,00	68,75	06,25
Dar passagem	100,00	93,33	06,67
Apontar	58,33	54,17	04,16
Dúvida	100,00	100,00	00,00
Mais alto	66,67	33,33	33,34
Mais baixo	81,25	75,00	06,75
Silêncio	100,00	100,00	00,00

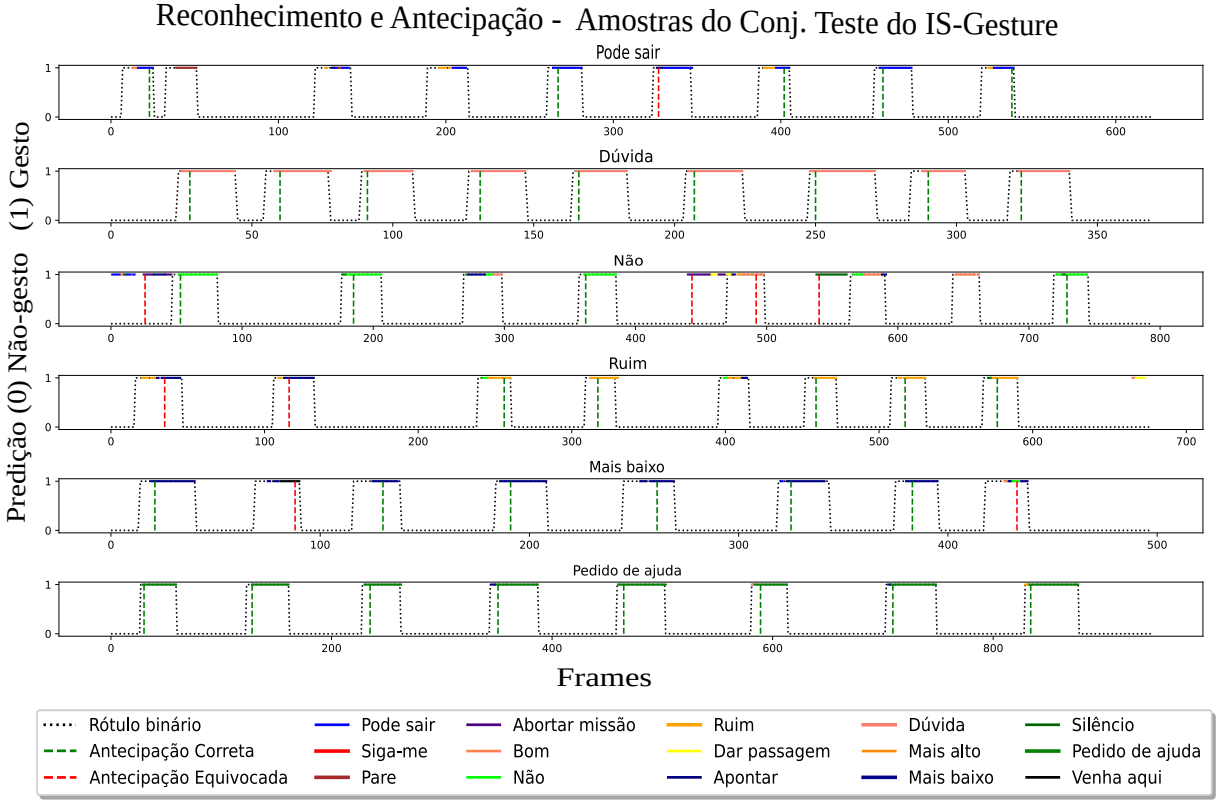
Fonte: Próprio autor.

Considerando-se a operação conjunta de ambos os modelos, a acurácia de reconhecimento das 16 classes (15 *gestos* e 1 *não-gesto*) foi de 83,71%. Isso significa que, no geral, mais de 83% dos $frames$ são classificados corretamente. Comparando esse resultado com os 72% obtido pelo $BSKL$, houve um ganho de 16% com o uso do $BSKL_{hand}$. Quanto ao índice de Jaccard, o valor obtido foi de 0,582. Isso significa que nas predições do modelo de *spotting* pertencentes à classe *gesto*, mais de 58% delas foram classificadas corretamente pelo modelo classificador. Note-se que em relação ao $BSKL$ (0,251) o ganho obtido pelo $BSKL_{hand}$ foi de 131,8%. Isso demonstra uma grande melhora do modelo após o uso da informação contextual.

Considerando-se agora a capacidade de antecipação do modelo, utilizando-se um limiar de incerteza de $u = 1{,}4$, ele foi capaz de antecipar corretamente 63% dos gestos utilizando em média 24% das observações, o que representa um ganho de 50% em relação aos 42% do $BSKL$.

Para um visão mais completa do desempenho do $BSKL_{hand}$, a Figura 62 apresenta uma ilustração da predição feita por ele em seis sequências completas do conjunto de testes

Figura 62 – Reconhecimento e antecipação em seis sequência completas de gestos do conjunto de teste do IS-Gesture. A classe 0 corresponde a um *não-gesto* já a classe 1 corresponde tanto à classe *gesto* predita pelo modelo de *spotting*, quanto a uma das 15 classes de gestos predita pelo classificador. A área correspondente à execução de um gesto que não possui uma das linhas tracejadas verde ou vermelha significa que o gesto ali representado não foi antecipado.



Fonte: Próprio autor.

do IS-Gesture. Veja-se que apenas alguns gestos não foram antecipados, ou a antecipação foi equivocada. Outra observação é que todos os gestos das sequências *dúvida* e *pedido de ajuda* foram antecipados corretamente após a observação de alguns poucos *frames*. Dentre os gestos apresentados, os da sequência *não* são os que o modelo de antecipação mais falhou. Isso se deve principalmente ao erro do modelo de *spotting* que acabou segmentando errado alguns *frames*, o que induziu o modelo de classificação a erro. Aqui é possível ver, mais uma vez, a importância do modelo de segmentação para o reconhecimento e antecipação de gestos em um ambiente real.

5.3.3.3 Tempo de resposta do sistema

Para avaliar a viabilidade do uso do modelo proposto em uma aplicação real, como dito antes, os resultados das predições foram obtidos simulando o espaço inteligente utilizando os vídeos do conjunto de teste como entrada e armazenando os resultados do

Tabela 32 – Tempos médios de resposta do sistema calculados para a predição dos gestos no conjunto de dados de teste do Is-Gesture utilizando o $BSKL_{hand}$ e o $BSKL$. Observe-se que apenas os modelos e o *Openpose* podem ser executados em CPU ou em GPU. O processo de cálculo da extração das mãos e o serviço de comunicação sempre são executados em CPU. Todos os valores estão arredondados para o maior inteiro mais próximo.

Nome do modelo	Dispositivo	Tempo médio (ms)				
		Modelo	*Openpose*	Extração das mãos	Comunicação	Resposta
$BSKL_{hand}$	CPU	126	1263	3	11	1403
	GPU	16	69	-	-	98
$BSKL$	CPU	3	1263	Não utiliza	11	1277
	GPU	1	69	Não utiliza	-	81

Fonte: Próprio autor.

serviço de reconhecimento de gestos (recorde-se o exemplo da Figura 55), além dos tempos de processamento e comunicação dos principais serviços utilizados. Assim foi possível avaliar o tempo médio de resposta do sistema como um todo. Um resumo dos resultados está apresentado na Tabela 32. Veja-se que sendo executado em GPU, o tempo médio de resposta do $BSKL_{hand}$ para cada observação foi de $98ms$. Já em CPU, o seu tempo de resposta chega a ultrapassar os $1.400ms$. Com isso, utilizando GPU é possível ter um reconhecedor (antecipador) de gestos que responda a uma aplicação de interação natural em menos de $100ms$, ou seja, consegue operar a pelo menos a 10 FPS, taxa máxima de operação das aplicações do espaço inteligente em questão. A diferença no tempo de resposta entre o BSKL e o BSKL_{hand} é de $17ms$ quando executado em GPU, ou seja, foi possível obter melhora significativa tanto do reconhecimento quanto na antecipação de gestos com o BSKL_{hand}, a um custo de apenas $17ms$ no tempo de resposta do sistema.

5.4 Comentários gerais

Neste capítulo, fechou-se o escopo do problema abordado pelo conjunto de dados Montalbano. Aqui foi proposto um modelo que pode ser executado em tempo real, e que obteve um dos melhores resultados da literatura para o reconhecimento de gestos em seus vídeos, sendo utilizadas apenas informações extraídas das imagens RGB. Além disso, o modelo proposto tem a capacidade de segmentar, reconhecer e (ou) antecipar os gestos de forma contínua. Com isso, foi possível alcançar resultados competitivos quanto aos reconhecimento *online* e *offline*. Além disso, este trabalho é um dos poucos que aborda a antecipação de gestos para tal conjunto de dados. No entanto, diferentemente dos demais trabalhos, aqui foram apresentados os tempos de resposta da proposta. Isso mostra a sua viabilidade na utilização em um ambiente interacional real.

Aqui foi proposto também um modelo capaz de reconhecer e antecipar gestos de modo *online* utilizando um ambiente interacional multicâmeras. Tal problema pode

ser considerado muito mais complexo que o do Montalbano. Isso se deve ao fato de que o conjunto de dados do IS-Gesture, que possui menos de 1/5 da quantidade de gestos do Montalbano, foi capturado em dois ambientes distintos, por câmeras com diferentes ângulos de visão e utilizando voluntários distintos. Além disso, as imagens das várias câmeras incrementa significativamente a quantidade de dados a ser tratada e dificulta principalmente a extração de informação contextual. Recorde-se que no Montalbano, as mãos são sempre visualizadas, são capturadas com uma câmera de frente para o usuário, e estão majoritariamente em um lado específico da imagem. No entanto, no IS-Gesture, a depender da posição do usuário na cena em um determinado instante de tempo, cada uma das mãos pode ser capturada por câmeras diferentes em locais diferentes. Assim, tanto a caracterização espacial das mãos quanto a captura das dependências temporais entre tais características tornam a tarefa de reconhecimento (antecipação) de gestos, com o IS-Gesture, um problema bem mais complexo de resolver. Mesmo assim, os resultados alcançados podem ser considerados satisfatórios.

Mesmo com os significativos resultados alcançados, ainda existem algumas limitações que necessitam ser abordadas. O método de reconstrução do esqueleto 3D utilizado ainda é muito ruidoso. Mesmo com o usuário parado, é comum a perda de algumas juntas, ainda que não haja oclusão. Isso pode ser uma consequência dos ruídos fornecidos pelos esqueletos 2D extraídos com o *Openpose*. Ruído esse que influencia também a extração das mãos. Note-se que esse problema também prejudicou o modelo *BStar* $iRGB_{hand}$ (Seção 4.5). Sendo que no conjunto de dados Montalbano, onde a câmera está em apenas um ponto de visão e de frente para o usuário, esse ruído prejudicou bem menos que no IS-Gesture, onde existem quatro câmeras com pontos de vista distintos e todas estão com mais de quatro metros de distância do voluntário.

Ainda em se tratando da informação das mãos, assim como discutido na Seção 4.6, modelos treinados especificamente para o reconhecimento das mãos poderiam fornecer dados menos ruidosos, além de, possivelmente, diminuir de forma considerável o tempo de resposta do sistema quando executado em GPU. Outra maneira interessante de passar a informação de contexto seria treinar um modelo não só para detectar as mãos, mas também para classificá-las de acordo com a sua forma, como por exemplo, mão aberta, mão fechada, polegar para cima, polegar para baixo, dentre outras. Essa informação condensada de contexto faria com que o reconhecedor tivesse uma resposta ainda mais rápida e provavelmente alcançaria resultados superiores aos do $BLSK_{hand}$. Recorde-se como a informação condensada de contexto (posição da bola e pontos dos olhos) ajudou na tarefa de antecipação de ação do modelo proposto na Seção 4.4.

Por fim, num espaço inteligente, é comum haver múltiplos usuários interagindo ao mesmo tempo. Dessa maneira, cada um pode realizar gestos diferentes a cada instante de tempo. Nesse cenário, qual a abordagem deveria ser usada: um modelo para cada

usuário, ou apenas um modelo que leva em consideração todos os usuários de uma só vez? Para ambos os casos, é necessário primeiro identificar unicamente cada usuário em todas as observações, o que é um outro problema ainda em aberto na literatura. Soluções para essas duas perguntas devem ser cuidadosamente analisadas, para que o sistema não seja sobrecarregado pela quantidade de modelos em execução simultaneamente, ou pelo processamento demandado por um único modelo que tente resolver todo o problema sozinho.

6 CONCLUSÕES E TRABALHOS FUTUROS

> Under Bayes' theorem, no theory is perfect. Rather, it is a work in progress, always subject to further refinement and testing.
>
> Nate Silver

6.1 Conclusões

As máquinas estão cada dia mais presentes na vida cotidiana das pessoas. No entanto, para que essa participação tenha maior efetividade, é preciso que elas também possuam as capacidades de reconhecimento e antecipação de gestos e ações inerentes ao ser humano.

No contexto de visão computacional, gestos e ações podem ser representados por variações de intensidades em partes específicas de uma sequência de imagens. Assim, como tais aplicações são utilizadas em um ambiente onde o processamento ocorre em tempo real, o dado de entrada de um modelo de aprendizagem de máquina é comumente um *stream* de vídeo, onde as imagens são fornecidas uma a uma por uma fonte de dados específica.

Dentro desse escopo, após várias observações e pesquisas bibliográficas este trabalho levantou três hipóteses:

1. Por meio de uma representação compacta de movimento de um vídeo, um método baseado apenas em imagens RGB pode ser tão efetivo para o reconhecimento de gestos dinâmicos quanto os métodos multimodais são.
2. A escolha empírica da informação de contexto pode ser uma alternativa efetiva para distinguir diferentes ações ou gestos representados por movimentos ambíguos.
3. Para o problema de antecipação, a incerteza sobre a predição é um limiar eficaz e mais confiável do que o valor da probabilidade estimada pelo modelo.

Dessa forma, a fim de adquirir evidências que sustentassem cada uma das hipóteses levantadas, foram apresentadas seis principais propostas, quais sejam:

- Uma técnica de representação de movimento (Star RGB) capaz de melhorar o reconhecimento de gestos dinâmicos, mesmo em situações de baixa variância interclasse.

- Uma variante iterativa da técnica proposta (Star iRGB) que possa ser utilizada por modelos de natureza sequencial.
- Um modelo de antecipação de ações que usa a informação contextual em adição à de movimento, e a incerteza como limiar de tomada de decisão.
- Um modelo de antecipação e reconhecimento de gestos.
- A combinação das propostas anteriores a fim de alcançar resultados satisfatórios no reconhecimento e antecipação de gestos ou ações de forma *online*.
- Um sistema capaz de antecipar e reconhecer gestos de forma *online* e em tempo real, em um ambiente interacional multicâmeras.

Os experimentos com o *Star RGB* foram realizados usando os conjuntos de dados Montalbano, GRIT e IsoGD. Para o conjunto de dados Montalbano, a abordagem proposta alcançou uma acurácia média entre classes de 94,58%. Esse resultado atinge o estado da arte ao considerar apenas informações de cor para esse conjunto de dados. Para o conjunto de dados GRIT, alcançou-se mais de 98% de acurácia, *recall*, precisão e *F1-score*, superando a abordagem dos autores em mais de 6%. Em relação ao conjunto de dados IsoGD, a proposta alcançou 52,18% de acurácia média. Considerando a complexidade desse último conjunto de dados (oito categorias de gestos diferentes) e a quantidade de classes (249), considera-se a proposta competitiva com as anteriores, pois foram empregadas apenas informações de cor para reconhecer gestos em vez de todas as funções multimodais dos dados disponíveis, geralmente usadas por outros métodos.

Já os experimentos com o *Star iRGB* foram realizados utilizando apenas o Montalbano, alcançando-se, assim, uma acurácia média de 94,96%. Comparando os resultados obtidos com os apresentados em trabalhos anteriores, têm-se aqui, para o conjunto de dados Montalbano, os melhores resultados utilizando apenas imagens RGB, e um dos melhores resultados gerais, ficando atrás, por pouco, de alguns métodos multimodais que utilizam todas as informações de dados fornecidas. Dessa forma, estes resultados sustentam a primeira hipótese levantada.

Os experimentos com a proposta de antecipação de ações foram executados sobre o conjunto de dados Acticipate. Nessa proposta, além do movimento realizado pela pessoa no vídeo, o *gaze* e a posição do objeto manipulado foram utilizados como informações de contexto. Além disso, a incerteza, ao invés da probabilidade, foi considerada como o limiar para a predição de ações. Como resultado, obteve-se uma acurácia média de 98,75% na tarefa de antecipação, usando apenas uma média de 25% das observações. Além disso, considerando-se que um bom modelo de antecipação também deve ter um bom desempenho na tarefa de reconhecimento de ações, alcançou-se uma acurácia média de 100% no reconhecimento de ações no conjunto de dados Acticipate, quando toda a sequência de observações foi usada. Em complemento, utilizou-se dos conhecimento adquiridos nessa abordagem para propor um modelo capaz de antecipar gestos. Os experimentos também

foram executados sobre Montalbano, e alcançou uma acurácia média de reconhecimento no último *frame* de cada sequência de 97,46% utilizando apenas informação de cor. Quanto à tarefa de antecipação, ele foi capaz de antecipar corretamente 89% dos gestos utilizando apenas 22% das observações. Comparando esses com os principais resultados da literatura, os resultados aqui obtidos, ou foram maiores ou muito próximos. Isso mostra a qualidade das propostas apresentadas, além de fornecer evidências que sustentam as duas últimas hipóteses levantadas.

A fim de aplicar o conhecimento adquirido no reconhecimento e antecipação de gestos de modo *online*, as duas últimas propostas foram apresentadas. A primeira delas foi direcionada aos gestos do Montalbano. Sendo assim, após os experimentos, ela obteve uma acurácia média na classificação binária (*spotting*) e multiclasse (*spotting* + classificador) *frame* a *frame* de 93,12% e 88,57%, respectivamente. Obteve ainda uma acurácia média de reconhecimento dos gestos no último *frame* da sequência de 89,44%, e um índice de *Jaccard* de 0,807. Em relação à sua capacidade de antecipação *online* de gestos, a proposta alcançou uma acurácia de 77,16%. Com essa proposta, este trabalho apresentou resultados para todas as modalidades do desafio *Chalearn 2014: Looking At people (Track 3: Gesture Recognition)* (reconhecimento de gestos isolados, segmentados, e reconhecimento de gestos *online*), além de ser um dos poucos trabalhos a abordar a antecipação de gestos para tal conjunto de dados. Dessa forma, em resumo, ao comparar com os principais trabalhos, todos os resultados apresentados aqui para o Montalbano estão entre os melhores, sendo que são os melhores dentre aqueles que utilizaram apenas informação de cor. Dessa forma, têm-se ainda mais evidências que sustentam as hipóteses levantadas.

Partindo para a última proposta, ela foi pensada com o intuito de apresentar uma solução para um problema ainda mais complexo e desafiador que os tratados nas cinco primeiras propostas: o de reconhecer e antecipar gestos em um ambiente interacional multicâmeras. Para os experimentos, foi utilizado o conjunto de dados IS-Gesture, levantado pelos integrantes dessa pesquisa para tal fim. Dessa maneira, alcançou-se uma acurácia média na classificação binária e multiclasse *frame* a *frame* de 92,68% e 83,71%, respectivamente. Obteve-se ainda um índice de *Jaccard* de 0,582. Isso significa que para as predições do modelo de *spotting* pertencentes à classe *gesto*, mais de 58% delas foram classificadas corretamente pelo modelo classificador. Em relação à sua capacidade de antecipação, o modelo proposto foi capaz de antecipar corretamente 63% dos gestos utilizando em média 24% das observações. Diante da complexidade do problema, consideram-se os resultados bem promissores e que demonstram a viabilidade do uso da proposta em uma aplicação de interação natural, em um espaço inteligente multicâmeras.

Por fim, a maioria das propostas mostraram-se passíveis de serem implementadas, para aplicações em tempo real, com uma taxa de fornecimento de imagens de até 10 FPS. Essa também é a taxa máxima de funcionamento das principais aplicações dos espaços

inteligentes utilizados nos experimentos da última proposta (Capítulo 5, Seção 5.3)

6.2 Trabalhos futuros

Apesar de considerados satisfatórios, alguns trabalhos ainda precisam ser feitos a fim de alcançar resultados ainda mais significativos. As próximas subseções abordarão algumas delas com detalhes.

6.2.1 Representação de movimento

Após a análise dos vídeos do Montalbano, percebeu-se que os erros mais recorrentes do modelo de *spotting* proposto no Capítulo 5, Seção 5.2, eram justamente em gestos consecutivos realizados sem a etapa de descanso dos braços. Isso pode ser um indício de que o *Star iRGB* não foi capaz de capturar a informação de movimento necessária para a segmentação dos dois gestos. Dessa forma, apesar do *Star RGB* e *iRGB* terem se mostrado eficazes como representações de movimentos para o reconhecimento e a antecipação de gestos dinâmicos, ainda é necessário um estudo mais aprofundado sobre as suas limitações quanto à capacidade de representação de movimentos rápidos e lentos, bem como dos curtos e dos longos.

Um trabalho interessante seria o estudo sobre a importância de cada canal do *Star RGB*, e sobre a possibilidade de ter canais com diferentes quantidades de *frames*; já que na proposta aqui apresentada, os três possuem, a princípio, a mesma quantidade de *frames*, com a possibilidade do canal central possuir apenas dois a mais que os outros. Um estudo como esse pode trazer evidências que servirão de hipóteses para a quantidade ideal de informação que cada canal deve conter, a fim de maximizar os resultados do modelo quando da utilização do *Star RGB* ou do *Star iRGB* como informação de movimento. Por exemplo, para um problema em específico, ao detectar que o canal central do *Star RGB* deve ter o dobro de *frames* que os outros dois, caso o modelo não seja sequencial, pode-se simplesmente dobrar o número de *frames* contido no canal central do *Star RGB*. Por outro lado, uma vez que o *Star iRGB* permite uma representação de movimento com N canais, caso o modelo seja iterativo, o *Star iRGB* poderia ser representado por quatro canais com a mesma quantidade de *frames* cada. Atente-se para o fato de que, para uma imagem com um número de canais diferente de três, faz-se necessário a utilização de métodos extras aplicados à imagem de entrada para que a transferência de aprendizado das principais redes CNNs seja utilizada.

6.2.2 O problema dos múltiplo usuários

Como comentado no Capítulo 5, em um ambiente interacional real, é comum se ter mais de uma pessoa realizando gestos e ações ao mesmo tempo. Dessa forma, considerando que o espaço será o protagonista no reconhecimento de tais gestos e ações, como ele deverá agir? A cada instante de tempo determinadas ações (gestos) poderão estar em diferentes fases, a depender das pessoas que as estão realizando. Sendo assim, é necessário estudos com propostas para tal problema. Só depois disso, poder-se-á tornar os reconhecedores e antecipadores de ações e gestos mais efetivos e propensos a serem utilizados nos mais diversos ambientes.

6.2.3 Relação entre pessoas e objetos

Como visto no Capítulo 4, a relação entre a pessoa e o objeto foi decisiva para uma antecipação efetiva de cada ação. No entanto, no problema abordado, as ações dependiam apenas de um único objeto, sendo que, em cada instante de tempo, ele estava relacionado a apenas uma pessoa. No caso de ações mais complexas, que envolvam mais objetos e mais pessoas, como é possível obter as informações das relações objeto-pessoa, pessoa-pessoa e objeto-objeto?

Como visto na subseção anterior, tratar o problema multiusuário já não é fácil. Dessa forma, quando ele envolve multiusuários e multiobjetos, ele fica ainda mais complexo. Sendo assim, estudos nessa direção são de primordial importância para que os robôs ou os ambiente interacionais possa antecipar e reconhecer ações e gestos dos indivíduos presentes na cena.

Um exemplo desse problema é o seguinte: imagine-se um local com várias pessoas e que uma aplicação é a responsável por diminuir o som ambiente sempre que alguém utiliza o telefone para atender ou fazer uma ligação. Como saber qual delas está ao telefone? Em um ambiente desse tipo, podem haver vários telefones sobre as mesas, ou mesmo sendo movimentados nas mãos das pessoas. Nesse sentido, o modelo deveria ser capaz de identificar os telefones, identificar as pessoas e identificar corretamente se houve um movimento de levar o aparelho ao ouvido ou à boca. Note-se que não é um problema simples, pois são diversas as variáveis envolvidas. Nesse tipo de aplicação, falsos positivos não são desejáveis, pois pode incomodar todas as pessoas presentes, e falsos negativos podem dificultar o entediamento da ligação por parte do usuário que está ao telefone.

6.2.4 As forma das mãos como fonte de informação contextual

Como discutido anteriormente, as mãos são uma rica fonte de informação contextual para o problema de reconhecimento e antecipação de gestos. Porém, usar o *Openpose* não é a solução mais eficiente para extraí-las das imagens. Ele introduz muito ruído nas juntas detectadas, além de necessitar de um poder de processamento que impossibilita algumas aplicações interacionais de serem executadas em tempo real. Dessa maneira, o uso de pequenas CNNs treinadas, apenas para esse fim, poderão trazer menos ruídos na detecção e serem executadas muito mais rapidamente que o *Openpose*.

Considerando as características dos ambientes multicâmeras, algo ainda melhor poderia ser feito. No problema de antecipação de ações do Capítulo 4, a informação de contexto referente ao objeto não foi a sua imagem recortada, mas sim o seu ponto central de localização na imagem. Essa informação condensada permitiu que o modelo aprendesse sem a necessidade de uma quantidade maior de amostras de treinamento. Algo similar pode ser feito para o reconhecimento (antecipação) de gestos: ao invés de passar os recortes das mãos para uma CNN, pode-se classificar a forma da mão em cada *frame* e fornecer essa informação condensada para o modelo. A princípio, indica-se utilizar a mesma estrutura de *embedding* proposta para problema de antecipação de ação, porém outros mecanismos do tipo também podem ser proposto. Quanto à sua quantidade, as formas das mão podem ser determinadas por uma análise da base de dados e podem ser automaticamente obtidas por uma modelo de *cluster*. No entanto, da mesma forma que na indicação do *embedding*, outras abordagens também podem ser bem efetivas.

6.2.5 Reconstrução de esqueleto 3D a partir de várias imagens

Mesmo que o reconstrutor de esqueleto 3D proposto em Queiroz et al. (2018) tenha possibilitado o reconhecimento e a antecipação de gestos em um espaço inteligente multicâmeras, ele ainda é muito ruidoso, e por isso, pode ter prejudicado significativamente os resultados obtidos. Nesse sentido, deve-se atentar também para técnicas cada vez mais robustas de determinação de esqueletos 3D em espaços inteligentes multicâmeras. Os mesmos devem ser rápidos e assertivos quanto às juntas dos esqueletos reconstruídos. A perda de juntas, ou mesmo a confusão entre juntas de partes distintas, pode fornecer uma representação ruidosa que prejudica o aprendizado do modelo.

6.2.6 A importância do modelo de *spotting*

Como foi visto no último capítulo, o modelo de *spotting* é muito importante para garantir a qualidade do modelo de classificação quando executado em uma aplicação de tempo real e *online* (dados fornecidos via *stream*). Como visto, não adianta ter um

modelo de classificação que alcança uma acurácia próxima a 100% se o modelo de *spotting* consegue uma taxa bem abaixo dessa. É importante lembrar que um falso positivo do modelo de *spotting* (segmentar uma sequência de *observações* que não contém um gesto ou uma ação) induzirá o modelo de classificação a erro. Dessa forma, propostas de modelos segmentadores, bem mais elaboradas que as apresentadas aqui neste trabalho, podem trazer resultados ainda melhores para o reconhecimento e a antecipação *online* de gestos, tanto no Montalbano quanto no IS-Gesture.

6.2.7 Tomada de decisão na antecipação

De acordo com o Princípio da Utilidade Moral de Bernoulli (mais detalhes em Cusinato e Júnior (2005)), as decisões não devem ser tomadas de acordo com o seu valor monetário (função de custo) mas sim pela sua utilidade (função de utilidade). Dessa forma, apesar de uma das principais contribuições deste trabalho ser a utilização de um limiar sobre a incerteza epistêmica estimada pelo modelo para poder tomar a decisão de quando o gesto (ação) deve ser antecipado, em uma situação real, uma função de utilidade pode ser ainda mais eficaz. Sendo assim, um estudo pode ser feito no sentido de determinar qual a utilidade (riscos, perigos, prejuízos) de cada classe antecipada corretamente e erroneamente para o sistema e seus usuários, a fim de determinar os pesos que possam ponderar cada antecipação realizada, de maneira a melhorar a experiência do usuário. Por exemplo, em um vocabulário de gestos, antecipar erroneamente um gesto que corresponde a uma ação complexa de um robô, como carregar uma carga por uma longa distância, é bem pior que antecipar erroneamente um gesto de dúvida ou pedido de atenção para um robô assistente. Assim sendo, no momento da antecipação de alguns gestos, mesmo utilizando a incerteza como limiar, o modelo poderia esperar por mais observações para classes de gestos ou ações com maior utilidade, a fim de cometer erros menos graves.

REFERÊNCIAS

AGETHEN, S.; LEE, H.-C.; HSU, W. H. Anticipation of human actions with pose-based fine-grained representations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. [S.l.: s.n.], 2019. p. 0–0. Citado na página 118.

ALIAKBARIAN, M. S. et al. Deep action-and context-aware sequence learning for activity recognition and anticipation. arXiv preprint arXiv:1611.05520, 2016. Citado na página 119.

ALIAKBARIAN, M. S. et al. Encouraging lstms to anticipate actions very early. In: Proceedings of the IEEE International Conference on Computer Vision. [S.l.: s.n.], 2017. p. 280–289. Citado na página 118.

ALIAKBARIAN, M. S. et al. Encouraging lstms to anticipate actions very early. In: Proceedings of the IEEE International Conference on Computer Vision. [S.l.: s.n.], 2017. p. 280–289. Citado na página 119.

ALMONFREY, D. et al. A flexible human detection service suitable for Intelligent Spaces based on a multi-camera network. International Journal of Distributed Sensor Networks, SAGE PublicationsSage UK: London, England, v. 14, n. 3, p. 155014771876355, mar 2018. ISSN 1550-1477. Citado na página 184.

BALTRUSAITIS, T. et al. Openface 2.0: Facial behavior analysis toolkit. In: IEEE. 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018). [S.l.], 2018. p. 59–66. Citado na página 132.

BARADEL, F. et al. Object level visual reasoning in videos. In: Proceedings of the European Conference on Computer Vision (ECCV). [S.l.: s.n.], 2018. p. 105–121. Citado na página 40.

BARROS, P. et al. A multichannel convolutional neural network for hand posture recognition. In: SPRINGER. International Conference on Artificial Neural Networks. [S.l.], 2014. p. 403–410. Citado 2 vezes nas páginas 220 e 222.

BARROS, P. et al. Real-time gesture recognition using a humanoid robot with a deep neural architecture. In: IEEE. Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on. [S.l.], 2014. p. 646–651. Citado 8 vezes nas páginas x, 38, 40, 72, 76, 78, 80 e 81.

BILEN, H. et al. Dynamic image networks for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2016. p. 3034–3042. Citado na página 118.

BILEN, H. et al. Action recognition with dynamic image networks. IEEE transactions on pattern analysis and machine intelligence, IEEE, v. 40, n. 12, p. 2799–2813, 2017. Citado na página 40.

BISHOP, C. M. Pattern recognition and machine learning. [S.l.]: springer, 2006. Citado na página 50.

BLEI, D. M.; KUCUKELBIR, A.; MCAULIFFE, J. D. Variational inference: A review for statisticians. Journal of the American Statistical Association, Taylor & Francis, v. 112, n. 518, p. 859–877, 2017. Citado na página 68.

BLOOM, V.; ARGYRIOU, V.; MAKRIS, D. Linear latent low dimensional space for online early action recognition and prediction. Pattern Recognition, Elsevier, v. 72, p. 532–547, 2017. Citado na página 120.

BLUNDELL, C. et al. Weight uncertainty in neural network. In: International Conference on Machine Learning. [S.l.: s.n.], 2015. p. 1613–1622. Citado 2 vezes nas páginas 69 e 163.

BOBICK, A. F.; DAVIS, J. W. The recognition of human movement using temporal templates. IEEE Transactions on Pattern Analysis & Machine Intelligence, IEEE, n. 3, p. 257–267, 2001. Citado na página 76.

BOCHKOVSKIY, A.; WANG, C.-Y.; LIAO, H.-Y. M. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934, 2020. Citado na página 164.

CAO, C.; ZHANG, Y.; LU, H. Multi-modal learning for gesture recognition. In: IEEE. 2015 IEEE International Conference on Multimedia and Expo (ICME). [S.l.], 2015. p. 1–6. Citado 4 vezes nas páginas 76, 94, 96 e 165.

CAO, C.; ZHANG, Y.; LU, H. Multi-modal learning for gesture recognition. In: IEEE. 2015 IEEE International Conference on Multimedia and Expo (ICME). [S.l.], 2015. p. 1–6. Citado na página 168.

Cao, Z. et al. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. IEEE Transactions on Pattern Analysis and Machine Intelligence, p. 1–1, 2019. Citado na página 131.

CARMO, A. P. et al. Programmable intelligent spaces for Industry 4.0: Indoor visual localization driving attocell networks. Transactions on Emerging Telecommunications Technologies, John Wiley & Sons, Ltd, p. e3610, apr 2019. ISSN 2161-3915. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3610>. Citado na página 184.

CARREIRA, J.; ZISSERMAN, A. Quo vadis, action recognition? a new model and the kinetics dataset. In: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2017. p. 6299–6308. Citado na página 40.

CARREIRA, J.; ZISSERMAN, A. Quo vadis, action recognition? a new model and the kinetics dataset. In: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2017. p. 6299–6308. Citado 2 vezes nas páginas 72 e 130.

CHAI, X. et al. Two streams recurrent neural networks for large-scale continuous gesture recognition. In: IEEE. 2016 23rd International Conference on Pattern Recognition (ICPR). [S.l.], 2016. p. 31–36. Citado 3 vezes nas páginas 169, 171 e 172.

CHANDRASHEKAR, G.; SAHIN, F. A survey on feature selection methods. Computers & Electrical Engineering, Elsevier, v. 40, n. 1, p. 16–28, 2014. Citado na página 55.

CHANG, J. Y. Nonparametric gesture labeling from multi-modal data. In: SPRINGER. European Conference on Computer Vision. [S.l.], 2014. p. 503–517. Citado na página 178.

CHATTOPADHAY, A. et al. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In: IEEE. Applications of Computer Vision (WACV), 2018 IEEE Winter Conference on. [S.l.], 2018. p. 839–847. Citado na página 110.

CHEN, G. et al. Multi-modality gesture detection and recognition with un-supervision, randomization and discrimination. In: SPRINGER. European Conference on Computer Vision. [S.l.], 2014. p. 608–622. Citado na página 178.

CHEN, X.; KOSKELA, M. Using appearance-based hand features for dynamic RGB-D gesture recognition. Proceedings - International Conference on Pattern Recognition, p. 411–416, 2014. ISSN 10514651. Citado na página 75.

CHOI, E. et al. Doctor ai: Predicting clinical events via recurrent neural networks. In: Machine Learning for Healthcare Conference. [S.l.: s.n.], 2016. p. 301–318. Citado na página 66.

CHOUTAS, V. et al. Potion: Pose motion representation for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2018. p. 7024–7033. Citado na página 40.

CHUNG, J. et al. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555, 2014. Citado na página 66.

CUSINATO, R. T.; JÚNIOR, S. P. teoria da decisão sob incerteza e a hipótese da utilidade esperada. Santa Cruz do Sul-RS: Estudos do CEPE, v. 22, p. 7–38, 2005. Citado na página 204.

DEVLIN, J. et al. Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). [S.l.: s.n.], 2019. p. 4171–4186. Citado 2 vezes nas páginas 40 e 65.

DEY, A. K.; ABOWD, G. D. Towards a Better Understanding of Context and Context-Awareness. Computing Systems, v. 40, n. 3, p. 304–307, 1999. ISSN 00219266. Citado na página 116.

DOLLÁR, P. et al. Behavior recognition via sparse spatio-temporal features. In: IEEE. 2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance. [S.l.], 2005. p. 65–72. Citado na página 223.

DUAN, J. et al. A unified framework for multi-modal isolated gesture recognition. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), ACM New York, NY, USA, v. 14, n. 1s, p. 1–16, 2018. Citado na página 97.

DUARTE, N. F. et al. Action anticipation: reading the intentions of humans and robots. IEEE Robotics and Automation Letters, IEEE, v. 3, n. 4, p. 4132–4139, 2018. Citado 5 vezes nas páginas 115, 117, 119, 125 e 131.

EFTHIMIOU, E. et al. The MOBOT rollator human-robot interaction model and user evaluation process. 2016 IEEE Symposium Series on Computational Intelligence (SSCI), p. 1–8, 2016. Citado 4 vezes nas páginas 74, 76, 96 e 165.

ESCALERA, S.; ATHITSOS, V.; GUYON, I. Challenges in multi-modal gesture recognition. In: Gesture Recognition. [S.l.]: Springer, 2017. p. 1–60. Citado na página 37.

ESCALERA, S. et al. Chalearn looking at people challenge 2014: Dataset and results. In: SPRINGER. Workshop at the European Conference on Computer Vision. [S.l.], 2014. p. 459–473. Citado 5 vezes nas páginas xi, 74, 86, 87 e 178.

ESCALERA, S. et al. Multi-modal gesture recognition challenge 2013: Dataset and results. In: ACM. Proceedings of the 15th ACM on International conference on multimodal interaction. [S.l.], 2013. p. 445–452. Citado na página 74.

ESCOBEDO, C. E.; CAMARA, C. G. A robust gesture recognition using hand local data and skeleton trajectory. In: 2015 IEEE International Conference on Image Processing (ICIP). [S.l.]: IEEE, 2015. p. 1240–1244. ISBN 978-1-4799-8339-1. Citado 5 vezes nas páginas 75, 76, 94, 96 e 165.

ESCOBEDO-CARDENAS, E.; CAMARA-CHAVEZ, G. A robust gesture recognition using hand local data and skeleton trajectory. In: IEEE. 2015 IEEE International Conference on Image Processing (ICIP). [S.l.], 2015. p. 1240–1244. Citado na página 167.

FEICHTENHOFER, C.; PINZ, A.; ZISSERMAN, A. Convolutional two-stream network fusion for video action recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2016. p. 1933–1941. Citado na página 72.

FELSEN, P.; AGRAWAL, P.; MALIK, J. What will happen next? forecasting player moves in sports videos. In: Proceedings of the IEEE International Conference on Computer Vision. [S.l.: s.n.], 2017. p. 3342–3351. Citado na página 120.

FERNANDO, B. et al. Modeling video evolution for action recognition. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, v. 07-12-June, p. 5378–5387, 2015. ISSN 10636919. Citado 4 vezes nas páginas 75, 76, 96 e 165.

FUKUSHIMA, K.; MIYAKE, S. Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In: Competition and cooperation in neural nets. [S.l.]: Springer, 1982. p. 267–285. Citado 2 vezes nas páginas 55 e 57.

GAL, Y. Uncertainty in deep learning. Tese (Doutorado) — PhD thesis, University of Cambridge, 2016. Citado 2 vezes nas páginas 70 e 71.

GAL, Y.; GHAHRAMANI, Z. Dropout as a bayesian approximation. In: 33rd International Conference on Machine Learning, ICML 2016. [S.l.: s.n.], 2016. v. 3, p. 1661–1680. Citado 2 vezes nas páginas 69 e 163.

GAL, Y.; GHAHRAMANI, Z. A theoretically grounded application of dropout in recurrent neural networks. In: Advances in neural information processing systems. [S.l.: s.n.], 2016. p. 1019–1027. Citado 2 vezes nas páginas 135 e 163.

GHANBARI, M.; OHLER, U. Deep neural networks for interpreting rna-binding protein target preferences. Genome Research, Cold Spring Harbor Lab, v. 30, n. 2, p. 214–226, 2020. Citado na página 66.

GIRSHICK, R. Fast r-cnn. In: International Conference on Computer Vision (ICCV). [S.l.: s.n.], 2015. Citado na página 169.

GITE, S.; AGRAWAL, H. Early prediction of driver's action using deep neural networks. International Journal of Information Retrieval Research (IJIRR), IGI Global, v. 9, n. 2, p. 11–27, 2019. Citado na página 119.

GITE, S.; AGRAWAL, H.; KOTECHA, K. Early anticipation of driver's maneuver in semiautonomous vehicles using deep learning. Progress in Artificial Intelligence, Springer, v. 8, n. 3, p. 293–305, 2019. Citado na página 119.

GODØY, R. I.; LEMAN, M. Musical gestures: Sound, movement, and meaning. [S.l.]: Routledge, 2010. Citado na página 39.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. Deep learning. [S.l.]: MIT press, 2016. Citado 6 vezes nas páginas 49, 50, 54, 56, 57 e 73.

GRAVES, A. Practical variational inference for neural networks. In: Advances in neural information processing systems. [S.l.: s.n.], 2011. p. 2348–2356. Citado 2 vezes nas páginas 68 e 163.

GRAVES, A. et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the 23rd international conference on Machine learning. [S.l.: s.n.], 2006. p. 369–376. Citado na página 169.

GUO, Y. et al. Deep learning for visual understanding: A review. Neurocomputing, Elsevier, v. 187, p. 27–48, 2016. Citado na página 83.

GUPTA, V. et al. Progression modelling for online and early gesture detection. In: IEEE. 2019 International Conference on 3D Vision (3DV). [S.l.], 2019. p. 289–297. Citado 6 vezes nas páginas 151, 160, 164, 165, 170 e 178.

GUYON, I. et al. Feature extraction: foundations and applications. [S.l.]: Springer, 2008. v. 207. Citado na página 55.

HAFNER, D. et al. Reliable uncertainty estimates in deep neural networks using noise contrastive priors. arXiv preprint arXiv:1807.09289, 2018. Citado na página 71.

HARA, K.; KATAOKA, H.; SATOH, Y. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2018. p. 6546–6555. Citado na página 72.

HARSHMAN, R. A.; LUNDY, M. E. Parafac: Parallel factor analysis. Computational Statistics & Data Analysis, Elsevier, v. 18, n. 1, p. 39–72, 1994. Citado na página 223.

HAYKIN, S. Redes neurais: princípios e prática. [S.l.]: Bookman Editora, 2007. Citado 2 vezes nas páginas 52 e 58.

HE, K. et al. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2016. p. 770–778. Citado na página 40.

HE, K. et al. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2016. p. 770–778. Citado 2 vezes nas páginas 56 e 83.

HINTON, G. E.; OSINDERO, S.; TEH, Y.-W. A fast learning algorithm for deep belief nets. Neural computation, MIT Press, v. 18, n. 7, p. 1527–1554, 2006. Citado na página 54.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. Neural computation, MIT Press, v. 9, n. 8, p. 1735–1780, 1997. Citado na página 65.

HORNIK, K. et al. Multilayer feedforward networks are universal approximators. Neural networks, v. 2, n. 5, p. 359–366, 1989. Citado na página 53.

HOSSEINI, B.; MONTAGNE, R.; HAMMER, B. Deep-aligned convolutional neural network for skeleton-based action recognition and segmentation. In: IEEE. 2019 IEEE International Conference on Data Mining (ICDM). [S.l.], 2019. p. 1096–1101. Citado na página 178.

HU, J.-F. et al. Early action prediction by soft regression. IEEE transactions on pattern analysis and machine intelligence, IEEE, 2018. Citado 2 vezes nas páginas 118 e 120.

HUBEL, D. H.; WIESEL, T. N. Receptive fields of single neurones in the cat's striate cortex. The Journal of physiology, Wiley-Blackwell, v. 148, n. 3, p. 574, 1959. Citado na página 58.

HUBEL, D. H.; WIESEL, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. The Journal of physiology, Wiley-Blackwell, v. 160, n. 1, p. 106, 1962. Citado na página 58.

HUBEL, D. H.; WIESEL, T. N. Receptive fields and functional architecture in two nonstriate visual areas (18 and 19) of the cat. Journal of neurophysiology, v. 28, n. 2, p. 229–289, 1965. Citado na página 58.

HUBEL, D. H.; WIESEL, T. 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. Citado na página 58.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167, 2015. Citado na página 84.

JAIN, A. et al. Car that knows before you do: Anticipating maneuvers via learning temporal driving models. In: Proceedings of the IEEE International Conference on Computer Vision. [S.l.: s.n.], 2015. p. 3182–3190. Citado 3 vezes nas páginas 119, 120 e 163.

JAIN, A. et al. Recurrent neural networks for driver activity anticipation via sensory-fusion architecture. In: IEEE. 2016 IEEE International Conference on Robotics and Automation (ICRA). [S.l.], 2016. p. 3118–3125. Citado 2 vezes nas páginas 119 e 120.

JAMES, G. et al. An introduction to statistical learning. [S.l.]: Springer, 2013. v. 112. Citado 2 vezes nas páginas 50 e 51.

JI, S. et al. 3d convolutional neural networks for human action recognition. IEEE transactions on pattern analysis and machine intelligence, IEEE, v. 35, n. 1, p. 221–231, 2013. Citado na página 72.

JI, Y. et al. One-shot learning based pattern transition map for action early recognition. Signal Processing, Elsevier, v. 143, p. 364–370, 2018. Citado na página 120.

JOSHI, A. et al. Comparing random forest approaches to segmenting and classifying gestures. Image and Vision Computing, Elsevier B.V., v. 58, p. 86–95, 2017. ISSN 02628856. Citado 3 vezes nas páginas 74, 75 e 94.

KARPATHY, A. et al. Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2014. p. 1725–1732. Citado na página 40.

KASSNER, M.; PATERA, W.; BULLING, A. Pupil: an open source platform for pervasive eye tracking and mobile gaze-based interaction. In: ACM. Proceedings of the 2014 ACM international joint conference on pervasive and ubiquitous computing: Adjunct publication. [S.l.], 2014. p. 1151–1160. Citado na página 125.

KENDALL, A.; GAL, Y. What uncertainties do we need in bayesian deep learning for computer vision? In: Advances in neural information processing systems. [S.l.: s.n.], 2017. p. 5574–5584. Citado na página 71.

KINGMA, D. P.; SALIMANS, T.; WELLING, M. Variational dropout and the local reparameterization trick. In: Advances in Neural Information Processing Systems. [S.l.: s.n.], 2015. p. 2575–2583. Citado 2 vezes nas páginas 69 e 163.

KINGMA, D. P.; WELLING, M. Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114, 2013. Citado 2 vezes nas páginas 68 e 69.

KOCH, G.; ZEMEL, R.; SALAKHUTDINOV, R. Siamese neural networks for one-shot image recognition. In: LILLE. ICML deep learning workshop. [S.l.], 2015. v. 2. Citado na página 182.

KOLMOGOROV, A. N. On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition. In: RUSSIAN ACADEMY OF SCIENCES. Doklady Akademii Nauk. [S.l.], 1957. v. 114, n. 5, p. 953–956. Citado na página 53.

KONG, Y.; FU, Y. Human action recognition and prediction: A survey. arXiv preprint arXiv:1806.11230, 2018. Citado na página 115.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. [S.l.: s.n.], 2012. p. 1097–1105. Citado 2 vezes nas páginas 40 e 222.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. [S.l.: s.n.], 2012. p. 1097–1105. Citado na página 56.

KUMAR, S. K. On weight initialization in deep neural networks. preprint, 2017. Citado na página 58.

KWON, H. et al. First person action recognition via two-stream convnet with long-term fusion pooling. Pattern Recognition Letters, Elsevier, v. 112, p. 161–167, 2018. Citado 2 vezes nas páginas 40 e 130.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. nature, Nature Publishing Group, v. 521, n. 7553, p. 436, 2015. Citado na página 57.

LECUN, Y.; BENGIO, Y. et al. Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks, v. 3361, n. 10, p. 1995, 1995. Citado 2 vezes nas páginas 40 e 55.

LECUN, Y.; BENGIO, Y. et al. Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks, v. 3361, n. 10, p. 1995, 1995. Citado na página 57.

LECUN, Y. et al. Backpropagation applied to handwritten zip code recognition. Neural computation, MIT Press, v. 1, n. 4, p. 541–551, 1989. Citado na página 57.

LECUN, Y. et al. Generalization and network design strategies. Connectionism in perspective, Elsevier Zurich, Switzerland, v. 19, p. 143–155, 1989. Citado 2 vezes nas páginas 55 e 57.

LEE, J.-H.; ANDO, N.; HASHIMOTO, H. Intelligent space for human and mobile robot. In: IEEE. Advanced Intelligent Mechatronics, 1999. Proceedings. 1999 IEEE/ASME International Conference on. [S.l.], 1999. p. 784–784. Citado na página 166.

LI, C. et al. Joint distance maps based action recognition with convolutional neural networks. IEEE Signal Processing Letters, IEEE, v. 24, n. 5, p. 624–628, 2017. Citado 2 vezes nas páginas 74 e 75.

LIN, C. et al. Large-scale isolated gesture recognition using a refined fused model based on masked res-c3d network and skeleton lstm. In: IEEE. 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018). [S.l.], 2018. p. 52–58. Citado na página 97.

LIN, T.-Y. et al. Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. [S.l.: s.n.], 2017. p. 2980–2988. Citado na página 164.

LIU, B. et al. Spatiotemporal relationship reasoning for pedestrian intent prediction. IEEE Robotics and Automation Letters, IEEE, v. 5, n. 2, p. 3485–3492, 2020. Citado na página 119.

LIU, J. et al. Skeleton-based online action prediction using scale selection network. IEEE transactions on pattern analysis and machine intelligence, IEEE, 2019. Citado na página 119.

LIU, W. et al. Ssd: Single shot multibox detector. In: SPRINGER. European conference on computer vision. [S.l.], 2016. p. 21–37. Citado na página 164.

LIU, X. et al. Hidden states exploration for 3d skeleton-based gesture recognition. In: IEEE. 2019 IEEE Winter Conference on Applications of Computer Vision (WACV). [S.l.], 2019. p. 1846–1855. Citado 5 vezes nas páginas 75, 76, 80, 96 e 167.

LIU, X. et al. 3d skeletal gesture recognition via hidden states exploration. IEEE Transactions on Image Processing, IEEE, v. 29, p. 4583–4597, 2020. Citado na página 165.

LIU, X.; ZHAO, G. 3d skeletal gesture recognition via sparse coding of time-warping invariant riemannian trajectories. In: SPRINGER. International Conference on Multimedia Modeling. [S.l.], 2019. p. 678–690. Citado na página 75.

MCNEILL, D. Hand and mind: What gestures reveal about thought. [S.l.]: University of Chicago press, 1992. Citado 3 vezes nas páginas 35, 36 e 80.

MITCHELL, T. M. et al. Machine learning. 1997. Burr Ridge, IL: McGraw Hill, v. 45, n. 37, p. 870–877, 1997. Citado na página 49.

MITRA, S.; ACHARYA, T. Gesture recognition: A survey. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), IEEE, v. 37, n. 3, p. 311–324, 2007. Citado na página 37.

MOLCHANOV, P. et al. Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. In: Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2016. p. 4207–4215. Citado 7 vezes nas páginas 151, 160, 164, 165, 169, 170 e 178.

MONNIER, C.; GERMAN, S.; OST, A. A multi-scale boosted detector for efficient and robust gesture recognition. In: SPRINGER. European Conference on Computer Vision. [S.l.], 2014. p. 491–502. Citado na página 178.

MONTALVAO, J.; CANUTO, J.; CARVALHO, E. Convolutional structures and marginal statistics–a study based on k-nearest neighbours. Journal of Communication and Information Systems, v. 33, n. 1, 2018. Citado na página 141.

MURPHY, K. P. Machine learning: a probabilistic perspective. [S.l.]: MIT press, 2012. Citado 2 vezes nas páginas 50 e 58.

NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th international conference on machine learning (ICML-10). [S.l.: s.n.], 2010. p. 807–814. Citado na página 57.

NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th international conference on machine learning (ICML-10). [S.l.: s.n.], 2010. p. 807–814. Citado na página 84.

NALLAPATI, R. et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. arXiv preprint arXiv:1602.06023, 2016. Citado na página 66.

NARAYANA, P.; BEVERIDGE, R.; DRAPER, B. A. Gesture recognition: Focus on the hands. In: Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2018. p. 5235–5244. Citado na página 97.

NEUMANN, L.; ZISSERMAN, A.; VEDALDI, A. Future event prediction: If and when. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. [S.l.: s.n.], 2019. p. 0–0. Citado na página 120.

NEVEROVA, N. et al. Moddrop: adaptive multi-modal gesture recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE, v. 38, n. 8, p. 1692–1706, 2015. Citado 2 vezes nas páginas 169 e 171.

NEVEROVA, N. et al. Moddrop: adaptive multi-modal gesture recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE, v. 38, n. 8, p. 1692–1706, 2016. Citado 7 vezes nas páginas 36, 37, 40, 74, 76, 165 e 178.

NEVEROVA, N. et al. Multi-scale deep learning for gesture detection and localization. In: SPRINGER. Workshop at the European conference on computer vision. [S.l.], 2014. p. 474–490. Citado 3 vezes nas páginas 40, 75 e 178.

NIU, K. et al. Improving description-based person re-identification by multi-granularity image-text alignments. arXiv preprint arXiv:1906.09610, 2019. Citado na página 66.

NIXON, M.; AGUADO, A. Feature extraction and image processing for computer vision. [S.l.]: Academic press, 2019. Citado na página 55.

PAN, S. J.; YANG, Q. A survey on transfer learning. IEEE Transactions on knowledge and data engineering, IEEE, v. 22, n. 10, p. 1345–1359, 2010. Citado 2 vezes nas páginas 59 e 61.

PASCANU, R.; MIKOLOV, T.; BENGIO, Y. On the difficulty of training recurrent neural networks. In: International conference on machine learning. [S.l.: s.n.], 2013. p. 1310–1318. Citado na página 64.

PASZKE, A. et al. Automatic differentiation in pytorch. 2017. Citado na página 89.

PAULINO, C. D. et al. Estatística Bayesiana. [S.l.]: Lisboa: Fundação Calouste Gulbenkian, 2018. v. 2ed. Citado 2 vezes nas páginas 67 e 68.

PAVLAKOS, G. et al. Kinect-based multimodal gesture recognition using a two-pass fusion scheme. In: IEEE. 2014 IEEE International Conference on Image Processing (ICIP). [S.l.], 2014. p. 1495–1499. Citado na página 75.

PENG, X. et al. Action and gesture temporal spotting with super vector representation. In: SPRINGER. European Conference on Computer Vision. [S.l.], 2014. p. 518–527. Citado na página 178.

PIGOU, A. C. A Treatise on Probability. [S.l.]: JSTOR, 1921. Citado na página 41.

PIGOU, L. et al. Sign language recognition using convolutional neural networks. In: SPRINGER. European Conference on Computer Vision. [S.l.], 2014. p. 572–578. Citado 2 vezes nas páginas 165 e 178.

PIGOU, L. et al. Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. International Journal of Computer Vision, Springer, v. 126, n. 2-4, p. 430–439, 2018. Citado 5 vezes nas páginas 75, 95, 96, 165 e 178.

PIRRI, F. et al. Anticipation and next action forecasting in video: an end-to-end model with memory. arXiv preprint arXiv:1901.03728, 2019. Citado na página 118.

QUEIROZ, F. M. et al. Estimating tridimensional coordinates of skeleton joints in a multicamera system. In: Anais do XIV Workshop de Visão Computacional - WVC 2018. [S.l.: s.n.], 2018. p. 108–114. Citado 6 vezes nas páginas 167, 168, 170, 171, 180 e 203.

QUEIROZ, F. M. de. Sistema de localização de gestos utilizando um sistema multicâmeras. 2019. Citado 2 vezes nas páginas 171 e 186.

RAUTARAY, S. S.; AGRAWAL, A. Vision based hand gesture recognition for human computer interaction: a survey. Artificial Intelligence Review, Springer, v. 43, n. 1, p. 1–54, 2015. Citado na página 37.

REN, S. et al. Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. [S.l.: s.n.], 2015. p. 91–99. Citado na página 164.

RODRIGUEZ, C.; FERNANDO, B.; LI, H. Action anticipation by predicting future dynamic images. In: Proceedings of the European Conference on Computer Vision (ECCV). [S.l.: s.n.], 2018. p. 0–0. Citado 2 vezes nas páginas 40 e 118.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning internal representations by error propagation. [S.l.], 1985. Citado na página 57.

RUSSAKOVSKY, O. et al. Imagenet large scale visual recognition challenge. International Journal of Computer Vision, Springer, v. 115, n. 3, p. 211–252, 2015. Citado 2 vezes nas páginas 56 e 83.

SAIKIA, S.; SAHARIA, S. A survey on vision-based dynamic gesture recognition. International Journal of Computer Applications, Foundation of Computer Science, v. 138, n. 1, 2016. Citado na página 37.

SALEME, E. B.; CELESTRINI, J. R.; SANTOS, C. A. S. Time evaluation for the integration of a gestural interactive application with a distributed mulsemedia platform. In: Proceedings of the 8th ACM on Multimedia Systems Conference. [S.l.: s.n.], 2017. p. 308–314. Citado na página 170.

SAMATELO, J. L. A.; SALLES, E. O. T. A new change detection algorithm for visual surveillance system. IEEE Latin America Transactions, IEEE, v. 10, n. 1, p. 1221–1226, 2012. Citado na página 79.

SANTOS, C. A. S.; NETO, A. N. R.; SALEME, E. B. An event driven approach for integrating multi-sensory effects to interactive environments. In: IEEE. 2015 IEEE International Conference on Systems, Man, and Cybernetics. [S.l.], 2015. p. 981–986. Citado 2 vezes nas páginas 167 e 170.

SANTOS, C. C. et al. Modelagem de um comportamento interacional entre homem e robô para um espaço inteligente baseado em visao computacional. Simpósio Brasileiro de Automaçao Inteligente, SBAI, 2017. Citado na página 170.

SCHUSTER, M.; PALIWAL, K. K. Bidirectional recurrent neural networks. IEEE transactions on Signal Processing, Ieee, v. 45, n. 11, p. 2673–2681, 1997. Citado na página 104.

SCHYDLO, P. et al. Anticipation in human-robot cooperation: A recurrent neural network approach for multiple action sequences prediction. In: IEEE. 2018 IEEE International Conference on Robotics and Automation (ICRA). [S.l.], 2018. p. 1–6. Citado 14 vezes nas páginas xvii, 115, 116, 117, 119, 125, 131, 134, 136, 137, 139, 141, 142 e 148.

SHI, W. et al. Edge computing: Vision and challenges. IEEE internet of things journal, IEEE, v. 3, n. 5, p. 637–646, 2016. Citado na página 82.

SHI, Y.; FERNANDO, B.; HARTLEY, R. Action anticipation with rbf kernelized feature mapping rnn. In: Proceedings of the European Conference on Computer Vision (ECCV). [S.l.: s.n.], 2018. p. 301–317. Citado na página 118.

SIMONYAN, K.; ZISSERMAN, A. Two-stream convolutional networks for action recognition in videos. In: Advances in neural information processing systems. [S.l.: s.n.], 2014. p. 568–576. Citado 2 vezes nas páginas 40 e 130.

SIMONYAN, K.; ZISSERMAN, A. Two-stream convolutional networks for action recognition in videos. In: Advances in neural information processing systems. [S.l.: s.n.], 2014. p. 568–576. Citado na página 72.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014. Citado 3 vezes nas páginas 40, 56 e 57.

SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014. Citado na página 83.

SRIVASTAVA, N. et al. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, JMLR. org, v. 15, n. 1, p. 1929–1958, 2014. Citado na página 58.

SZEGEDY, C. et al. Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2015. p. 1–9. Citado na página 56.

TSIRONI, E. et al. An analysis of convolutional long short-term memory recurrent neural networks for gesture recognition. Neurocomputing, Elsevier, v. 268, p. 76–86, 2017. Citado 11 vezes nas páginas xvi, 38, 40, 77, 86, 93, 95, 97, 105, 106 e 110.

VASWANI, A. et al. Attention is all you need. In: Advances in neural information processing systems. [S.l.: s.n.], 2017. p. 5998–6008. Citado 4 vezes nas páginas 40, 65, 66 e 104.

WAN, J. et al. Chalearn looking at people rgb-d isolated and continuous datasets for gesture recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. [S.l.: s.n.], 2016. p. 56–64. Citado 2 vezes nas páginas 77 e 87.

WANG, D.; YUAN, Y.; WANG, Q. Early action prediction with generative adversarial networks. IEEE Access, IEEE, v. 7, p. 35795–35804, 2019. Citado na página 119.

WANG, D.; YUAN, Y.; WANG, Q. Early action prediction with generative adversarial networks. IEEE Access, IEEE, v. 7, p. 35795–35804, 2019. Citado na página 120.

WANG, H.; FENG, J. Delving into 3d action anticipation from streaming videos. arXiv preprint arXiv:1906.06521, 2019. Citado na página 118.

WANG, H.; SCHMID, C. Action recognition with improved trajectories. In: Proceedings of the IEEE international conference on computer vision. [S.l.: s.n.], 2013. p. 3551–3558. Citado na página 110.

WANG, H.; WANG, L. Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2017. p. 499–508. Citado na página 74.

WANG, H.; WANG, L. Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2017. p. 499–508. Citado na página 169.

WANG, X. et al. Progressive teacher-student learning for early action prediction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2019. p. 3556–3565. Citado na página 120.

WANG, Y. et al. Tacotron: Towards end-to-end speech synthesis. arXiv preprint arXiv:1703.10135, 2017. Citado na página 66.

WEXELBLAT, A. Research challenges in gesture: Open issues and unsolved problems. In: SPRINGER. International Gesture Workshop. [S.l.], 1997. p. 1–11. Citado na página 72.

WU, D. et al. Deep Dynamic Neural Networks for Multimodal Gesture Segmentation and Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, v. 38, n. 8, p. 1583–1597, 2016. ISSN 01628828. Citado 3 vezes nas páginas 76, 96 e 165.

WU, D.; SHAO, L. Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., p. 724–731, 2014. ISSN 10636919. Citado 3 vezes nas páginas 75, 165 e 178.

YAO, A.; GOOL, L. V.; KOHLI, P. Gesture recognition portfolios for personalization. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, p. 1923–1930, 2014. ISSN 10636919. Citado na página 75.

ZEILER, M. D.; FERGUS, R. Visualizing and understanding convolutional networks. In: SPRINGER. European conference on computer vision. [S.l.], 2014. p. 818–833. Citado na página 60.

ZHAO, C.; PAN, W.; HU, H. Interactive indoor environment mapping through human-robot interaction. International Journal of Modelling Identification & Control, Citeseer, 2013. Citado na página 170.

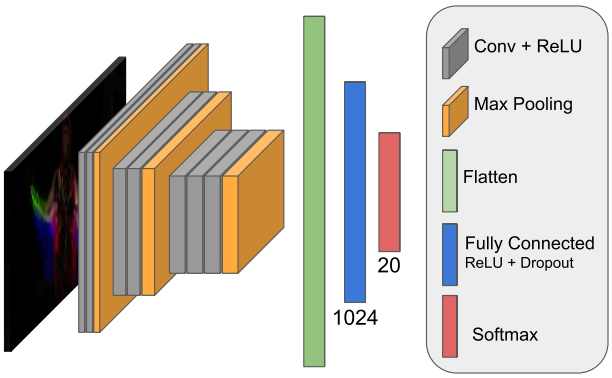
ZHU, G. et al. Multimodal gesture recognition using 3-d convolution and convolutional lstm. Ieee Access, IEEE, v. 5, p. 4517–4524, 2017. Citado na página 97.

Apêndices

APÊNDICE A – EXPERIMENTOS QUE CORROBORAM A ESCOLHA DO *STAR RGB*

Este apêndice apresenta todos os experimentos e resultados que levaram à proposta do *Star RGB* no Capítulo 3. No total, foram realizados 12 experimentos. Todos eles utilizaram uma mesma arquitetura de aprendizagem baseada num extrator de características formado pelas sete primeiras camadas convolucionais da CNN VGG16 e um classificador formado por uma MLP com apenas uma camada escondida de 1024 neurônios. A VGG16 foi escolhida por ser uma das CNNs mais utilizadas para testes de *baselines*. A Figura 63 ilustra a arquitetura utilizada nos experimentos. Cada experimento objetivou avaliar a capacidade de representação espacial e temporal de uma determinada técnica de pré-processamento apresentada na Seção 3.3.1.

Figura 63 – Arquitetura utilizada nos experimentos.



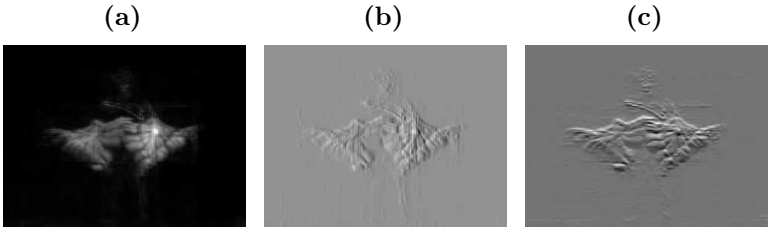
Fonte: próprio autor.

A.1 Descrição dos experimentos

Todos os experimentos realizados estão apresentados na Tabela 33. Cada um deles consiste de uma representação condensada de um vídeo do conjunto de dados Montalbano utilizando, ou a Equação (3.1), ou a (3.1), sempre em duas versões: uma que pondera a informação temporal e a outra sem tal ponderação. Para cada tipo de representação foi realizada uma busca utilizando um conjunto de valores (*grid search* a fim de chegar aos hiperparâmetros (também presentes na Tabela 33) que trouxessem os melhores resultados. Uma breve explanação sobre cada experimento é descrita a seguir:

- O $Star_{sobel}$ aplica a proposta apresentada em Barros et al. (2014a) sobre a base de dados Montalbano. Assim, a imagem de três canais fornecida à VGG16 é composta de uma imagem resultante da *representação star* (Equação (3.1)), juntamente com os resultados da filtragem por um filtro de Sobel em suas direções x e y. Um exemplo dos três canais resultantes pode ser visto na Figura 64. O termo w_k da Equação (3.1) foi desativado ($w_k = 1$) para não ponderar cada imagem de acordo com a sua ordem temporal. O objetivo desse experimento é avaliar a proposta da *representação star* sobre o conjunto de dados Montalbano e obter resultados que sirvam de base comparativa para os demais experimentos.

Figura 64 – Exemplo de entrada para $Star_{sobel}$. (a) imagem da representação star em escala de cinza e os resultados do filtro de Sobel em (b) x e (c) y.



Fonte: próprio autor.

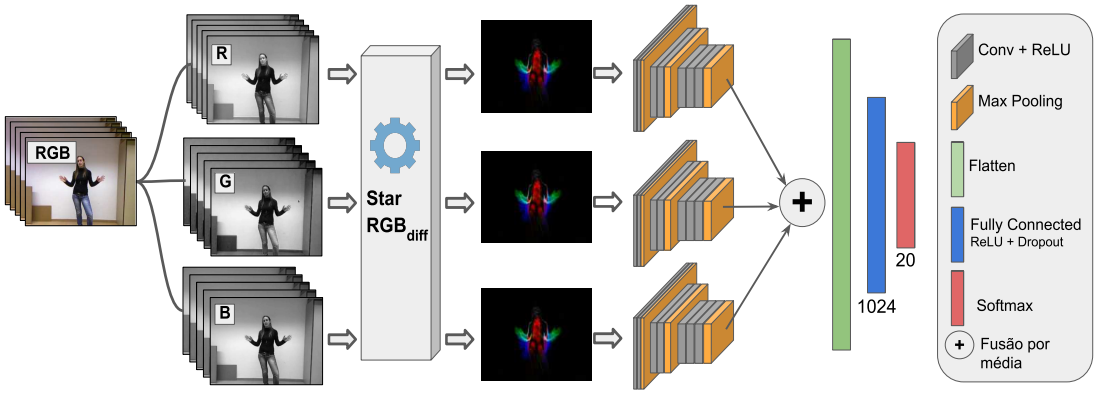
- O $Star_{gray}$ elimina os dois canais com os resultados do filtro de Sobel em $Star_{sobel}$, e fornece uma imagem com três canais iguais apenas triplicando a imagem em tons de cinza da *representação star*. Neste experimento será possível observar qual a contribuição das imagens resultantes do filtro de Sobel para a resolução do problema. Aqui, w_k também foi desativado.

- O $Star_{cos}$ têm a mesma configuração do $Star_{gray}$, porém, utiliza a equação de similaridade (3.5) ao invés da (3.1). O objetivo deste experimento é medir a contribuição da informação de cor para o problema e reconhecimento de gestos. Note-se que, assim como nos outros dois experimentos anteriores, aqui, a entrada da VGG16 também é uma imagem em tons de cinza com três canais iguais.

- O $StarRGB_{diff}$ gera uma imagem com três canais distintos, assim como descrito na Seção 3.3.1, no entanto, utiliza a equação de similaridade (3.5) ao invés da (3.1). O intuito desse experimento é mostrar a contribuição da informação temporal inserida pela divisão do vídeo em três partes em relação aos experimentos $Star_{gray}$ e $Star_{sobel}$.

- O $StarRGB$ tem a mesmas configuração do $StarRGB_{diff}$, no entanto, utiliza a equação de similaridade (3.5) ao invés da (3.1). Em outras palavras, este utiliza a técnica apresentada na Seção 3.3.1 como proposta principal de pré-processamento. Comparando os resultados deste experimento com os experimentos anteriores, será

Figura 65 – Representação completa da proposta do $Star3RGB_{diff}$. O $Star3RGB_{diff+w}$ é obtido por meio da ativação do termo de ponderação w_k.



Fonte: Próprio autor.

possível observar o ganho da junção da informação de cor por meio da aplicação da Equação (3.5) com a representação temporal fornecida pela divisão do vídeo em três partes.

- O $Star3RGB_{diff}$ representa uma alternativa ao $StarRGB$ no sentido de tentar melhorar a informação de cor extraída do vídeo. Nele, ao invés de transformar cada imagem RGB em tons de cinza antes de aplicar a Equação(3.1), a técnica do $StarRGB_{diff}$ é aplicada a cada canal de cor da imagem original (R,G,B), gerando assim três imagens RGBs distintas. Em seguida, cada imagem RGB é dada como entrada para três VGG16 com as mesmas configurações, mas que não compartilham os pesos, o que resulta em três mapas de características distintos, um para acada imagem. Esses mapas de características são então fundidos por meio de uma média simples, e dados como entrada para uma MLP. Essa proposta não foi abordada anteriormente, pois, além de utilizar mais recursos para capturar a informação de cor dos vídeos, não obteve resultados que justificassem a sua utilização em relação ao *Star RGB*. Por outro lado, mostra a importância da informação de cor quando comparado com o $StarRGB_{diff}$. O processo completo deste experimento pode ser visto na Figura 65.
- Os outros experimentos implementam uma versão idêntica a cada um dos experimentos anteriores, no entanto, utilizam ponderação temporais entre as imagens por meio do termo w_k. Dessa forma, será possível comparar como o termo de ponderação w_k contribui para representação temporal do vídeo. Sendo $w_k = k/N \; \forall \; k \in \{1,2,3,...,N\}$, onde k representa o índice de cada *frame* e N é o número total de *frames* do vídeo.

Todos os experimentos apresentados na Tabela 33 foram realizados com as mesmas configurações de *hardware* e *software* apresentadas na Seção 3.3.3.

Tabela 33 – Resultados dos experimentos. O hiperparâmetro *lr* corresponde à taxa de aprendizagem do extrator e do classificador, respectivamente. *bs* é tamanho do lote e *wd* é a taxa de decaimento dos pesos utilizado no regularizador do tipo L1.

Nome	Equação de similaridade	Hiperparâmetros	Acurácia
Sem ponderação temporal ($w_k = 1$)			
$Star_{sobel}$	(3.1)	lr = (1e-5,1e-4), bs = (96), wd = (1e-5)	79,65%
$Star_{gray}$	(3.1)	lr = (1e-4,1e-3), bs = (96), wd = (1e-5)	83,31%
$Star_{cos}$	(3.5)	lr = (1e-5,1e-4), bs = (96), wd = (1e-5)	84,10%
$StarRGB_{diff}$	(3.1)	lr = (1e-4,1e-3), bs = (96), wd = (1e-5)	87,17%
$Star3RGB_{diff}$	(3.1)	lr = (1e-5,1e-4), bs = (96), wd = (1e-5)	88,62%
$StarRGB$	(3.5)	lr = (1e-5,1e-4), bs = (96), wd = (1e-5)	**89,85**%
Com ponderação temporal ($w_k = k/N$)			
$Star_{sobel+w}$	(3.1)	lr = (1e-5,1e-4), bs = (96), wd = (1e-5)	83,40%
$Star_{gray+w}$	(3.1)	lr = (1e-4,1e-3), bs = (96), wd = (1e-5)	84,12%
$Star_{cos+w}$	(3.5)	lr = (1e-5,1e-4), bs = (96), wd = (1e-5)	84,93%
$StarRGB_{diff+w}$	(3.1)	lr = (1e-4,1e-3), bs = (96), wd = (1e-5)	86,39%
$Star3RGB_{diff+w}$	(3.1)	lr = (1e-5,1e-4), bs = (96), wd = (1e-5)	88,82%
$StarRGB_{w}$	(3.5)	lr = (1e-5,1e-4), bs = (96), wd = (1e-5)	89,27%

Fonte: próprio autor.

A.2 Discussão dos resultados

Como pode ser observado, o pior resultado foi alcançado pela proposta original em (BARROS et al., 2014a) ($Star_{sobel}$), a qual alcançou apenas 79,65% de acurácia média. Note-se que o resultado do $Star_{gray}$ (83,31%) mostra que não é necessário utilizar as imagens resultantes dos filtro de Sobel, o que já era esperado, uma vez que experimentos empíricos mostraram que as primeiras camadas de uma CNN consistem basicamente de filtros detectores de bordas (KRIZHEVSKY; SUTSKEVER; HINTON, 2012a). Já o experimento $Star_{cos}$ (84,10%) mostra o ganho do uso da informação de cor fornecida pela Equação (3.5). Importante observar que os experimentos $Star_{sobel+w}$ (83,40%), $Star_{gray+w}$ (84,12%) e $Star_{cos+w}$ (84,93%) obtiveram melhores resultados que as suas respectivas versões sem ponderação temporal, $Star_{sobel}$ (79,65%), $Star_{gray}$ (83,10%) e $Star_{cos}$ (84,10%). Isso mostra a deficiência de tais propostas ao tentar resolver o problema utilizando representações em tons de cinza resultantes da aplicação das Equações 3.1 e 3.5.

Analisando apenas as abordagens que objetivam melhorar a representação da informação temporal por meio da divisão o vídeo em três partes, aquelas que calculam a similaridade entre os *frames* utilizando a diferença absoluta entre eles ($StarRGB_{diff}$ (87,17%), $Star3RGB_{diff}$ (88,62%), $StarRGB_{diff+w}$ (86,39%) e $Star3RGB_{diff+w}$ (88,82%)) alcançaram piores resultados que aquelas que utilizam a distância cosseno ($StarRGB$ (89,85%) e $StarRGB_{w}$ (89,27%)). No entanto, note-se a importância da informação de cor apresentada pelo experimento $Star3RGB_{diff}$. Nele, como cada canal do vídeo gera uma

imagem distinta, as três CNNs foram capazes de melhorar a extração da informação de cor do vídeo quando comparada às demais. Porém, mesmo com a alta capacidade de aprendizado (utiliza três VGG16), não se conseguiu resultados melhores que a proposta principal do *Star RGB* ($StarRGB$). Por fim, é importante observar ainda que a ponderação temporal entre os *frames* do vídeo não trouxe melhores resultados para o $StarRGB_{diff+w}$ e para o $StarRGB_{w}$, em comparação às suas respectivas versões não ponderadas, $StarRGB_{diff}$ e $StarRGB$. Como tais representações já possuem intrinsecamente parte da informação temporal contida no vídeo, utilizar o termo de ponderação $w_k = k/N \; \forall \; k \in \{1,2,3,...,N\}$ prejudica a informação espacial da imagem RGB resultante. Esse termo diminui de maneira significativa o valores resultantes do cálculo da similaridade entre os primeiros *frames* das sequências que compõem cada canal RGB. Isso mostra o porquê de fixar o valor de $w_k = 1$ na Equação (3.5).

Sendo assim, as análises apresentadas trazem os dados necessários para justificar a escolha do *Star RGB* como proposta principal para a representação do movimento de um gesto contido um vídeo.

A.3 Experimentos adicionais

Após os resultados fornecidos pelo *Star RGB* para reconhecimento de gesto, foi proposta uma arquitetura própria baseada em oito camadas convolucionais 3D a qual objetivava extrair as informações temporais de um vídeo de forma a não necessitar de uma etapa de pré-processamento, ou seja, era um proposta para aprendizagem fim a fim. A ideia foi implementar uma versão baseada em aprendizado profundo da clássica técnica de representação temporal conhecida como cuboid (DOLLÁR et al., 2005). Após alguns experimentos iniciais, os resultados não ultrapassaram os 60% de acurácia média sobre o Montalbano, por esse motivo, tal proposta não foi levada à diante.

Outra abordagem analisada para a representação temporal de um vídeo foi a Decomposição de Posto Tensorial por meio do método PARAFAC (do inglês *Parallel factor analysis*) (HARSHMAN; LUNDY, 1994). O objetivo era decompor um vídeo em três principais componentes: informações de cor nas direções x, y e z (temporal). Após a decomposição, cada uma das três características seriam passadas para redes neurais distintas, que seriam fundidas e depois classificadas. Apesar de ser interessante, essa ideia não foi concluída, pois o PARAFAC, além de utilizar um processo de otimização iterativo, necessita de um considerável poder computacional e capacidade de memória para manipular os tensores. O que dificulta a decomposição de longos vídeos. Dessa forma, nem foram obtidos resultados preliminares para o reconhecimento de gestos.