UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO, DE CIÊNCIAS EXATAS E EDUCAÇÃO
DEPARTAMENTO DE ENG. DE CONTROLE, AUTOMAÇÃO E COMPUTAÇÃO
CURSO DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Lauro Weingärtner Neto

# Reconhecimento de atividades em grupo baseado em mapa de calor

Blumenau
2023

Lauro Weingärtner Neto

# Reconhecimento de atividades em grupo baseado em mapa de calor

Trabalho de Conclusão de Curso de Graduação em Engenharia de Controle e Automação do Centro Tecnológico, de Ciências Exatas e Educação da Universidade Federal de Santa Catarina como requisito para a obtenção do título de Engenheiro de Controle e Automação.
Orientador: Prof. Marcos Vinicius Matsuo, Dr.

Blumenau
2023

Ficha de identificação da obra

A ficha de identificação é elaborada pelo próprio autor.
Orientações em:
http://portalbu.ufsc.br/ficha

Lauro Weingärtner Neto

**Reconhecimento de atividades em grupo baseado em mapa de calor**

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de "Engenheiro de Controle e Automação" e aprovado em sua forma final pelo Curso de Graduação em Engenharia de Controle e Automação.

Blumenau, 14 de dezembro de 2023.

**Banca Examinadora:**


Documento assinado digitalmente
**Marcos Vinicius Matsuo**
Data: 08/01/2024 09:29:13-0300
CPF: ***.580.029-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Marcos Vinicius Matsuo, Dr.
Universidade Federal de Santa Catarina


Documento assinado digitalmente
**Carlos Roberto Moratelli**
Data: 10/01/2024 10:45:29-0300
CPF: ***.999.369-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Carlos Roberto Moratelli, Dr.
Universidade Federal de Santa Catarina


Documento assinado digitalmente
**Maiquel de Brito**
Data: 10/01/2024 09:22:12-0300
CPF: ***.657.150-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Maiquel de Brito, Dr.
Universidade Federal de Santa Catarina

Dedico este trabalho a meus pais, Mauricio e Angela, pelo
amor e apoio incondicionais.

## AGRADECIMENTOS

Gostaria de expressar minha profunda gratidão à Universidade Federal de Santa Catarina, campus de Blumenau, por fornecer um ambiente acadêmico excepcional e uma graduação de alta qualidade em Engenharia de Controle e Automação, que foi fundamental para o meu desenvolvimento profissional e pessoal. Um agradecimento especial a todos os professores da área de computação, cujo conhecimento e dedicação não apenas expandiram meus horizontes na área, mas também me instigaram a buscar continuamente mais aprendizado e crescimento. Um agradecimento particular ao meu orientador Prof. Marcos Vinicius Matsuo, cujos conhecimentos passados na disciplina de Visão Computacional em Robótica foram cruciais para a escolha do tema deste trabalho. Sua orientação, paciência e sugestões valiosas foram inestimáveis na elaboração deste trabalho, ajudando-me a superar desafios e aprimorar minha pesquisa.

O comportamento é um espelho em que cada um vê a sua própria imagem. (GOETHE)

## RESUMO

Este estudo objetivou a implementação de um algoritmo baseado em mapas de calor para classificar atividades em grupos de indivíduos. O sistema é composto por duas partes: geração dos mapas de calor e classificação das atividades humanas. Para a geração dos mapas, aplicaram-se conceitos da termodinâmica, especificamente da transferência de calor. A classificação foi realizada com o uso de diversas arquiteturas de redes neurais convolucionais, incluindo *AlexNet*, *SqueezeNet*, *GoogLeNet* e *ResNet*. Diferentemente de outros algoritmos que focam em atividades individuais, este trabalho concentra-se exclusivamente na identificação de atividades em grupo. Cada rede neural foi treinada (utilizando a técnica de *Transfer Learning*) duas vezes: uma com 3 classes de atividades em grupo e outra com 4 classes. Com 3 classes, a maioria das arquiteturas atingiu acurácia acima de 80% no conjunto de teste. Ao adicionar uma quarta classe, observou-se uma redução na acurácia e sinais de *overfitting* na arquitetura *ResNet*. No entanto, a variante ResNet-50, de maior complexidade computacional, alcançou uma acurácia de $82,38\%$.


**Palavras-chave**: Atividade em grupo; Mapa de calor; Rede Neural Convolucional.

**ABSTRACT**

This study aimed to implement an algorithm based on heatmaps to classify activities among groups of individuals. The system consists of two parts: the generation of heat maps and the classification of human activities. To generate the maps, thermodynamic concepts were applied, specifically those of the heat transfer field. Classification was performed using several convolutional neural network architectures, including *AlexNet*, *SqueezeNet*, *GoogLeNet* and *ResNet*. Unlike other algorithms that focus on individual activities, this work focuses exclusively on identifying group activities. Each neural network was trained (using the *Transfer Learning* technique) twice: once with 3 classes of group activities and another with 4 classes. With 3 classes, most architectures achieved accuracy above 80% on the test set. When adding a fourth class, a reduction in accuracy and signs of *overfitting* were observed in the *ResNet* architecture. However, the ResNet-50 variant, with greater computational complexity, achieved an accuracy of 82.38%.**Keywords**: Group activitiy; Heatmap; Convolutional Neural Network.

## LISTA DE FIGURAS


Figura 1 – Função de duas variáveis. . . . . 19
Figura 2 – Estrutura de uma imagem *raster*. . . . . 20
Figura 3 – Representação de uma imagem vetorial. . . . . 21
Figura 4 – Imagem binária. . . . . 21
Figura 5 – Imagem em escala de cinza. . . . . 22
Figura 6 – Imagem colorida. . . . . 23
Figura 7 – Operações monádicas de alteração de brilho, alteração de contraste e obtenção do negativo em imagens digitais. . . . . 25
Figura 8 – Operações monádicas de limiarização e posterização em imagens digitais. 26
Figura 9 – Adição de duas imagens digitais. . . . . 28
Figura 10 – Multiplicação entre duas imagens digitais. . . . . 28
Figura 11 – Transformação espacial geométrica de escalamento. . . . . 30
Figura 12 – Transformação espacial geométrica de rotação. . . . . 31
Figura 13 – Transformação espacial geométrica de translação. . . . . 32
Figura 14 – Transformação espacial geométrica de espelhamento. . . . . 33
Figura 15 – Transformação espacial geométrica de cisalhamento. . . . . 34
Figura 16 – Filtragem 2D em imagens digitais. . . . . 35
Figura 17 – Varredura do *kernel* na filtragem 2D. . . . . 35
Figura 18 – Decaimento térmico exponencial. . . . . 38
Figura 19 – Rede neural *feedforward* densamente conectada. . . . . 40
Figura 20 – Redes pré-treinadas no *dataset ImageNet*. . . . . 42
Figura 21 – Tipos de camadas em uma rede neural convolucional. . . . . 43
Figura 22 – Arquiteturas *AlexNet*, *SqueezeNet* e *GoogLeNet*. . . . . 44
Figura 23 – Arquitetura *ResNet*. . . . . 46
Figura 24 – Algoritmo de detecção de atividades em grupo. . . . . 49
Figura 25 – Algoritmo de obtenção de mapas de calor. . . . . 50
Figura 26 – Etapas para obter a posição de cada indivíduo. . . . . 51
Figura 27 – *Bounding-boxes* e centroides em um frame. . . . . 52
Figura 28 – Divisão de uma imagem por *patches*. . . . . 53
Figura 29 – Mapeamento entre *patches* e *pixels*. . . . . 53
Figura 30 – Conversão entre posições com coordenadas de *pixels* e *patches*. . . . . 54
Figura 31 – *Patches* com múltiplas trajetórias. . . . . 55
Figura 32 – Algoritmo de obtenção das trajetórias de cada *patch*. . . . . 56
Figura 33 – Estrutura de dados para as trajetórias de cada *patch*. . . . . 57
Figura 34 – Algoritmo para o cálculo das fontes de calor de cada indivíduo em um *frame*. . . . . 59
Figura 35 – Fontes de calor para um *frame*. . . . . 60

Figura 36 – Exemplo de trajetórias para 2 indivíduos. . . . . 60
Figura 37 – Fontes de calor resultantes do exemplo. . . . . 61
Figura 38 – Algoritmo de difusão espacial dos mapas de calor. . . . . 63
Figura 39 – Mapa de calor após a difusão espacial. . . . . 64
Figura 40 – Mapa de calor como superfície. . . . . 64
Figura 41 – Exemplo do algoritmo de difusão de calor. . . . . 65
Figura 42 – Mapas de calor com diferentes valores de decaimento temporal. . . . . 66
Figura 43 – Mapas de calor com diferentes valores de difusão espacial. . . . . 66
Figura 44 – Mapa de calor como imagem em escala de cinza. . . . . 67
Figura 45 – Etapas de elaboração e utilização do *dataset*. . . . . 68
Figura 46 – Algoritmo de obtenção dos mapas de calor de uma atividade em grupo. 70
Figura 47 – Mapas de calor dos *frames* alvo de uma atividade. . . . . 71
Figura 48 – Mapas de calor das atividades selecionadas. . . . . 72
Figura 49 – Mapa de calor original. . . . . 74
Figura 50 – Mapas de calor sintéticos. . . . . 75
Figura 51 – Curvas de treinamento e validação com 3 classes de atividades em grupo para a *AlexNet*. . . . . 77
Figura 52 – Curvas de treinamento e validação com 3 classes de atividades para a *SqueezeNet*. . . . . 79
Figura 53 – Curvas de treinamento e validação com 3 classes de atividades para a *GoogLeNet*. . . . . 81
Figura 54 – Curvas de treinamento e validação com 3 classes de atividades para a *ResNet-18*. . . . . 83
Figura 55 – Curvas de treinamento e validação com 3 classes de atividades para a *ResNet-50*. . . . . 84
Figura 56 – Curvas de treinamento e validação com 4 classes de atividades para a *AlexNet*. . . . . 86
Figura 57 – Curvas de treinamento e validação com 4 classes de atividades para a *SqueezeNet*. . . . . 88
Figura 58 – Curvas de treinamento e validação com 4 classes de atividades para a *GoogLeNet*. . . . . 90
Figura 59 – Curvas de treinamento e validação com 4 classes de atividades para a *ResNet-18*. . . . . 92
Figura 60 – Curvas de treinamento e validação com 4 classes de atividades para a *ResNet-50*. . . . . 94

## LISTA DE QUADROS

Quadro 1 – Descrições das atividades em grupo. . . . . 69

## LISTA DE TABELAS


Tabela 1 – Hiperparâmetros de treinamento. . . . . 76
Tabela 2 – Métricas de desempenho para a *AlexNet* com 3 classes de atividades em grupo. . . . . 78
Tabela 3 – Matriz de confusão para a *AlexNet* com 3 classes de atividades em grupo (conjunto de teste). . . . . 78
Tabela 4 – Métricas de desempenho para a *SqueezeNet* com 3 classes de atividades em grupo. . . . . 78
Tabela 5 – Matriz de confusão para a *SqueezeNet* com 3 classes de atividades em grupo. . . . . 80
Tabela 6 – Métricas de desempenho para a *GoogLeNet* com 3 classes de atividades em grupo. . . . . 81
Tabela 7 – Matriz de confusão para a *GoogLeNet* com 3 classes de atividades em grupo. . . . . 82
Tabela 8 – Métricas de desempenho para a *ResNet-18* com 3 classes de atividades em grupo. . . . . 82
Tabela 9 – Matriz de confusão para a *ResNet-18* com 3 classes de atividades em grupo. . . . . 84
Tabela 10 – Métricas de desempenho para a *ResNet-50* com 3 classes de atividades em grupo. . . . . 85
Tabela 11 – Matriz de confusão para a *ResNet-50* com 3 classes de atividades em grupo. . . . . 85
Tabela 12 – Métricas de desempenho para a *AlexNet* com 4 classes de atividades em grupo. . . . . 87
Tabela 13 – Matriz de confusão para a *AlexNet* com 4 classes de atividades em grupo. 87
Tabela 14 – Métricas de desempenho para a SqueezeNet com 4 classes de atividades em grupo. . . . . 89
Tabela 15 – Matriz de confusão para a *SqueezeNet* com 4 classes de atividades em grupo. . . . . 89
Tabela 16 – Métricas de desempenho para a *GoogLeNet* com 4 classes de atividades em grupo. . . . . 91
Tabela 17 – Matriz de confusão para a *GoogLeNet* com 4 classes de atividades em grupo. . . . . 91
Tabela 18 – Métricas de desempenho para a *ResNet-18* com 4 classes de atividades em grupo. . . . . 93
Tabela 19 – Matriz de confusão para a *ResNet-18* com 4 classes de atividades em grupo. . . . . 93

Tabela 20 – Métricas de desempenho para a *ResNet-50* com 4 classes de atividades em grupo. . . . . 95
Tabela 21 – Matriz de confusão para a *ResNet-50* com 4 classes de atividades em grupo. . . . . 95
Tabela 22 – *Ranking* de acurácias para 3 classes de atividades em grupo. . . . . 96
Tabela 23 – *Ranking* de acurácias para 4 classes de atividades em grupo. . . . . 97

## LISTA DE ABREVIATURAS E SIGLAS

| | |
|---|---|
| ARN | *Attention Relation Network* |
| CNN | *Convolutional Neural Network* |
| EPS | *Encapsulated PostScript* |
| FPGA | *Field-Programmable Gate Array* |
| GIF | *Graphics Interchange Format* |
| IA | Inteligência Artificial |
| ILSVRC | *Large Scale Visual Recognition Challenge* |
| JPEG | *Joint Photographic Experts Group* |
| LR | *Learning Rate* |
| LSTM | *Long Short-Term Memory* |
| PDF | *Portable Document Format* |
| PNG | *Portable Network Graphics* |
| RGB | *Red, Green and Blue* |
| SVG | *Scalable Vector Graphics* |
| TIFF | *Tagged Image File Format* |

**SUMÁRIO**


**1 INTRODUÇÃO . . . . . 17**
1.1 OBJETIVO GERAL . . . . . 18
1.2 OBJETIVOS ESPECÍFICOS . . . . . 18
1.3 ORGANIZAÇÃO DOS CAPÍTULOS . . . . . 18
**2 FUNDAMENTAÇÃO TEÓRICA . . . . . 19**
2.1 IMAGENS DIGITAIS . . . . . 19
**2.1.1 Imagens *raster* e vetoriais . . . . . 19**
**2.1.2 Imagens binárias, em escala de cinza e coloridas . . . . . 20**
2.2 PROCESSAMENTO DIGITAL DE IMAGENS . . . . . 23
**2.2.1 Operações monádicas . . . . . 23**
*2.2.1.1 Alteração de brilho . . . . . 24*
*2.2.1.2 Alteração de contraste . . . . . 24*
*2.2.1.3 Obtenção de negativo . . . . . 24*
*2.2.1.4 Limiarização . . . . . 24*
*2.2.1.5 Posterização . . . . . 26*
**2.2.2 Operações diádicas . . . . . 27**
*2.2.2.1 Adição . . . . . 27*
*2.2.2.2 Multiplicação . . . . . 27*
**2.2.3 Transformações geométricas . . . . . 28**
*2.2.3.1 Escalamento . . . . . 29*
*2.2.3.2 Rotação . . . . . 29*
*2.2.3.3 Translação . . . . . 30*
*2.2.3.4 Espelhamento . . . . . 31*
*2.2.3.5 Cisalhamento . . . . . 31*
**2.2.4 Filtragem 2D . . . . . 32**
2.3 TRANSFERÊNCIA DE CALOR . . . . . 36
**2.3.1 Modos de transferência de calor . . . . . 36**
**2.3.2 Difusão térmica . . . . . 37**
**2.3.3 Decaimento temporal . . . . . 37**
2.4 *MACHINE LEARNING* . . . . . 39
**2.4.1 Redes neurais artificiais . . . . . 39**
**2.4.2 Treinamento e descida do gradiente . . . . . 40**
**2.4.3 Redes neurais convolucionais . . . . . 42**
*2.4.3.1 AlexNet . . . . . 43*
*2.4.3.2 SqueezeNet . . . . . 45*
*2.4.3.3 GoogLeNet . . . . . 45*
*2.4.3.4 ResNet . . . . . 46*

**2.4.4** ***Transfer Learning*** . . . . . **47**
**2.4.5** **Generalização, *overfitting* e *Data Augmentation*** . . . . . **47**
**3** **DESENVOLVIMENTO** . . . . . **49**
3.1 OBTENÇÃO DOS MAPAS DE CALOR DAS ATIVIDADES EM GRUPO 50
**3.1.1** **Posições dos indivíduos** . . . . . **51**
**3.1.2** **Trajetórias em cada patch** . . . . . **53**
**3.1.3** **Fontes de calor em cada frame** . . . . . **55**
**3.1.4** **Difusão espacial do calor** . . . . . **62**
3.2 CLASSIFICAÇÃO DAS ATIVIDADES EM GRUPO . . . . . 67
**3.2.1** **Obtenção do *dataset* de mapas de calor** . . . . . **69**
**3.2.2** **Criação do algoritmo de *Data Augmentation*** . . . . . **72**
**3.2.3** **Treinamento das redes neurais** . . . . . **73**
**4** **RESULTADOS** . . . . . **76**
4.1 CLASSIFICAÇÃO COM 3 CLASSES DE ATIVIDADES EM GRUPO 76
**4.1.1** ***AlexNet*** . . . . . **76**
**4.1.2** ***SqueezeNet*** . . . . . **78**
**4.1.3** ***GoogLeNet*** . . . . . **81**
**4.1.4** ***ResNet-18*** . . . . . **82**
**4.1.5** ***ResNet-50*** . . . . . **83**
4.2 CLASSIFICAÇÃO COM 4 CLASSES DE ATIVIDADES EM GRUPO 86
**4.2.1** ***AlexNet*** . . . . . **86**
**4.2.2** ***SqueezeNet*** . . . . . **88**
**4.2.3** ***GoogLeNet*** . . . . . **90**
**4.2.4** ***ResNet-18*** . . . . . **92**
**4.2.5** ***ResNet-50*** . . . . . **94**
4.3 ANÁLISE DOS RESULTADOS . . . . . 96
**5** **CONCLUSÃO** . . . . . **98**
**REFERÊNCIAS** . . . . . **100**

# 1 INTRODUÇÃO

Dentro do campo de estudos sobre rastreamento de pessoas, pesquisas sobre o reconhecimento automático de atividades humanas em grupo visam desenvolver metodologias para identificar movimentações espaciais de grupos de indivíduos. Especificamente, o reconhecimento de atividades humanas ganha relevância por conseguir suprir demandas em diversas áreas como entretenimento, saúde, simulações e sistemas de vigilância (voltado para segurança pública) (D'SA; PRASAD, 2019).

Umas das principais dificuldades do reconhecimento de atividades humanas está na diferenciação entre atividades coletivas e individuais. Desse modo, a identificação de quando ocorre uma interação entre os indivíduos é essencial para reconhecer atividades em grupo (CHANG; ZHENG; ZHANG, J., 2015). Em particular, as atividades humanas podem ser classificadas analisando as interações entre indivíduos ou considerando ações e movimentações coletivas. Um exemplo de trabalho nessa área é o de (SHU *et al.*, 2021) que utiliza a arquitetura de rede neural LSTM (*Long Short-Term Memory*) para reconhecer primeiramente atividades a nível de indivíduo, para em seguida classificar as atividades humanas a nível de grupo.

Para a identificação precisa de atividades humanas, uma variedade de algoritmos e tecnologias é empregada, adaptando-se ao contexto e aos requisitos específicos da tarefa. Segundo (D'SA; PRASAD, 2019), a categorização do reconhecimento de atividades humanas depende fundamentalmente dos dispositivos de captura utilizados, os quais se dividem principalmente em dois grupos: baseados em sensores e em técnicas de visão computacional. Essa subdivisão enfatiza a importância da escolha do método de captura na determinação da abordagem mais adequada para o reconhecimento eficiente de padrões de atividades.

No contexto da identificação de atividades em grupo por meio de técnicas de visão computacional, diversas fases de extração de informações são implementadas, as quais são frequentemente complementadas por métodos avançados de classificação, como as redes neurais. Dentro deste contexto, (WANG; MOHAMED, 2023) adotam a técnica de processamento digital de imagem conhecida como *skeleton*, integrada a um classificador ARN (*Attention Relation Network*). Uma abordagem alternativa para a representação de atividades humanas envolve a geração de imagens baseadas em princípios físicos, como a mecânica dos fluidos, mecânica dos sólidos, energia e entropia, conforme discutido por (ZHANG, X.; YU, Q.; YU, H., 2018). Estas representações podem então ser analisadas e classificadas por uma variedade de métodos.

A fim de contribuir com a área de estudo de identificação de atividades em grupo, o presente trabalho propõe desenvolver um algoritmo para a geração e classificação de mapas de calor que representam atividades de grupos de indivíduos. A etapa de geração dos mapas de calor baseia-se no algoritmo proposto por (LIN *et al.*, 2013), que utiliza equa-

ções e princípios da termodinâmica. Posteriormente, a etapa de classificação adota uma abordagem diferente (em relação a (LIN *et al.*, 2013)), empregando diferentes arquiteturas de Redes Neurais Convolucionais.

## 1.1 OBJETIVO GERAL

O principal objetivo do trabalho é a implementação de um sistema capaz de gerar mapas de calor que representem atividades de grupos de indivíduos, possibilitando a posterior classificação desses comportamentos.

## 1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos que devem ser alcançados, de modo a atingir o objetivo geral, são:

- Obter um conjunto de dados contendo a representação de diferentes atividades em grupos e processá-lo para extrair as informações necessárias para o sistema de desenvolvido.
- Aplicar conhecimentos da termodinâmica para a geração de mapas de calor.
- Treinar diferentes redes neurais para a tarefa específica de identificação de atividades em grupo.
- Avaliar os resultados parciais do sistema de modo a verificar o funcionamento das etapas intermediárias de processamento.
- Avaliar a acurácia do sistema na identificação de atividades em grupo utilizando diversas métricas de desempenho.

## 1.3 ORGANIZAÇÃO DOS CAPÍTULOS

A estrutura deste trabalho é organizada da seguinte forma: o Capítulo 2 aborda conceitos teóricos essenciais para o desenvolvimento do trabalho, estabelecendo os fundamentos necessários para a compreensão das técnicas empregadas; em seguida, o Capítulo 3 detalha os algoritmos e as etapas de processamento envolvidas na obtenção dos mapas de calor a partir de informações sobre o deslocamento dos indivíduos em uma cena, bem como discute a classificação das atividades em grupo utilizando redes neurais convolucionais; o Capítulo 4 é dedicado à apresentação dos resultados alcançados no treinamento das redes neurais e na classificação das atividades em grupo, onde são analisadas as eficácias e as limitações das abordagens adotadas; por fim, o Capítulo 5 trata da conclusão deste trabalho, recapitulando os principais pontos e realizações do trabalho, destacando as suas contribuições e propondo possíveis melhorias e direções futuras para os algoritmos desenvolvidos.

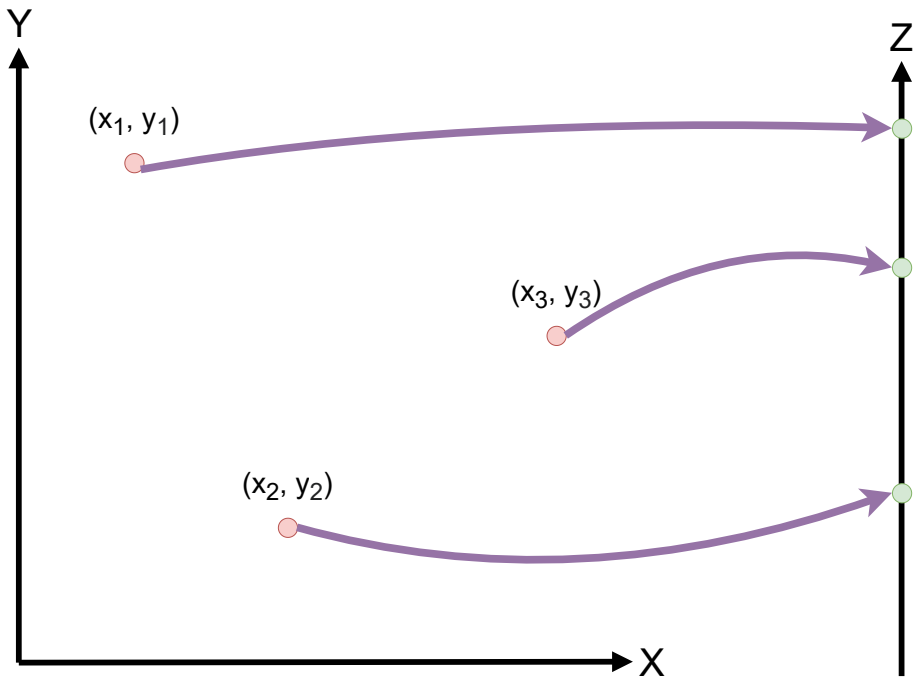
# 2 FUNDAMENTAÇÃO TEÓRICA

As seguintes seções possuem a fundamentação teórica necessária para aplicar os algoritmos desenvolvidos, assim como interpretar os resultados deste trabalho.

## 2.1 IMAGENS DIGITAIS

De acordo com Gonzalez e Woods (2009) pode-se definir uma imagem como uma função matemática $z = f(x, y)$ com duas variáveis independentes $x$ e $y$, conforme ilustrado pela Figura 1. Ademais, quando as grandezas $x$, $y$ e $z$ se limitam a quantidades finitas e discretas, é caracterizada uma imagem digital.

Figura 1 – Função de duas variáveis.




Fonte: elaborado pelo autor.

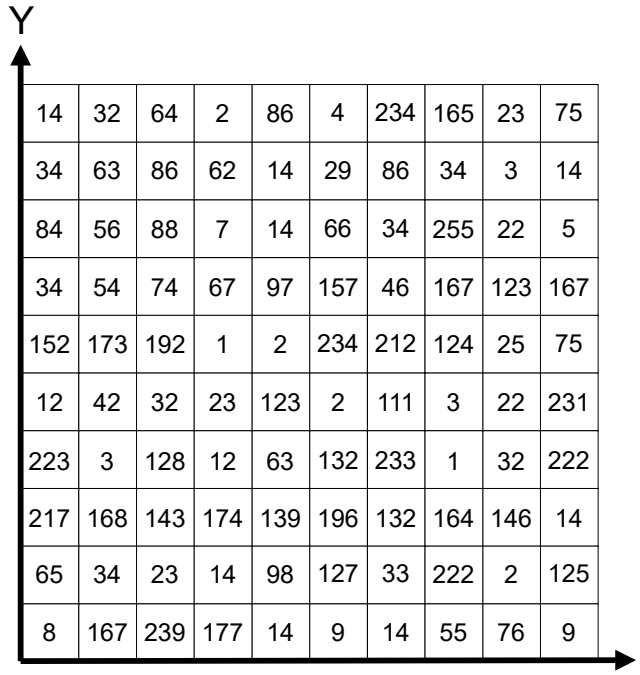
### 2.1.1 Imagens *raster* e vetoriais

A representação de imagens digitais pode ser feita de duas maneiras distintas: utilizando imagens *raster* ou imagens vetoriais. Esses dois formatos possuem diferentes usos e aplicações, permitindo que as imagens digitais sejam utilizadas em diferentes áreas do conhecimento, como *design*, publicidade, robótica e mecatrônica.

Imagens digitais do tipo *raster* são constituídas por *pixels*, sendo esses os menores elementos de representação gráfica. Usualmente, os *pixels* são organizados em um formato matricial, sendo atribuído a cada *pixel*, dois valores relacionados a sua posição (por exemplo,

coordenadas $x$ e $y$) e pelo menos um valor de intensidade (BURGER; BURGE, 2009), como se observa na Figura 2. As principais extensões desse tipo de imagem são: JPEG (*Joint Photographic Experts Group*), PNG (*Portable Network Graphics*), GIF (*Graphics Interchange Format*) e TIFF (*Tagged Image File Format*).

Figura 2 – Estrutura de uma imagem *raster*.



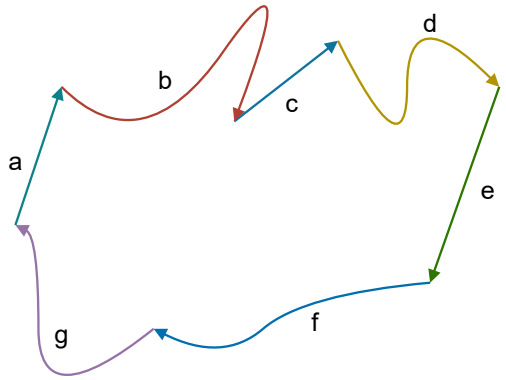

Fonte: elaborado pelo autor.

Alternativamente, pode-se observar que as imagens vetoriais são compostas por objetos geométricos baseados em equações matemáticas, e caracterizam-se pela utilização de vetores definidos por segmentos de curvas, os quais são limitados por pontos de controle, conforme ilustrado na Figura 3. Especificamente, tais imagens são rasterizadas (se tornam *raster*) quando precisam ser representadas em algum dispositivo físico, como um monitor de computador (BURGER; BURGE, 2009). A principal diferença entre imagens *raster* e vetorial é que a resolução da imagem vetorial não diminui ao aplicar-se operações de escalonamento (conhecidas informalmente como *zoom*). Essa característica tem origem no fato de que sempre que houver uma mudança de escala na imagem vetorial, as suas equações constituintes são recalculadas para em seguida serem exibidas. Os principais formatos de imagens vetoriais são: PDF (*Portable Document Format*), EPS (*Encapsulated PostScript*) e SVG (*Scalable Vector Graphics*).

### 2.1.2 Imagens binárias, em escala de cinza e coloridas

No âmbito das imagens *raster*, é possível classificá-las em três categorias distintas, isto é, imagem binária, escala de cinza ou colorida; cada uma com suas características

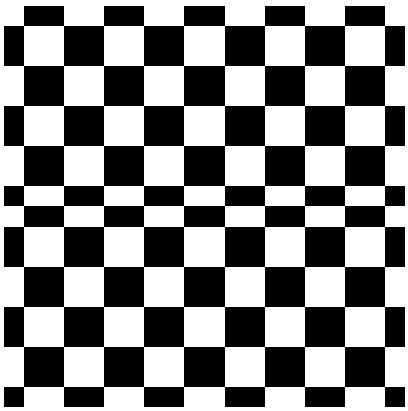
Figura 3 – Representação de uma imagem vetorial.




Fonte: elaborado pelo autor.

específicas e faixas de valores para representação dos *pixels*. Nas imagens binárias, cada *pixel* assume um entre dois valores, a saber: 0,0 ou 1,0 (se o tipo de dado utilizado para representação da imagem for ponto flutuante), ou ainda, 0 ou, 255 (se o tipo de dado utilizado for inteiro sem sinal de 8 *bits*, *uint8*). Especificamente, os valores 0,0 (ponto flutuante) e 0 (inteiro) representam o preto e $1,0$ ou 255 denotam o branco. Um exemplo básico desse tipo de imagem é observado na Figura 4.

Figura 4 – Imagem binária.

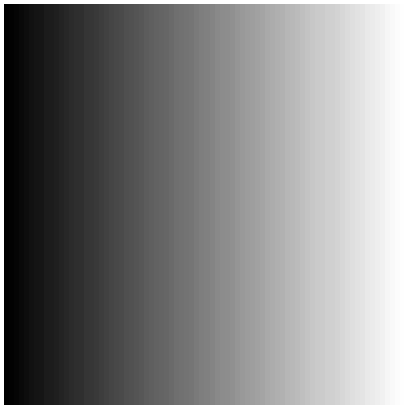


Fonte: elaborado pelo autor.

Em contrapartida, nas imagens em escala de cinza cada *pixel* pode assumir um valor em um dado intervalo, apresentando uma extensão na gama de valores disponíveis (em comparação com imagens binárias), proporcionando a representação de uma vasta diversidade de tons de cinza. Imagens em escala de cinza são observadas no mundo real pela reflexão de luzes monocromáticas, ou seja, que só possuem um comprimento de onda

(GONZALEZ; WOODS, 2009). Normalmente, os valores que podem ser assumidos por cada *pixel* variam de 0,0 até 1,0 (se o dado utilizado na representação da imagem digital for ponto flutuante), ou ainda, de 0 até 255 (para o caso de inteiro sem sinal de 8 bits), com os valores entre 0,0 e 1,0 (ou entre 0 e 255), representando tons de cinza que vão do preto absoluto (0,0 ou 0) até o branco absoluto (1,0 ou, 255). Um exemplo de imagem em escala de cinza é ilustrado na Figura 5.

Figura 5 – Imagem em escala de cinza.



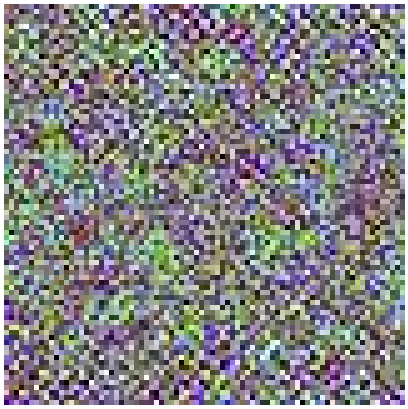
Fonte: elaborado pelo autor.

Por fim, nas imagens coloridas do tipo *raster*, a cor de cada *pixel* é representada por um conjunto de três valores, cujo significado depende do espaço de cor utilizado para representação das cores. Na Figura 6 é apresentado um exemplo de imagem colorida com os *pixels* possuindo cores aleatórias.

Em particular, um dos espaços de cores mais utilizados para representação de imagens coloridas é o RGB (*Red, Green and Blue*). Neste espaço, cada cor é representada como uma combinação das cores primárias vermelho (*Red*), verde (*Green*) e azul (*Blue*), com valores numéricos de cada tonalidade de cor variando de 0,0 até 1,0 (para dados do tipo ponto flutuante) ou de 0 até 255 (para inteiros sem sinal de 8 bits).

A percepção das cores nas imagens, na prática, é resultado da interação da luz policromática (semelhante àquela emitida pelo sol e composta por diversos comprimentos de onda) com as superfícies dos objetos, os quais podem refletir total ou parcialmente cada comprimento de onda (dependendo do tipo de material que constitui a superfície do objeto). Essas ondas refletidas, ao serem captadas pelos olhos humanos, excitam células do olho, denominadas cones, gerando os sinais elétricos que são, por sua vez, interpretados pelo cérebro como diferentes cores (GONZALEZ; WOODS, 2009).

Figura 6 – Imagem colorida.



Fonte: elaborado pelo autor.

## 2.2 PROCESSAMENTO DIGITAL DE IMAGENS

O processamento digital de imagens é um processo computacional que transforma uma ou mais imagens de entrada em uma imagem de saída (CORKE, 2011), visando, por exemplo, algum tipo de melhoramento na imagem, destaque de alguma característica específica, ou ainda detecção de um objeto de interesse na imagem. Dessa maneira, técnicas de processamento podem ser utilizadas em diversas áreas como: edição artística de imagens, visão computacional em robótica, astronomia e medicina.

Dentre as diversas técnicas de processamento digital de imagens, destacam-se: as operações monádicas, transformações geométricas e operações espaciais de filtragem, as quais são discutidas com mais detalhes nas seções seguintes.

### 2.2.1 Operações monádicas

Operações monádicas correspondem a um conjunto de algoritmos de processamento digital de imagens onde cada *pixel* de uma imagem de entrada $\mathbf{I}$ é processado por uma função $f(\cdot)$, sendo o resultado atribuído ao correspondente *pixel* na imagem de saída $\mathbf{O}$. Matematicamente, tem-se a expressão descrita na Equação (1), na qual $\mathbf{O}(u, v)$ e $\mathbf{I}(u, v)$ correspondem aos *pixels* na coordenada $(u, v)$ das imagens de saída e entrada, respectivamente.

$$\mathbf{O}(u, v) = f(\mathbf{I}(u, v)) \tag{1}$$

Especificamente, nas operações monádicas tem-se apenas uma única imagem de entrada e uma única imagem de saída (CORKE, 2011). Na sequência, são apresentadas e definidas as operações monádicas de alteração de brilho, alteração de contraste, obtenção de negativo, limiarização e posterização.

#### *2.2.1.1 Alteração de brilho*

A alteração de brilho é uma operação monádica que altera a intensidade de todos os *pixels* de uma imagem com base em uma constante $L$ (NIXON; AGUADO, 2002), conforme a Equação (2). Especificamente, essa operação altera o brilho total da imagem de forma linear, sendo muito empregada no tratamento de imagens que precisam ser clareadas (o que é obtido com $L > 0$) ou escurecidas ($L < 0$). Um exemplo de aplicação dessa operação pode ser observado na Figura 7a.

$$\mathbf{O}(u, v) = \mathbf{I}(u, v) + L \tag{2}$$

#### *2.2.1.2 Alteração de contraste*

A operação de alteração de contraste é muito similar à alteração de brilho, pois ela também altera o valor de todos os *pixels* de modo a aumentar ou diminuir o brilho total da imagem (NIXON; AGUADO, 2002). Entretanto, esse resultado é obtido ao multiplicar cada *pixel* da imagem de entrada por um escalar $K$, conforme a Equação (3). Dessa maneira, diferenciando-se da alteração de brilho, a alteração de contraste modifica a distância de intensidade entre os *pixels* mais claros e os mais escuros da imagem, como pode-se observar na Figura 7b.

$$\mathbf{O}(u, v) = K \cdot \mathbf{I}(u, v) \tag{3}$$

#### *2.2.1.3 Obtenção de negativo*

A operação monádica de obtenção de negativo em imagens digitais, definida na Equação (4), consiste essencialmente na substituição do valor de cada *pixel* em seu complementar, sendo este definido como o valor necessário para atingir o valor máximo do *pixel*. No contexto de dados do tipo *double*, o valor máximo é estabelecido em como $A = 1,0$, enquanto em dados *uint8*, comum em representações de imagens, tem-se $A = 255$ (BURGER; BURGE, 2009). Para imagens binárias, essa operação resulta na inversão completa dos valores dos *pixels*, onde todos os *pixels* brancos tornam-se pretos, e todos os *pixels* pretos tornam-se brancos. Na Figura 7c é apresentado o efeito prático de tal operação.

$$\mathbf{O}(u, v) = A - \mathbf{I}(u, v). \tag{4}$$

#### *2.2.1.4 Limiarização*

A limiarização é uma operação monádica usada para converter imagens em tons de cinza em imagens binárias, sendo amplamente utilizada em algoritmos de segmentação de imagens. Matematicamente, ela é definida conforme a Equação (5). Nessa equação, o valor de cada *pixel* da imagem de entrada $\mathbf{I}$ é comparado com um limiar $L$ (definido

Figura 7 – Operações monádicas de alteração de brilho, alteração de contraste e obtenção do negativo em imagens digitais.

(a) *Alteração de brilho.*

(b) Alteração de contraste.

(c) Obtenção de negativo.

Fonte: elaborado pelo autor.

pelo projetista), resultando na atribuição de valores binários, geralmente 0 (preto) ou 1 (branco), no *pixel* correspondente na imagem de saída. Conforme ilustrado na Figura 8a, ao se aplicar a operação de limiarização pode-se gerar uma imagem binária, na qual algumas regiões de interesse passam a possuir o mesmo valor de *pixel* (0 ou 1).

$$\mathbf{O}(u,v) = \begin{cases} 0, & se \quad \mathbf{I}(u,v) < L \\ 1, & se \quad \mathbf{I}(u,v) \geq L \end{cases} \tag{5}$$

#### *2.2.1.5 Posterização*

A operação monádica de posterização é empregada com o propósito de reduzir os níveis de cinza em uma imagem digital, resultando em um efeito visual que se assemelha à estilização da *pop art* (CORKE, 2011). Ela é fundamentada na Equação (6), a qual quantiza as intensidades dos *pixels* para um conjunto específico de níveis de cinza. Neste contexto, $\lfloor \cdot \rfloor$ simboliza a operação de arredondamento para o inteiro inferior mais próximo. A Figura 8b ilustra a operação, na qual a imagem original é submetida à posterização considerando $N = 5$, reduzindo assim a suavidade das transições e criando segmentos com somente 5 valores de cinza distintos na imagem.

$$\mathbf{O}(u,v) = \frac{1}{N} \cdot \lfloor \mathbf{I}(u,v) \cdot N \rfloor \tag{6}$$

Figura 8 – Operações monádicas de limiarização e posterização em imagens digitais.

(a) Limiarização. (b) Posterização.

Fonte: elaborado pelo autor.

### 2.2.2 Operações diádicas

As operações diádicas são procedimentos nos quais duas imagens de entrada $\mathbf{I}_1$ e $\mathbf{I}_2$ são processadas para gerar uma imagem de saída $\mathbf{O}$. Portanto, um requisito fundamental é que todas as três matrizes envolvidas, ou seja, as duas imagens de entrada e a imagem de saída, tenham a mesma dimensão. Nas operações diádicas, cada *pixel* da imagem de saída $\mathbf{O}(u, v)$ é determinado a partir do processamento (por uma função $f(\cdot, \cdot)$) dos *pixels* correspondentes, nas imagens de entrada, isto é, $\mathbf{I}_1(u, v)$ e $\mathbf{I}_2(u, v)$, conforme a Equação (7).

$$\mathbf{O}(u, v) = f\left(\mathbf{I}_1(u, v), \mathbf{I}_2(u, v)\right). \tag{7}$$

Especificamente, para a função de processamento $f(\cdot, \cdot)$ podem ser empregados diversos operadores aritméticos como os de adição, subtração e multiplicação elemento-a-elemento (CORKE, 2011). Tais operações diádicas desempenham um papel crucial em diversas aplicações de processamento de imagens, permitindo combinar informações de duas imagens distintas para obter os resultados desejados.

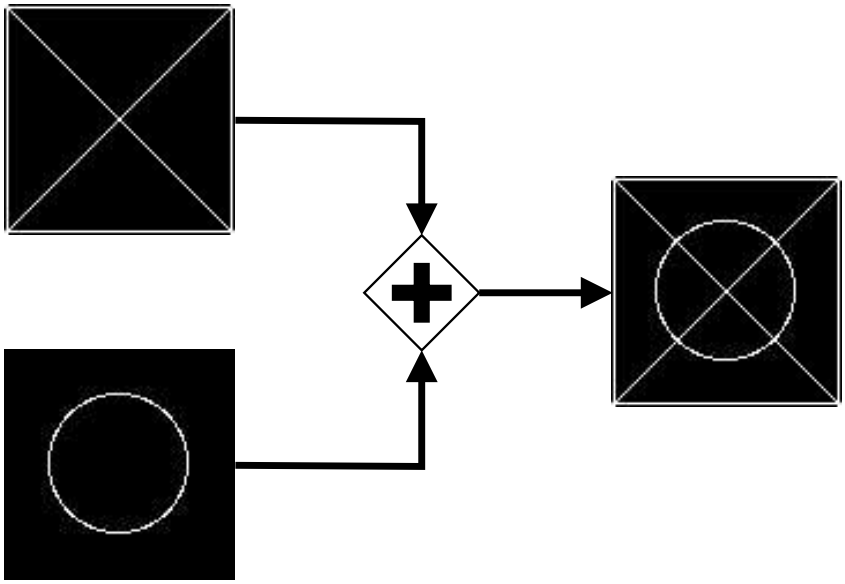
#### *2.2.2.1 Adição*

Nas operações de adição entre imagens digitais, o valor dos *pixels* correspondentes em cada imagem são somados, realizando uma operação de adição elemento-a-elemento entre as matrizes que representam as imagens (GONZALEZ; WOODS, 2009). Nesse contexto, é importante observar que, em imagens binárias, quando qualquer *pixel* branco é somado a um *pixel* preto, o resultado será um *pixel* branco, uma vez que o valor do *pixel* é saturado no valor máximo permitido pelo tipo de dado utilizado, como mostra a Figura 9. A operação de adição é fundamental em diversas aplicações de processamento de imagens, permitindo a combinação das informações de múltiplas imagens.

#### *2.2.2.2 Multiplicação*

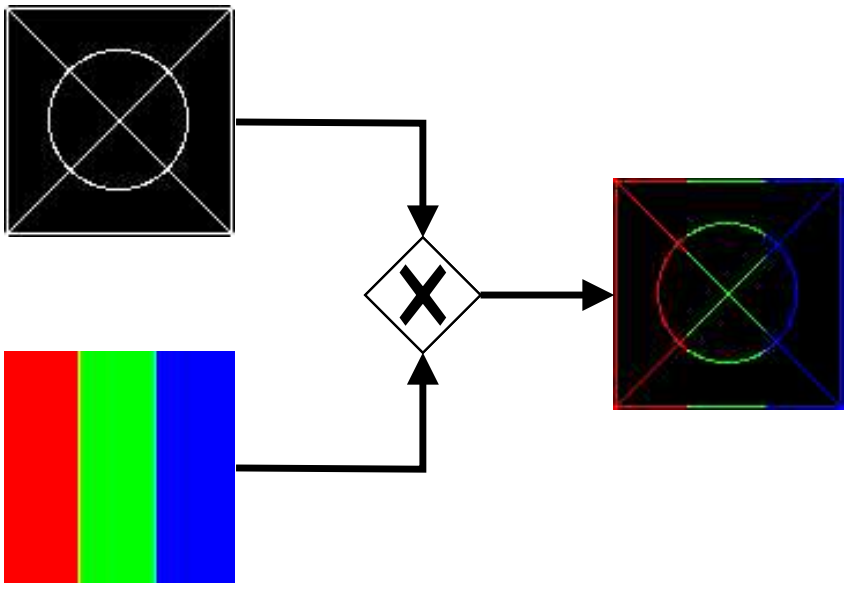
A operação diádica de multiplicação entre imagens difere da multiplicação tradicional entre matrizes, uma vez que é realizada a multiplicação elemento-a-elemento. Nesse processo, *pixels* correspondentes nas imagens de entrada (isto é, $\mathbf{I}_1(u, v)$ e $\mathbf{I}_2(u, v)$) são multiplicados, gerando um *pixel* $\mathbf{O}(u, v)$ na imagem de saída (GONZALEZ; WOODS, 2009). Quando uma das imagens de entrada é binária, a operação diádica de multiplicação tem o efeito de eliminar partes específicas da outra imagem de entrada, conforme ilustrado na Figura 10. Esse resultado se deve ao fato de que, na imagem binária, os *pixels* pretos, ao serem multiplicados pelos *pixels* correspondentes na outra imagem de entrada, resultam em *pixels* igualmente pretos. Por outro lado, os *pixels* brancos na imagem binária, quando multiplicados pelos *pixels* correspondentes na outra imagem de entrada, preservam a cor original do *pixel*.

Figura 9 – Adição de duas imagens digitais.



Fonte: elaborado pelo autor.

Figura 10 – Multiplicação entre duas imagens digitais.



Fonte: elaborado pelo autor.

### 2.2.3 Transformações geométricas

As transformações geométricas são operações que modificam a relação espacial entre *pixels* em uma imagem. Especificamente, nas transformações geométricas é realizado um mapeamento da posição dos *pixels* da imagem de entrada para outra posição na imagem de saída. Dentre as transformações geométricas existentes, as mais comuns são as Afins, que,

segundo Gonzalez e Woods (2009), fazem com que retas paralelas na imagem de entrada permaneçam paralelas na imagem de saída, mantendo a coerência das formas e estruturas. Os tipos mais comuns de transformações geométricas Afins incluem: escalamento, rotação, translação, espelhamento e cisalhamento. Em particular, o mapeamento das posições dos *pixels* nas transformações geométricas Afins é definido pela Equação (8), onde $u$ e $v$ são as coordenadas de coluna e linha do *pixel* na imagem de entrada e $u'$ e $v'$ representam a posição do *pixel* mapeado na imagem de saída. Nota-se que para a aplicação do mapeamento faz-se uso de coordenadas homogêneas, o que possibilita que todas as transformações afins sejam representadas por meio de multiplicações matriciais com vetores.

$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{12} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{8}$$

#### *2.2.3.1 Escalamento*

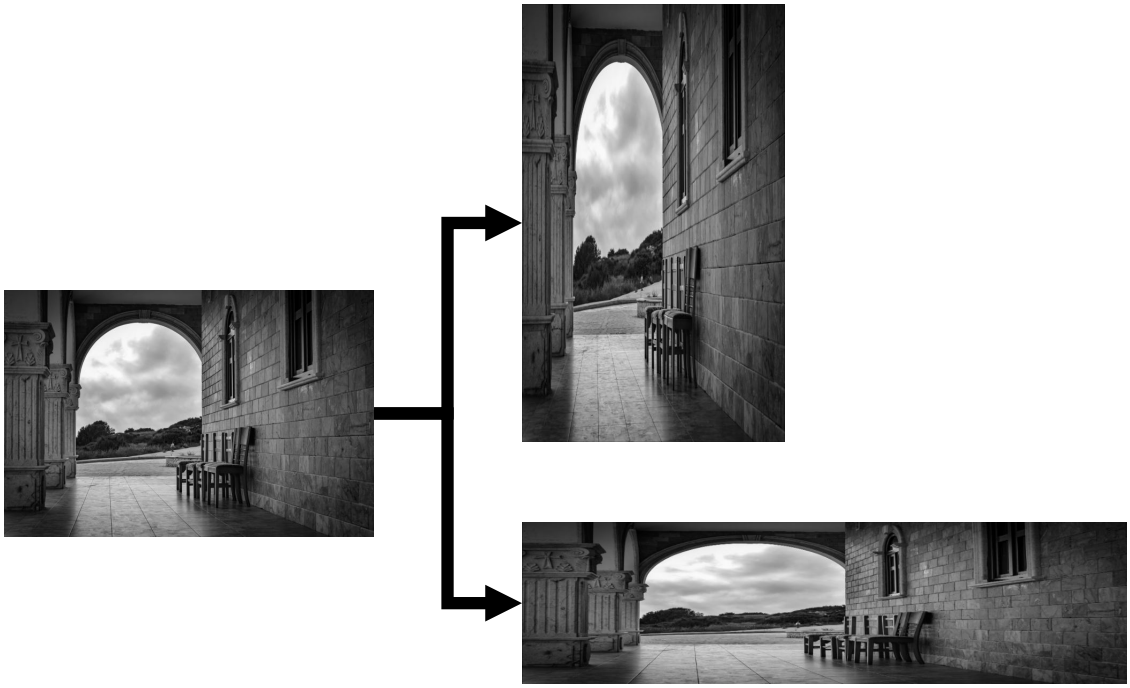
A transformação de escalamento é uma operação fundamental em processamento de imagem e visão computacional. Seu objetivo primordial é modificar as dimensões da imagem original (GONZALEZ; WOODS, 2009). Esta transformação é realizada por meio de uma matriz de transformação, utilizada na equação de mapeamento (9). Os parâmetros $a_{11}$ e $a_{22}$ dessa matriz desempenham papéis cruciais no processo, uma vez que $a_{11}$ está relacionado à alteração da largura da imagem, enquanto $a_{22}$ está associado à modificação de sua altura. O controle desses parâmetros permite ajustar o grau de escalamento desejado e, consequentemente, a proporção de aspecto da imagem resultante, como exibido pela Figura 11.

$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & 0 & 0 \\ 0 & a_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{9}$$

#### *2.2.3.2 Rotação*

A transformação de rotação visa reposicionar os *pixels* da imagem de entrada por um ângulo $\theta$ em torno do eixo Z (ortogonal ao plano da imagem) (GONZALEZ; WOODS, 2009). Essa operação é governada por uma matriz de transformação cujos parâmetros $a_{11}$, $a_{12}$, $a_{21}$ e $a_{22}$, que estão intrinsecamente relacionados com os valores de senos e cossenos do ângulo de rotação $\theta$, como expresso na Equação (10). Exemplos de rotação, tanto no sentido horário quanto anti-horário, podem ser observados na Figura 12.

Figura 11 – Transformação espacial geométrica de escalamento.



Fonte: elaborado pelo autor.

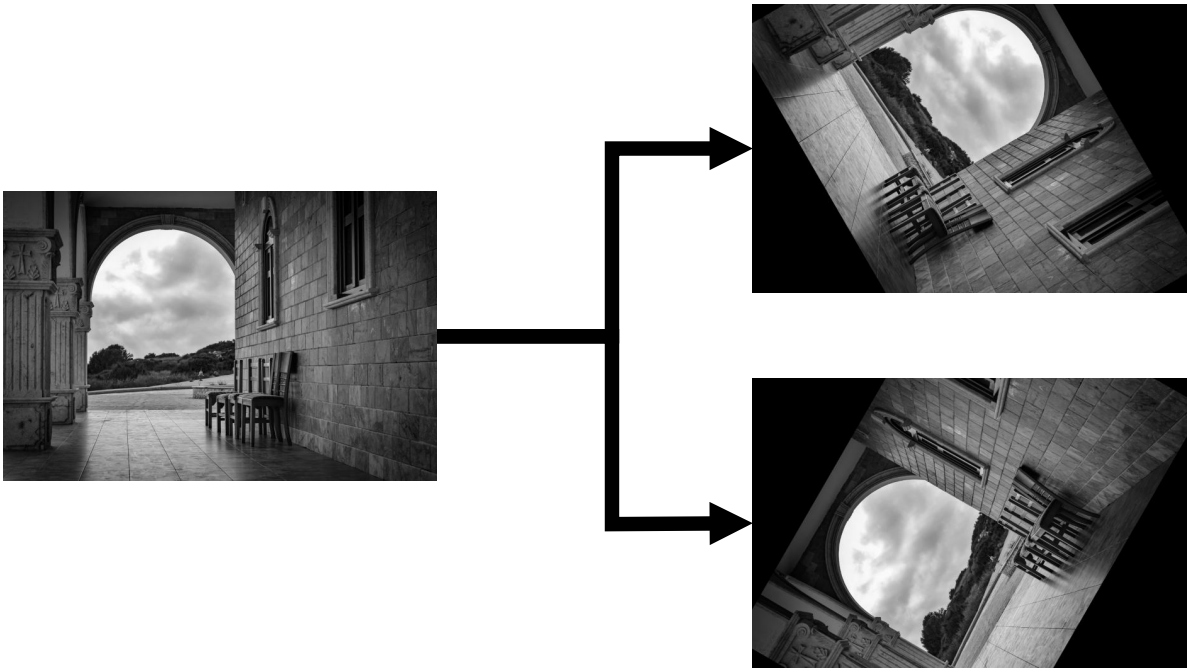
$$\begin{aligned}\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} &= \begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{21} & a_{22} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}\end{aligned} \tag{10}$$

#### *2.2.3.3 Translação*

Também é possível realizar a transformação geométrica de translação em imagens, cuja finalidade é deslocar a imagem ao longo dos eixos $u$ e $v$ (GONZALEZ; WOODS, 2009). Observa-se que os parâmetros $a_{13}$ e $a_{23}$, especificados na Equação (11), estão diretamente vinculados ao deslocamento horizontal e vertical, respectivamente. Exemplos destas translações podem ser observadas na Figura 13.

$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & a_{13} \\ 0 & 1 & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{11}$$

Figura 12 – Transformação espacial geométrica de rotação.



Fonte: elaborado pelo autor.

#### *2.2.3.4 Espelhamento*

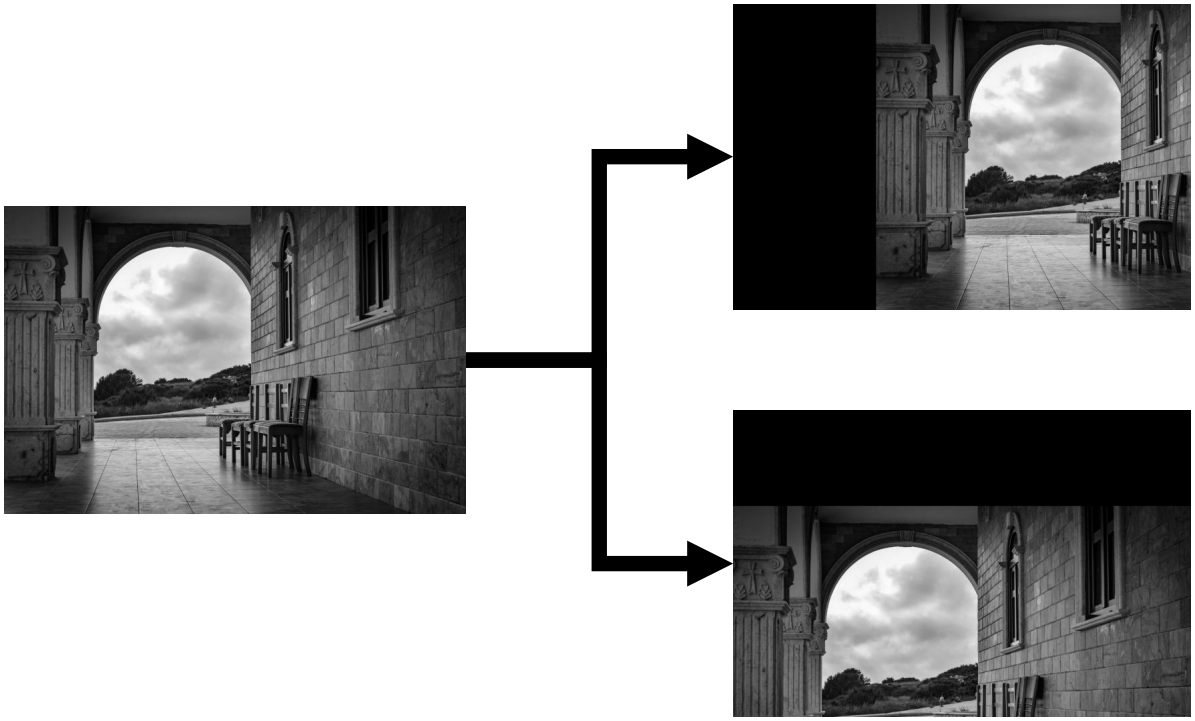
O espelhamento de imagens digitais é realizado para inverter a posição dos pixels da imagem de entrada, tanto em relação ao eixo horizontal quanto o vertical (GONZALEZ; WOODS, 2009). A matriz de transformação associada a essa operação é similar à da transformação de escalamento, conforme a Equação (9). No entanto, um espelhamento horizontal ou vertical ocorre quando aos coeficientes $a_{11}$ ou $a_{22}$ são atribuídos o valor de $-1$, respectivamente. Um exemplo desses espelhamentos é ilustrado na Figura 14.

#### *2.2.3.5 Cisalhamento*

A transformação geométrica de cisalhamento é empregada para deformar a imagem de entrada, tanto no eixo vertical como no horizontal (GONZALEZ; WOODS, 2009). Essa operação pode ser interpretada como o resultado da aplicação de uma tensão de cisalhamento ou tangencial nos eixos da imagem. Sua matriz de transformação é apresentada na Equação (12), na qual os parâmetros $a_{12}$ e $a_{21}$ estão diretamente associados respectivamente às distorções horizontais e verticais da imagem. Um exemplo da transformação de cisalhamento pode ser observado na Figura 15.

$$\begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & a_{12} & 0 \\ a_{21} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \tag{12}$$

Figura 13 – Transformação espacial geométrica de translação.



Fonte: elaborado pelo autor.

### 2.2.4 Filtragem 2D

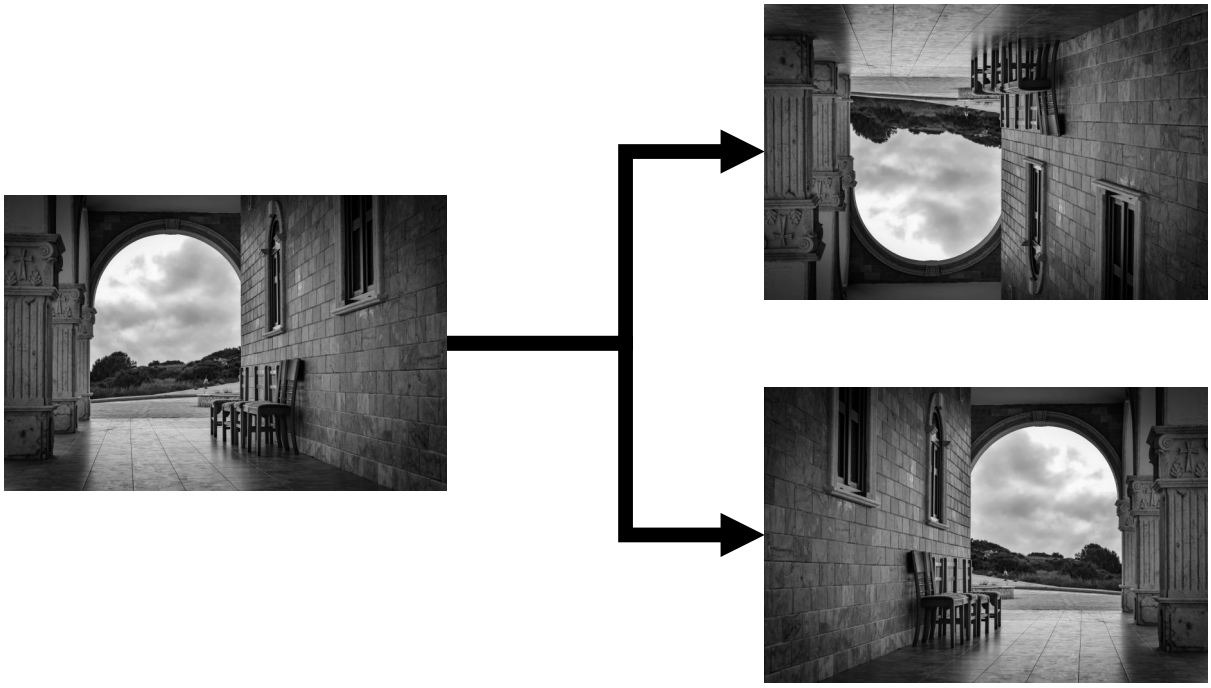
A filtragem 2D é um tipo de algoritmo que pertence à classe das operações espaciais. Em tais operações, cada *pixel* na imagem de saída $\mathbf{O}(u, v)$ é resultado do processamento dos *pixels* localizados em uma região em torno do *pixel* $\mathbf{I}(u, v)$ da imagem de entrada (CORKE, 2011). Especificamente, as operações espaciais ocorrem por meio de uma janela, que transita por todos os *pixels* da imagem, executando uma operação específica. Neste contexto, a operação aplicada determina a natureza do processamento, podendo ser do tipo linear ou não-linear (GONZALEZ; WOODS, 2009).

Especificamente, a filtragem 2D é uma operação espacial linear onde os *pixels* em uma janela são multiplicados pelos elementos correspondentes de uma matriz $\mathbf{K}$, denominada *Kernel*. Os produtos resultantes são então somados, gerando um valor atribuído ao *pixel* $\mathbf{O}(u, v)$ correspondente na imagem de saída. Matematicamente, tem-se a Equação (13), onde $\mathbf{K}(j, i)$ denota o elemento da coluna $j$ e linha $i$ do kernel, $\mathbf{I}(u + j, v + j)$ é o elemento da imagem de entrada localizado na posição $(j, i)$ da janela e $\mathbf{O}(u, v)$ caracteriza o elemento na imagem de saída que recebe o resultado da filtragem.

$$\mathbf{O}(u, v) = \sum_{i=-a}^{a} \sum_{j=-b}^{b} \mathbf{K}(i, j)\mathbf{I}(u + i, v + j) \tag{13}$$

O processo de filtragem 2D é exemplificado detalhadamente pelas Figuras 16 e 17,

Figura 14 – Transformação espacial geométrica de espelhamento.



Fonte: elaborado pelo autor.

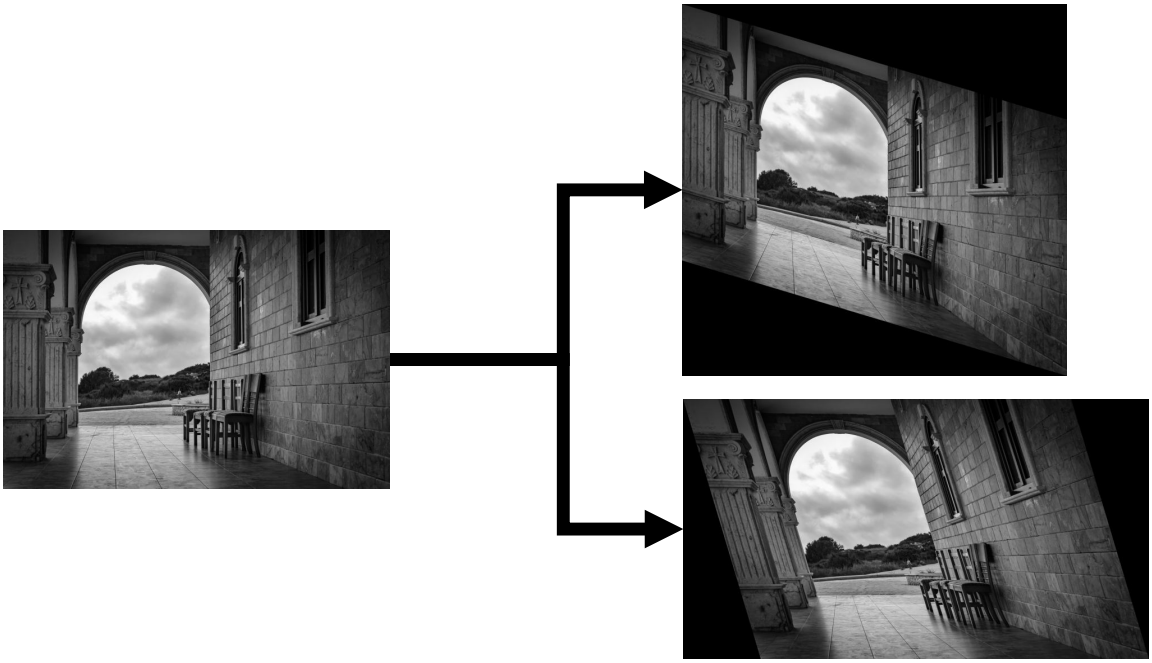
que mostram o processo de varredura do *Kernel* na imagem de entrada. Dependendo dos valores dos elementos do *Kernel*, diversos tipos de efeito podem ser obtidos, por exemplo, mitigação de ruídos e realce de bordas (CORKE, 2011).

Para a execução da filtragem, opta-se frequentemente por filtros com dimensões ímpares, proporcionando um *pixel* central $(u, v)$ claramente definido. A adoção de um *pixel* central se dá pela necessidade de um ponto de referência fixo quando o *kernel* $\mathbf{K}$ move-se pela imagem, permitindo que os valores dos *pixels* vizinhos sejam computados com mais precisão e simetria (GONZALEZ; WOODS, 2009). Ao assegurar que o *pixel* central do *kernel* transite por todos os *pixels* da imagem, frequentemente se adiciona uma borda de *pixels* em torno da imagem de entrada - processo conhecido como *padding* - com intensidade 0, objetivando a manutenção da dimensão original da imagem após o processo de filtragem (SZELISKI, 2010).

Em particular, na literatura da área de processamento digital de imagem, a operação de filtragem 2D é definida de duas formas distintas, a saber: correlação 2D e convolução 2D. A operação de correlação 2D é implementada conforme Equação (13), discutida anteriormente. Por outro lado, a operação de convolução 2D implementa A operação de filtragem 2D por meio da Equação (14).

$$\mathbf{O}(u, v) = \sum_{i=-a}^{a} \sum_{j=-b}^{b} \mathbf{K}(i, j)\mathbf{I}(u - i, v - j). \tag{14}$$

Figura 15 – Transformação espacial geométrica de cisalhamento.



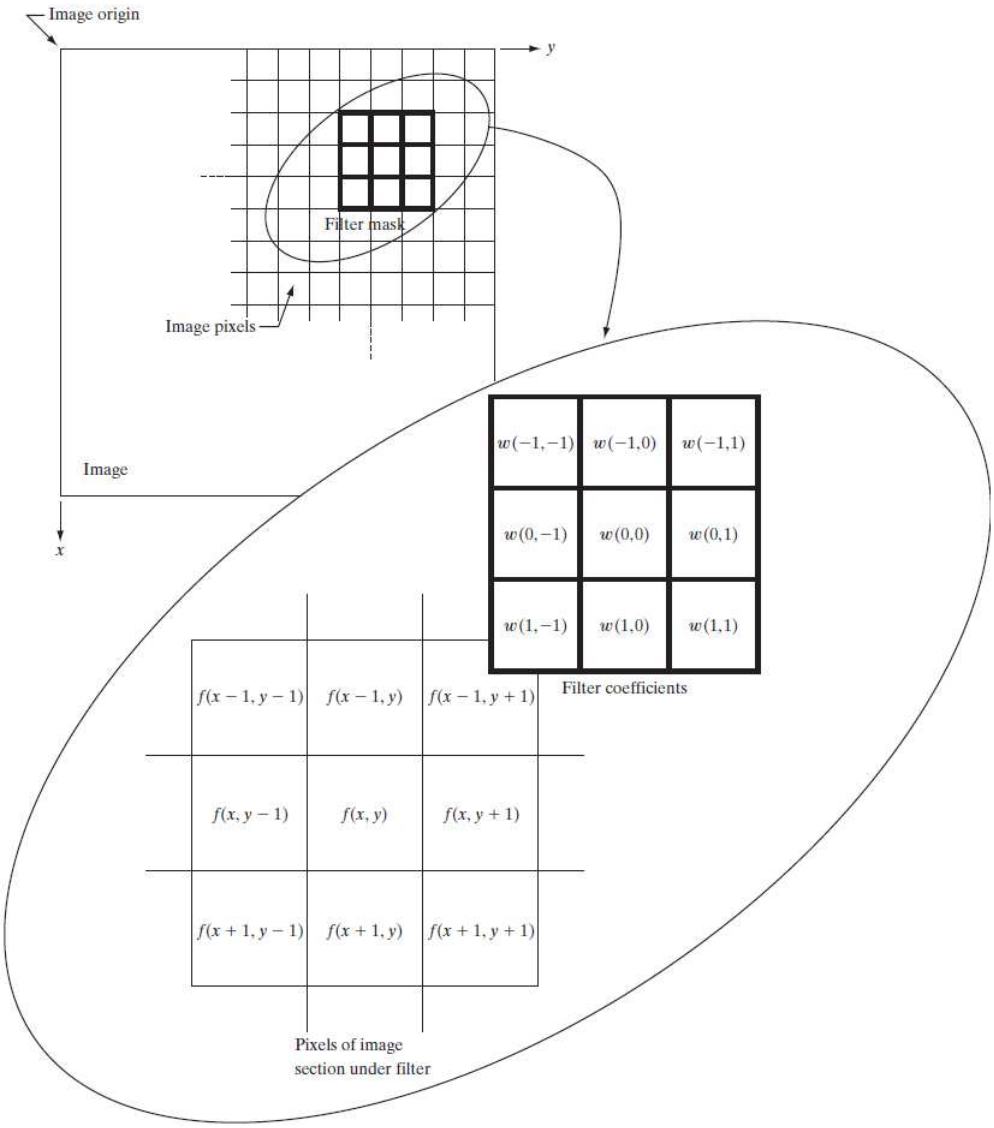
Fonte: elaborado pelo autor.

Observe que as operações de correção 2D e convolução 2D são bastantes semelhantes, se diferenciando apenas pelo sinal dos índices $i$ e $j$ na indexação dos elementos da imagem $\mathbf{I}$ na janela de processamento. Matematicamente, a convolução 2D também pode ser expressa como na Equação (15).

$$\mathbf{O}(u,v) = \sum_{p=-a}^{a} \sum_{q=-b}^{b} \mathbf{K}(-p,-q)\mathbf{I}(u+p, v+q). \tag{15}$$

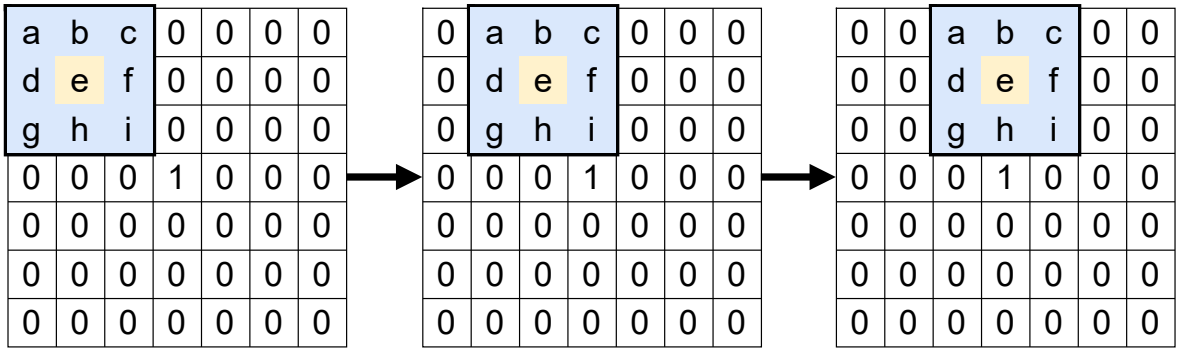
Nesse caso, troca-se apenas os sinais dos índices de indexação dos elementos do kernel $\mathbf{K}$, O que permite implementar a operação de convolução 2D da mesma forma que a correção 2D, bastando somente espelhar os elementos do kernel em relação aos eixos horizontal e vertical.

Figura 16 – Filtragem 2D em imagens digitais.




Fonte: (GONZALEZ; WOODS, 2009).

Figura 17 – Varredura do *kernel* na filtragem 2D.




Fonte: elaborado pelo autor.

## 2.3 TRANSFERÊNCIA DE CALOR

Conforme mencionado Capítulo 1, neste trabalho é implementado um sistema para identificação de comportamentos de agrupamentos de pessoas a partir da análise de mapas de calor que indicam a posição e a movimentação dos indivíduos na cena analisada. Especificamente, o mapa de calor gerado é baseado em conceitos relacionados com difusão e decaimento de calor, os quais são abordados nesta seção.

Para a elucidação de determinados fenômenos relativos à transferência de calor, torna-se necessário compreender a essência do conceito de calor. Especificamente, calor é definido como o processo de transferência de energia térmica entre dois sistemas, ocasionado pela existência de uma diferença de temperatura entre eles. Nota-se que, em termos precisos, um corpo não retém calor; o que ele possui é energia térmica. Assim, o calor somente é mensurado quando se verifica a transferência dessa energia térmica através da fronteira de um sistema (BORGNAKKE; SONNTAG, 2013).

Adicionalmente, é importante salientar que a transferência de calor sempre se dá do sistema com temperatura mais elevada para aquele com temperatura inferior, resultando, ao término do processo, em uma equalização das temperaturas entre ambos os sistemas. No âmbito do Sistema Internacional de Unidades, o calor é quantificado em *joules*, do mesmo modo que a energia. Outro ponto importante é que o calor pode ser transferido de três maneiras diferentes: por condução, convecção ou irradiação. Ademais, nos corpos sólidos, observa-se a capacidade do calor em difundir-se através das moléculas, bem como a possibilidade de seu decaimento ao desativar a sua fonte (sistema de maior temperatura).

### 2.3.1 Modos de transferência de calor

Segundo a termodinâmica, as moléculas de um corpo apresentam energia nas formas translacional, rotacional e vibracional. Estes tipos de energia conseguem ser transferidos de uma molécula a outra por meio de colisões, partindo do elemento com maior energia para aquele com menor. Este processo, conhecido como condução, é um dos modos primários de transferência de calor, cuja taxa de transferência pode ser descrita pela Equação (16). Esta equação correlaciona a quantidade de calor transferido com a condutividade térmica $k$ do material, a área $A$ pela qual o calor se transmite e o gradiente de temperatura entre as regiões em questão (BORGNAKKE; SONNTAG, 2013).

$$\dot{Q} = -kA\frac{dT}{dx} \tag{16}$$

Por sua vez, o fenômeno de convecção é caracterizado pela transferência de calor entre um fluido em movimento e uma superfície sólida. A intensidade da transferência de calor via convecção está diretamente relacionada à diferença de temperatura entre o fluido e a superfície em contato, conforme representado pela Equação (17). Essa equação

incorpora o coeficiente de convecção $h$, a área de contato $A$ e a diferença de temperatura $\Delta T$ entre o fluido e a superfície sólida (BORGNAKKE; SONNTAG, 2013).

$$\dot{Q} = Ah\Delta T \tag{17}$$

O terceiro mecanismo, a radiação, distingue-se por envolver a transmissão de energia por meio de ondas eletromagnéticas. Ocorrendo tanto no vácuo quanto em meios materiais, este processo permite a emissão e absorção de energia sem a necessidade de um meio físico contínuo. A Equação (18) ilustra os parâmetros que afetam a taxa de transferência de calor por radiação, englobando a emissividade $\varepsilon$ do material, a constante de Stefan-Boltzmann $\sigma$, a área superficial $A$ e a quarta potência da temperatura da superfície $T_s$ (BORGNAKKE; SONNTAG, 2013).

$$\dot{Q} = \varepsilon\sigma A T_s^4 \tag{18}$$

### 2.3.2 Difusão térmica

Para explicar o conceito de difusão térmica por condução, considere primeiramente um ponto arbitrário, denominado $Q$, situado em um corpo sujeito à variação térmica. A distância deste ponto a outro ponto de interesse $P$ é denotada por $r$. Assuma que o ponto $Q$ possui uma temperatura inicial $\theta$, a qual serve de referência para o processo de difusão. Ao observar a dinâmica térmica no ponto $P$ após um intervalo de tempo $t$, assume-se que sua temperatura resultante é influenciada por uma junção das temperaturas de porções de matéria de todas as regiões do corpo. Essa mistura considera, para cada porção, uma quantidade de temperatura transferida proporcional a $e^{-r^2/(4kt)}$, em que $k$ representa a difusividade térmica do material, implicando que regiões próximas ao ponto $P$ exercem maior influência na temperatura resultante do que aquelas mais afastadas (MAXWELL, 1904).

Dessa forma, a temperatura no ponto $P$, após o tempo $t$, pode ser descrita como uma média ponderada das temperaturas iniciais presentes no corpo. O fator de ponderação é calculado pela função exponencial mencionada, que reflete a relevância de cada ponto em função de sua distância até $P$ e do tempo considerado. O somatório desta distribuição sobre o volume do corpo determina a temperatura no ponto $P$. Esse modelo matemático capta o princípio físico de que a propagação do calor é mais significativa a partir de pontos mais próximos a $P$, diminuindo à medida que a distância aumenta, uma característica intrínseca dos processos de difusão térmica em meios sólidos (MAXWELL, 1904).

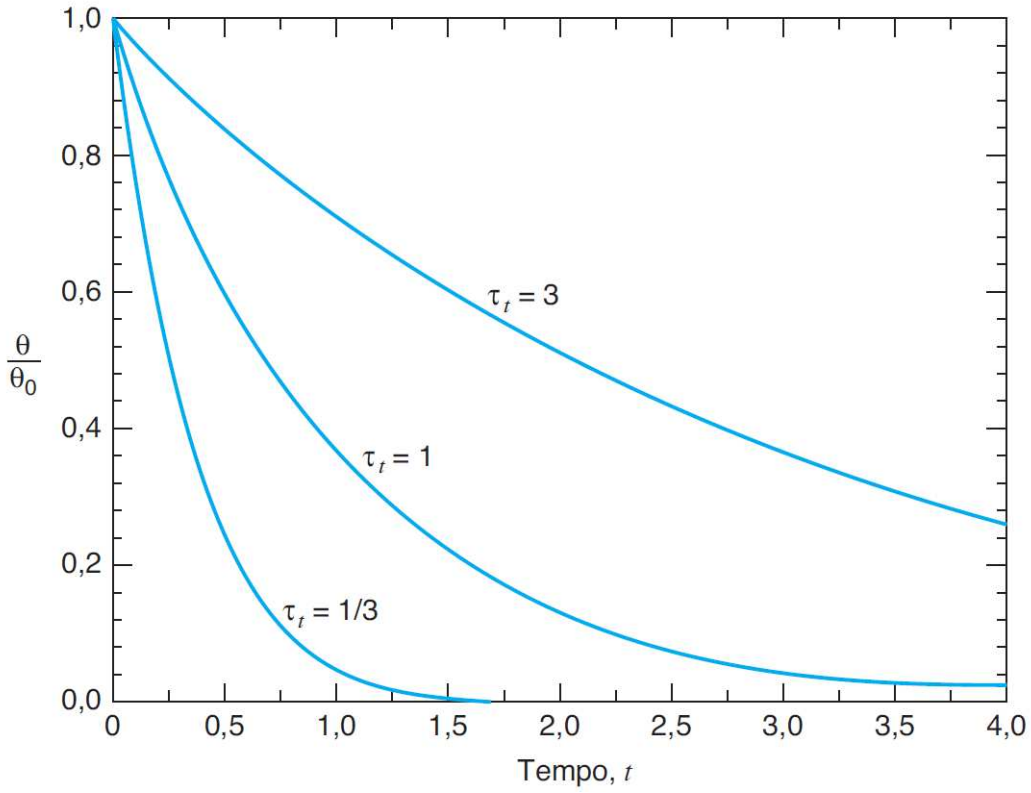
### 2.3.3 Decaimento temporal

A análise do decaimento térmico em um corpo imerso em um fluido a temperatura constante $T_\infty$ é crucial para entender os mecanismos de resfriamento ou aquecimento.

Quando um corpo de massa $m$ e calor específico a volume constante $C_v$ é submerso em um fluido, a diferença de temperatura entre o corpo e o fluido $(T - T_\infty)$ decresce exponencialmente com o tempo. Esse fenômeno é descrito pela Lei de Resfriamento de Newton dada pela Equação (19), na qual $C_q$ é o coeficiente de transferência de calor e $A$ é a área superficial do corpo. A solução dessa equação diferencial fornece a função que descreve a variação da temperatura do corpo (BORGNAKKE; SONNTAG, 2013).

$$mC_v\frac{dT}{dt} = -C_qA(T - T_\infty), \tag{19}$$

Figura 18 – Decaimento térmico exponencial.




Fonte: (BORGNAKKE; SONNTAG, 2013).

Em particular, a resolução dessa equação diferencial de primeira ordem retorna a Equação (20), na qual $T_0$ é a temperatura inicial do corpo e $\tau_t$ é a constante de tempo térmica definida pela Equação (21). A constante de tempo térmica $\tau_t$ determina a rapidez com que a temperatura do corpo se aproxima da temperatura do fluido envolvente. Borgnakke e Sonntag (2013) evidenciam que sistemas com uma menor constante de tempo térmica, como sensores termopares, respondem rapidamente às variações de temperatura, enquanto uma maior constante de tempo é desejável para materiais de isolamento utilizados em aplicações que visam o conforto térmico em edificações, prolongando o período de decaimento térmico e minimizando as trocas de calor não desejadas com o ambiente. Na

Figura 18 são apresentadas três curvas obtidas a partir da Equação (20) para diferentes parâmetros.

$$T(t) = T_\infty + (T_0 - T_\infty)e^{-\frac{t}{\tau_t}} \tag{20}$$

$$\tau_t = \frac{mC_v}{C_q A} \tag{21}$$

## 2.4 *MACHINE LEARNING*

O *Machine Learning* (aprendizado de máquina) é uma sub-área da IA (Inteligência Artificial). No geral, considera-se que um algoritmo de *Machine Learning* aprende ao melhorar seu desempenho em tarefas específicas através da experiência, conforme medido por uma métrica pré-estabelecida (MITCHELL, 1997). Tais algoritmos de aprendizado de máquina podem ser classificados em: supervisionado, não supervisionado e aprendizado por reforço. Dentre as técnicas de aprendizado de máquina, pode-se citar as árvores de decisão, a regressão linear, a regressão logística e as redes neurais artificiais.

### 2.4.1 Redes neurais artificiais

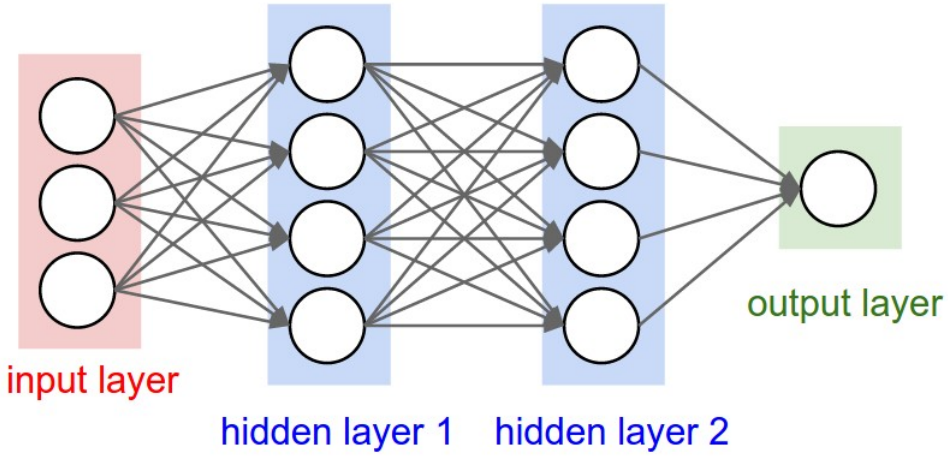
Redes neurais artificiais constituem uma classe de modelos do aprendizado de máquina intrinsecamente inspirados na complexa rede de neurônios biológicos. Atualmente existem diversas topologias de redes neurais, das mais simples e básicas (por exemplo, redes neurais *feedfoward* densamente conectadas) até as redes neurais mais complexas (como redes neurais convolucionais e recorrentes).

Nas redes neurais *feedfoward* densamente conectadas, os elementos básicos são os neurônios, também conhecidos como *perceptrons*. Essas redes são organizadas em camadas, possuindo no mínimo três, isto é: uma camada de entrada, pelo menos uma camada oculta e uma camada de saída. Cada camada é constituída por um dado número de neurônios (definido pelo projetista), sendo organizados em paralelo (de modo a processarem o mesmo conjunto de dados), como pode-se observar na Figura 19.

Em particular, um neurônio artificial, unidade básica de processamento dessas redes, apresenta-se como uma abstração computacional do neurônio biológico, em que sinapses são modeladas por conexões ponderadas. Especificamente, cada neurônio é constituído por um conjunto de pesos $w_i$ e por um *bias* $b$, com o processamento do neurônio ocorrendo da seguinte forma: cada entrada $a_i$ (escalar) fornecida ao neurônio é multiplicada por um peso $w_i$, com os resultados $w_i a_i$ somados entre si, com o esse último resultado sendo somado também com o *bias* $b$, conforme a Equação (22).

$$z = b + \sum_{i=0}^{N-1} w_i a_i. \tag{22}$$

Figura 19 – Rede neural *feedforward* densamente conectada.




Fonte: (CS231N, 2023).

Em particular, o processo de treinamento da rede neural visa obter os valores dos pesos $w_i$ e do *bias* $b$ de cada neurônio que reduzam o valor de uma dada função custo. Note que, enquanto o neurônio biológico ajusta a força sináptica em resposta a estímulos externos, o artificial altera seus pesos sinápticos durante o treinamento, visando minimizar discrepâncias entre as saídas previstas e os resultados desejados. Dessa forma, a correlação entre a ponderação sináptica e a eficácia da transmissão de sinais no domínio biológico é refletida na manipulação dos pesos em redes neurais artificiais, onde o ajuste desses pesos é fundamental para a capacidade de generalização do modelo (AGGARWAL, 2018).

### 2.4.2 Treinamento e descida do gradiente

Especificamente, no processo de treinamento de uma rede neural os valores dos pesos e *bias* são atualizados mediante um processo de otimização iterativo que visa a redução do valor de uma função custo $J$. Dentre os algoritmos de otimização mais utilizados para treinamento, destaca-se o método da descida do gradiente. Neste método a atualização de cada peso $w_i$ é realizado por meio da Equação (23).

$$w_i = w_i - \mu \frac{\partial J}{\partial w_i} \tag{23}$$

Nessa expressão $\mu$ é um escalar usualmente denominado taxa de aprendizado ou LR (*Learning Rate*) e $\partial J/\partial w_i$ representa o gradiente da função custo em relação ao parâmetro $w_i$. Em particular, o gradiente $\partial J/\partial w_i$ indica a influência do peso $w_i$ no valor da função custo $J$, por exemplo, $\partial J/\partial w_i > 0$ significa que um aumento no valor de $w_i$ acarreta um aumento da função custo. Por outro lado, se $\partial J/\partial w_i < 0$, tem-se que uma redução no valor de $w_i$ leva a um aumento da função custo. Tendo em vista que o objetivo do treinamento

é reduzir o valor de $J$, então o valor de $w_i$ deve ser atualizado (a cada iteração do processo e treinamento) com um valor proporcional ao negativo do gradiente, ou seja, $-\mu\partial J/\partial w_i$.

Observe que para se computar a Equação (23) é necessário conhecer o gradiente $\partial J/\partial w_i$. Em geral, tal gradiente é estimado (para um dado conjunto de dados de treinamento) através do algoritmo conhecido como *backpropagation*. O *backpropagation*, um algoritmo amplamente adotado para o cálculo das derivadas relativas ao gradiente, destaca-se por sua eficácia, empregando o princípio da regra da cadeia para avaliar as derivadas parciais da função de custo total. Esse cálculo se inicia na saída da última camada, prosseguindo regressivamente até a primeira camada, resultando em uma aproximação computacional do gradiente total (MACKAY, 2005).

A qualidade e diversidade de um conjunto de dados de treinamento são fundamentais para o sucesso de modelos de aprendizado de máquina, particularmente em tarefas de classificação. Um conjunto de dados bem elaborado deve abranger uma ampla gama de características dos objetos a serem classificados, garantindo que o modelo aprenda a partir de exemplos variados. Geralmente, o conjunto de dados total é dividido em três partes distintas: treinamento, validação e teste. O *dataset* de treinamento é utilizado para o aprendizado direto do modelo, onde este ajusta seus parâmetros internos para mapear as entradas às saídas desejadas. O *dataset* de validação, por outro lado, é empregado para avaliar o desempenho do modelo após cada época de treinamento, permitindo ajustes finos nos parâmetros do modelo para melhorar sua eficácia. Finalmente, o *dataset* de teste consiste em dados que não foram usados durante a fase de treinamento. Este conjunto é crucial para avaliar o desempenho final do modelo, fornecendo uma medida imparcial de como o modelo provavelmente se comporta ao ser exposto a novos dados, garantindo assim que o modelo seja robusto e confiável em situações reais (AGGARWAL, 2018).

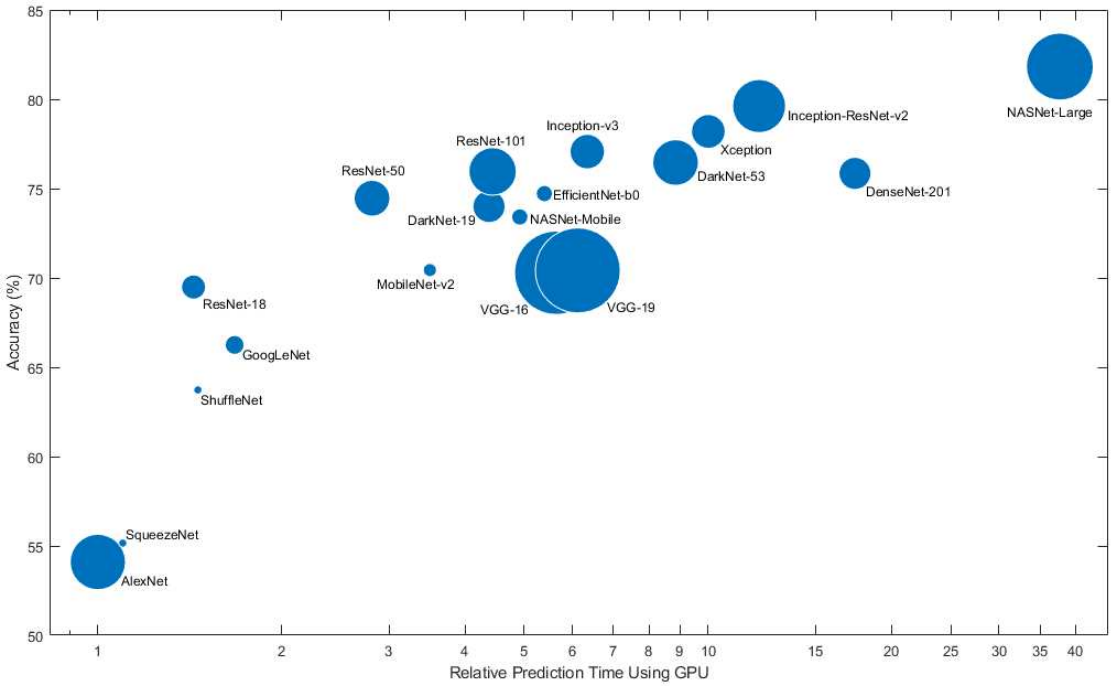
No processo de treinamento de redes neurais, os hiperparâmetros desempenham um papel crucial no ajuste do modelo, porque eles influenciam o treinamento da rede na totalidade. Um hiperparâmetro fundamental é o LR, responsável pela velocidade de ajuste dos pesos e *bias* da rede. Com um LR reduzido, o algoritmo pode levar mais tempo para encontrar a solução ótima. Em contrapartida, um LR elevado pode causar oscilações excessivas em torno da solução ótima, podendo até levar à instabilidade do modelo. Frequentemente, adota-se um LR adaptável, sendo necessário apenas configurar seu valor inicial (AGGARWAL, 2018). Outro hiperparâmetro crucial é o *minibatch size*, referente às amostras extraídas do *dataset* de treinamento para estimar o valor do gradiente (GOODFELLOW; BENGIO; COURVILLE, 2016). A escolha adequada destes parâmetros é vital para o desempenho eficaz do modelo, equilibrando a precisão e a eficiência do processo de aprendizado.

### 2.4.3 Redes neurais convolucionais

As redes neurais convolucionais (CNNs, do inglês *Convolutional Neural Networks*) são caracterizadas por possuir pelo menos uma camada em que é realizada a operação de convolução (discutida na Seção 2.2.4). Nesse caso, o processo de treinamento visa obter os valores dos elementos que compõem os *kernels* utilizados no processo de convolução. As redes neurais convolucionais são bastante apropriadas para processamento de dados de entrada que são por sua natureza organizados em matrizes ou grades, como, por exemplo, imagens digitais (GOODFELLOW; BENGIO; COURVILLE, 2016).

Além da operação de convolução, as CNNs também se valem de outra técnica de processamento conhecida como *pooling*. Esta operação segmenta a imagem em diversos setores e seleciona um valor representativo para cada um deles. Ademais, quando o valor máximo é escolhido dentro de cada setor, a operação é denominada *max pooling*, muito utilizada na prática devido à sua eficiência em tornar a representação da imagem aproximadamente invariante a pequenas translações (GOODFELLOW; BENGIO; COURVILLE, 2016). A arquitetura das CNNs pode variar em complexidade e topologia, adaptando-se ao número de camadas conforme a aplicação específica. Competições como a ILSVRC (*Large Scale Visual Recognition Challenge*), que utilizam o conjunto de dados ImageNet, impulsionam o desenvolvimento de diversas topologias inovadoras de CNNs, com destaque para *AlexNet*, *SqueezeNet*, *GoogLeNet* e *ResNet*. A Figura 20 apresenta a relação entre precisão e o custo computacional dessas arquiteturas.

Figura 20 – Redes pré-treinadas no *dataset ImageNet*.

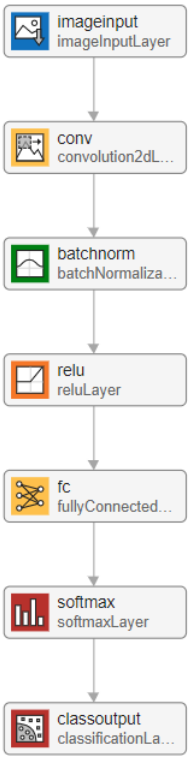



Fonte: (MATHWORKS, 2023a).

A classificação de camadas em redes neurais, conforme definido na Deep Learning Toolbox do MATLAB, organiza-as em categorias distintas. As Camadas de Entrada, em azul (Figura 21), estabelecem a forma e tipo dos dados de entrada. As Camadas de Convolução e Totalmente Conectadas, em amarelo, são essenciais para o aprendizado de características, enquanto as Camadas de Ativação, em laranja, introduzem não-linearidades necessárias para o bom funcionamento dos modelos.

As Camadas de Normalização, em verde, otimizam o treinamento, e as Camadas de Redimensionamento, em rosa, ajustam as dimensões dos dados. Em seguida, as Camadas de *Pooling* e *Unpooling*, em roxo, concentram-se na redução da dimensionalidade das amostras. Já as Camadas de Combinação, em amarelo-escuro, combinam informações de múltiplas fontes. Finalmente, as Camadas de Saída, em vermelho, definem a maneira como a rede apresenta seus resultados, em tarefas de classificação ou regressão.

Figura 21 – Tipos de camadas em uma rede neural convolucional.



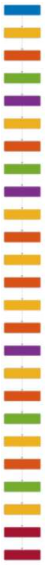

Fonte: (MATHWORKS, 2023b).

### *2.4.3.1 AlexNet*

A *AlexNet*, uma arquitetura inovadora de rede neural convolucional, foi desenvolvida por Krizhevsky, Sutskever e Hinton (2012) e apresentada na competição ILSVRC de 2012. Esta rede, que possui somente 25 camadas, conforme ilustrado na Figura 22a, representou um avanço significativo na área de visão computacional, não apenas pelo seu desempenho

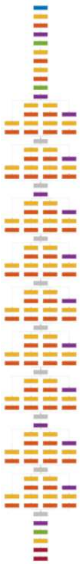
superior às demais concorrentes, mas também pela sua estrutura compacta. A arquitetura da *AlexNet* é notável por seu arranjo inovador, que inclui múltiplas camadas de convolução intercaladas com camadas de *max pooling* e normalização. A rede começa processando imagens de entrada com dimensões de $224 \times 224 \times 3$ *pixels*, utilizando um conjunto de 96 filtros de tamanho $11 \times 11 \times 3$ na primeira camada convolucional. As camadas subsequentes da rede empregam filtros com dimensões menores e um número maior de mapas de características, aprofundando progressivamente a análise das características visuais na imagem (AGGARWAL, 2018).

Figura 22 – Arquiteturas *AlexNet*, *SqueezeNet* e *GoogLeNet*.



(a) *AlexNet*.

(b) *SqueezeNet*.



(c) *GoogLenet*.

Fonte: elaborado pelo autor.

O notável desempenho da *AlexNet* na ILSVRC de 2012 pode ser atribuído à sua

taxa de erro top-5, de aproximadamente $15,4\%$. O erro top-5 é uma métrica que considera uma previsão como correta se a categoria real da imagem estiver entre as cinco principais categorias identificadas pelo modelo. Assim, um erro top-5 ocorre quando a categoria verdadeira não está entre as cinco principais previsões do modelo. Este índice de $15,4\%$ representou uma melhoria substancial em relação às abordagens anteriores, que exibiam taxas de erro superiores a 25%. Com esse desempenho, a *AlexNet* não só se destacou na competição por uma margem considerável, mas também estabeleceu novos padrões para a arquitetura de redes neurais convolucionais em aplicações de visão computacional. Além disso, as características extraídas de suas camadas intermediárias mostraram-se valiosas para uma variedade de outras tarefas de aprendizado de máquina (AGGARWAL, 2018).

#### *2.4.3.2 SqueezeNet*

A *SqueezeNet*, desenvolvida por Iandola *et al.* (2016), representa um avanço significativo na otimização de redes neurais convolucionais. Inspirada na *AlexNet*, a *SqueezeNet* foi projetada para manter um desempenho comparável em termos de precisão, mas com uma redução substancial no número de parâmetros, apesar de possuir 52 camadas (Figura 22b). Esta característica torna a *SqueezeNet* notavelmente eficiente em termos de armazenamento e processamento, facilitando sua implementação em plataformas com recursos limitados, como FPGA (*Field-Programmable Gate Array*) e veículos autônomos. Além disso, a *SqueezeNet* aceita imagens de entrada com dimensões de $227 \times 227 \times 3$ *pixels*, um padrão comum em redes pré-treinadas (MATHWORKS, 2023a). A combinação de alta precisão com uma estrutura compacta e econômica torna a *SqueezeNet* uma escolha atraente para uma grande variedade de aplicações em visão computacional e aprendizado profundo para sistemas embarcados (IANDOLA *et al.*, 2016).

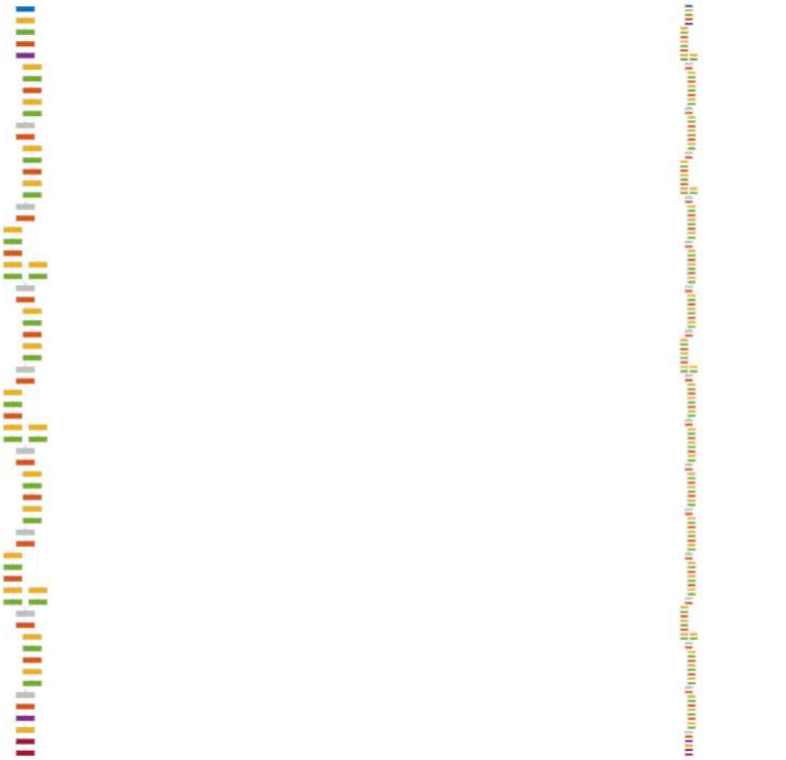
#### *2.4.3.3 GoogLeNet*

Desenvolvida por Szegedy *et al.* (2015) para competir na ILSVRC de 2014, a *GoogLeNet* introduziu o inovador conceito de arquitetura *inception*, na qual uma rede neural fica dentro de outra. Esta arquitetura é composta por uma série de módulos *inception*, apresentados na Figura 22c, totalizando 63 camadas. Tais módulos permitem ao modelo processar informações em diferentes níveis de detalhe, oferecendo a flexibilidade de capturar características visuais com variados tamanhos de filtros, como $1 \times 1$, $3 \times 3$ e $5 \times 5$. A *GoogLeNet* se destaca por seu design eficiente que reduz a necessidade de inúmeros parâmetros, ao mesmo tempo, em que mantém uma alta precisão na classificação de imagens. A arquitetura processa imagens de entrada com dimensões de $224 \times 224 \times 3$ *pixels*, seguindo um algoritmo que minimiza a redundância computacional, otimizando assim o uso de recursos de hardware (AGGARWAL, 2018). Ela alcançou um grande desempenho na ILSVRC de 2014, conquistando a primeira posição com uma taxa de erro significativamente baixa.

#### *2.4.3.4 ResNet*

Desenvolvida por He *et al.* (2016) para competir na ILSVRC de 2015, a *ResNet* (Rede Residual) destacou-se por sua arquitetura possuir uma profundidade elevada, marcando um avanço significativo em relação às arquiteturas anteriores. Esta inovação possibilitou que a *ResNet* alcançasse uma taxa de erro top-5 de apenas $3,6\%$, estabelecendo-se como o primeiro classificador com desempenho ao nível humano. A arquitetura *ResNet* possui diversas variações, incluindo a *ResNet-18*, que possui 65 camadas (Figura 23a) e a *ResNet-50*, com 168 camadas (Figura 23b), adaptando-se a diferentes requisitos e capacidades computacionais. Cada uma dessas variações mantém a essência da abordagem residual, mas com um número variado de camadas, oferecendo flexibilidade na escolha do modelo segundo a aplicação e os recursos disponíveis (AGGARWAL, 2018).

Figura 23 – Arquitetura *ResNet*.



(a) *ResNet-18*. (b) *ResNet-50*.

Fonte: elaborado pelo autor.

A arquitetura da *ResNet* distingue-se pelo emprego de conexões residuais, uma técnica inovadora que facilita a propagação do gradiente durante o treinamento de redes profundas. Conexões residuais são essencialmente atalhos que permitem que o gradiente seja transmitido diretamente através das camadas, evitando o problema comum do desaparecimento do gradiente em redes profundas. Isso é alcançado conectando a entrada de uma camada diretamente à saída de outra, tipicamente algumas camadas adiante. Essas conexões ajudam a preservar a informação e o gradiente ao longo do processo de

treinamento, permitindo o desenvolvimento de modelos com dezenas ou mesmo centenas de camadas. O uso de conexões residuais na *ResNet* resultou em uma ampla gama de aplicações, desde classificação e reconhecimento de imagens até tarefas mais complexas de visão computacional, tirando proveito da habilidade da rede em aprender representações ricas e detalhadas em vários níveis de abstração (AGGARWAL, 2018).

### 2.4.4 *Transfer Learning*

*Transfer Learning* é uma técnica no campo do aprendizado de máquina, na qual um modelo desenvolvido para uma tarefa específica é reaproveitado como ponto de partida para uma segunda tarefa relacionada. Essa abordagem é particularmente eficaz quando as tarefas compartilham características ou padrões internos, permitindo que o modelo transfira o conhecimento adquirido de um domínio para outro. Esta técnica tem mostrado ser extremamente valiosa em uma ampla gama de aplicações, como na adaptação de modelos de visão computacional treinados em um grande conjunto de dados para reconhecer objetos em contextos específicos com poucos dados de treinamento. A capacidade de transferir aprendizado de um contexto para outro reduz significativamente a quantidade de dados rotulados necessários e o esforço computacional para desenvolver modelos eficazes em novas tarefas, tornando *Transfer Learning* uma abordagem crucial no avanço rápido e eficiente de diversas áreas da inteligência artificial (GOODFELLOW; BENGIO; COURVILLE, 2016).

No âmbito de *Transfer Learning*, as redes neurais pré-treinadas desempenham um papel fundamental, permitindo a reutilização eficaz de modelos complexos em novas aplicações com ajustes mínimos. Essas redes, frequentemente desenvolvidas e treinadas em conjuntos de dados extensos e variados, podem ser adaptadas para tarefas específicas simplesmente alterando a camada de saída. Tal abordagem é eficiente, pois as camadas intermediárias, que já aprenderam representações gerais e abstratas dos dados, geralmente não necessitam de ajustes significativos. Em muitos casos, apenas os pesos da nova camada de saída são recalculados para se adequar à nova tarefa, enquanto as características aprendidas nas camadas anteriores são mantidas (AGGARWAL, 2018).

### 2.4.5 Generalização, *overfitting* e *Data Augmentation*

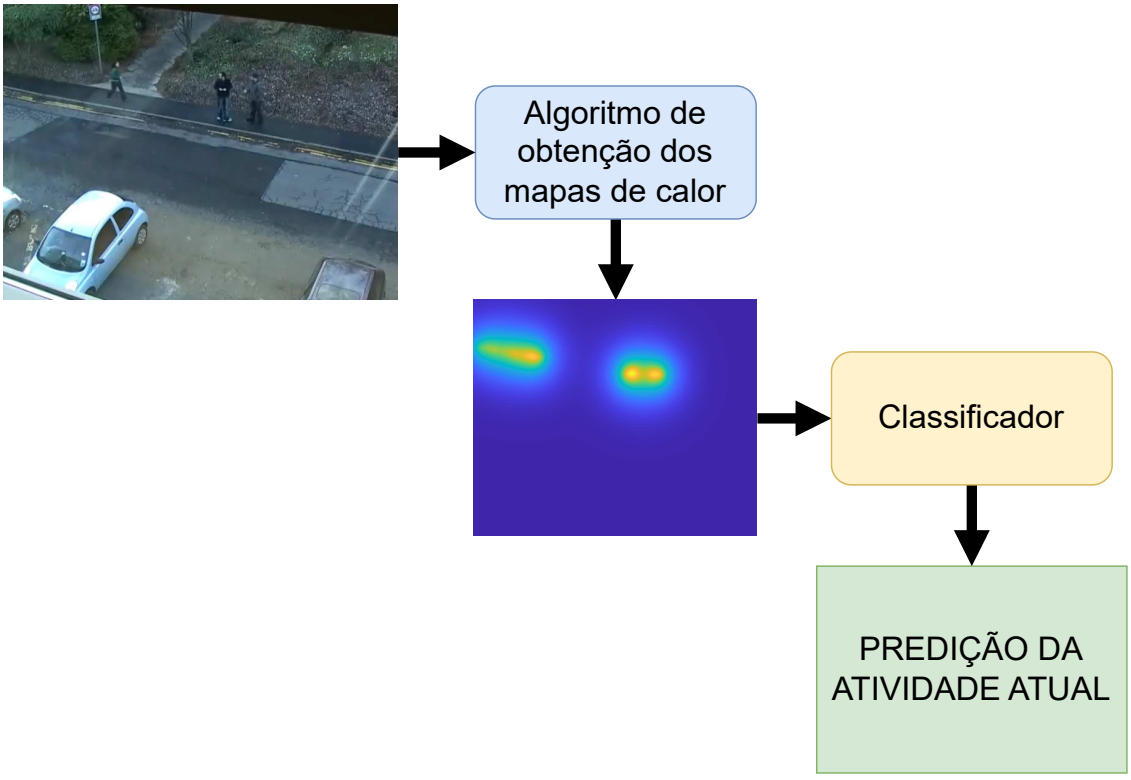
Um aspecto crucial no treinamento de redes neurais é a generalização, que se refere à capacidade da rede de performar eficientemente em dados não observados anteriormente (GOODFELLOW; BENGIO; COURVILLE, 2016). Quando uma rede neural é treinada de maneira que ela memorize as amostras do conjunto de treinamento, em vez de aprender suas características gerais, ocorre um fenômeno conhecido como *overfitting*. Esse fenômeno é caracterizado por uma alta precisão durante a etapa de treinamento, mas baixa eficácia em dados de teste, indicando uma falha em sua generalização (GOODFELLOW; BENGIO; COURVILLE, 2016).

Para mitigar o *overfitting* e melhorar a generalização em algoritmos de classificação de imagens digitais, dentre outros procedimentos, desenvolveu-se a técnica conhecida como *Data Augmentation*. Ela consiste na criação de amostras sintéticas a partir do conjunto de dados original, que por conseguinte são utilizadas durante o treinamento da rede neural. Essas amostras sintéticas são geradas a partir das amostras originais por meio de várias operações de processamento. No caso de imagens digitais, pode-se utilizar as operações discutidas na Seção 2.2. Desse modo, novas imagens são criadas aplicando-se alterações em brilho, contraste, escala, orientação, posição, entre outras modificações possíveis. Adicionalmente, filtragens 2D e a adição de ruído podem ser empregadas para aumentar a diversidade do conjunto de dados. Contudo, a etapa de *Data Augmentation* não requer necessariamente o armazenamento das imagens sintéticas no sistema de arquivos, podendo ser geradas automaticamente durante o treinamento (AGGARWAL, 2018).

# 3 DESENVOLVIMENTO

O sistema implementado destina-se à geração e classificação de mapas de calor visando identificar atividades em agrupamentos humanos. Em particular, este trabalho fundamenta-se fortemente nas metodologias apresentadas por (LIN *et al.*, 2013), especialmente nas etapas relacionadas à geração dos referidos mapas de calor. Entretanto, na fase de classificação, opta-se por uma abordagem diferente, empregando redes neurais convolucionais e estratégias de *data augmentation*. De forma simplificada, o fluxo de processamento do sistema implementado - ilustrado na Figura 24 - ocorre da seguinte forma: a partir de um vídeo que contém as posições rotuladas dos indivíduos, gera-se um mapa de calor para cada *frame*, conforme descrito na Seção 3.1. Posteriormente, cada um desses mapas é submetido a uma rede neural pré-treinada, detalhada na Seção 3.2, resultando na predição da atividade do agrupamento de pessoas em curso.

Figura 24 – Algoritmo de detecção de atividades em grupo.




Fonte: elaborado pelo autor.

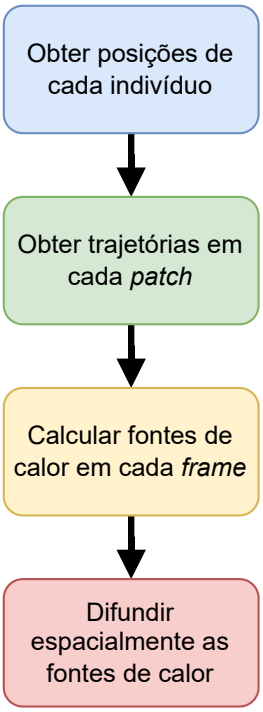
Para o desenvolvimento, validação e avaliação do sistema implementado, utilizou-se o *dataset* fornecido por (BLUNSDEN; FISHER, 2010), que consiste em oito clipes de vídeo de variadas durações. Estes clipes capturam a interação entre indivíduos em um estacionamento empresarial durante o horário de saída. É importante salientar que o *dataset* acompanha um documento contendo anotações de *bounding-box* para cada objeto

(pessoa) em determinados intervalos de *frames*. Tal informação viabiliza o cálculo da posição dos indivíduos. Além disso, o *dataset* também disponibiliza outro documento, que identifica os tipos de interação que ocorrem entre as pessoas com os correspondentes intervalos de *frames*. Tais demarcações são informações valiosas para o treinamento e avaliação das redes neurais responsáveis por classificar as atividades do agrupamento de pessoas.

## 3.1 OBTENÇÃO DOS MAPAS DE CALOR DAS ATIVIDADES EM GRUPO

Conforme demonstrado na Figura 25, o algoritmo para obtenção de mapas de calor estrutura-se em quatro etapas fundamentais. Inicialmente, utilizam-se as informações fornecidas no documento anexo ao *dataset* para determinar as posições de cada indivíduo em cada *frame* do vídeo. Em seguida, identificam-se as trajetórias presentes em cada *patch* (setores de *pixels*) do vídeo. Posteriormente, calculam-se os valores correspondentes às fontes de calor para cada instante do vídeo. Por fim, a quarta e última etapa envolve a difusão espacial dessas fontes de calor, com base nos valores calculados na etapa anterior. Este procedimento garante uma representação informativa das atividades em grupos.

Figura 25 – Algoritmo de obtenção de mapas de calor.



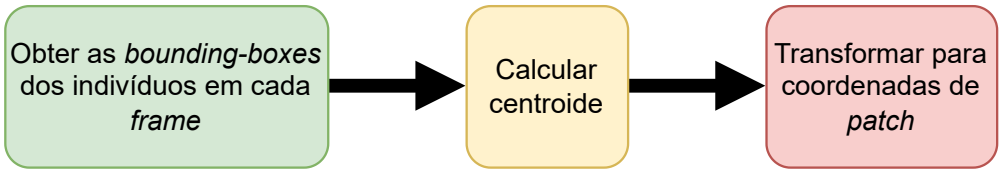

Fonte: elaborado pelo autor.

### 3.1.1 Posições dos indivíduos

No processo de obtenção das posições dos indivíduos em cada *frame* do vídeo, recorre-se aos *bounding-boxes* fornecidos pelo *dataset*. Especificamente, *bounding-box* é definido como o menor retângulo com arestas paralelas ao eixo da imagem que contém o objeto de interesse. Exemplos de *bounding-boxes* são apresentadas na Figura 27 (retângulos com contorno em vermelho). Ademais, um *bounding-box* pode ser completamente definido a partir das coordenadas dos seus pontos superior esquerdo $P_1 = (u_{min}, v_{min})$ e inferior direito $P_2 = (u_{max}, v_{max})$.

Conforme detalhado na Figura 26, após a extração das *bounding-boxes* mediante a Equação (24), a etapa subsequente envolve o cálculo dos seus centroides, conforme expresso na Equação (25). Estes centroides atuam como posições de referência para cada indivíduo. Ilustrativamente, a Figura 27 apresenta um *frame* do *dataset* contendo os centroides (pontos em verde) que definem a posição de cada indivíduo na cena.

Figura 26 – Etapas para obter a posição de cada indivíduo.




Fonte: elaborado pelo autor.

$$\{P_1,\ \ P_2\} = \{(u_{min}, v_{min}),\ \ (u_{max}, v_{max})\} \tag{24}$$

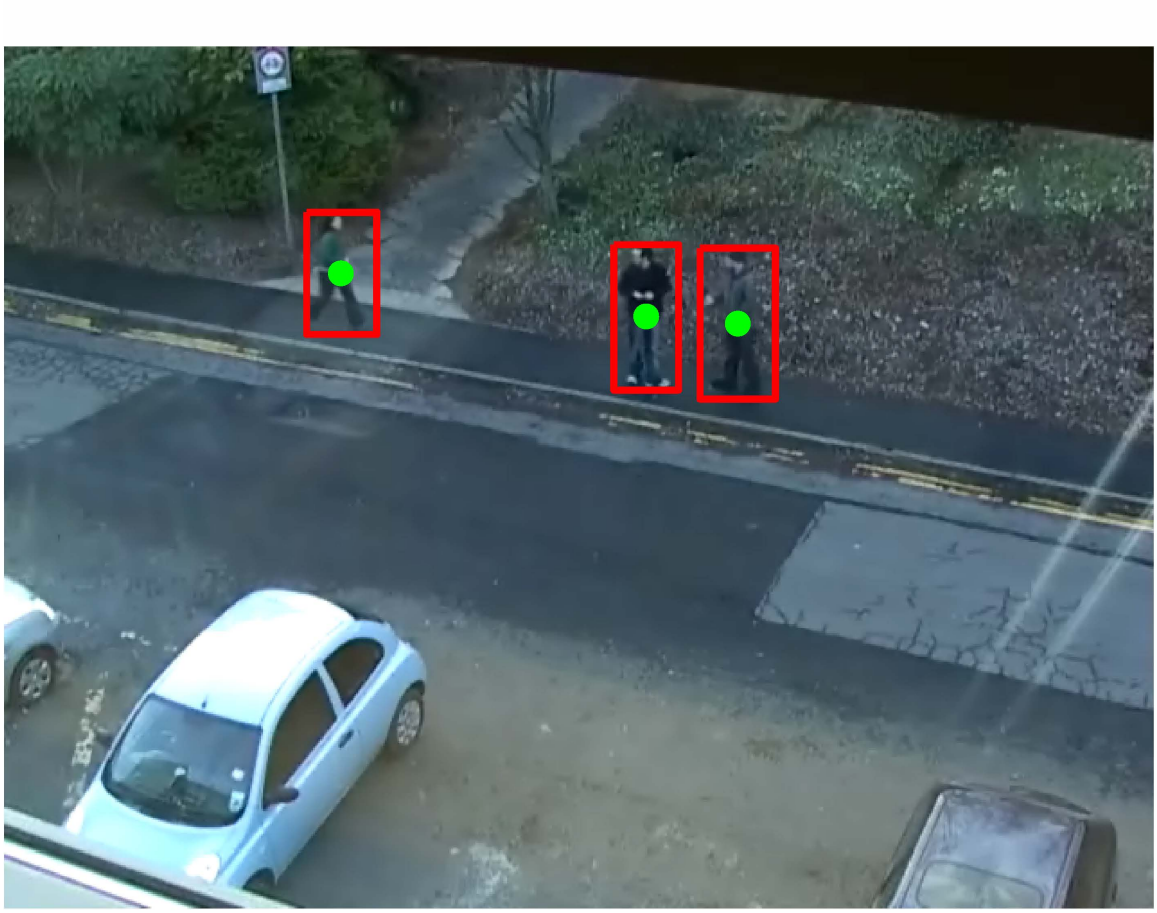
$$C = \left(\frac{u_{min} + u_{max}}{2},\ \ \frac{v_{min} + v_{max}}{2}\right). \tag{25}$$

Entretanto, deve-se evidenciar que as coordenadas obtidas são expressas em unidades de *pixel*. Desse modo, com o intuito de otimizar o tempo de processamento nas etapas descritas nas Seções 3.1.2, 3.1.3 e 3.1.4, adota-se uma estratégia de divisão da imagem por *patches*. Assim, a imagem original é dividida em diversos setores (*patches*) conforme ilustrado na

Figura 28. Esta divisão ocorre com base em um parâmetro $h$ (definido pelo projetista), que representa a largura e a altura de cada *patch*.

Ao término deste procedimento, a divisão por *patches* resulta em um remapeamento da imagem original em uma imagem de dimensões reduzidas, conforme demonstrado na Figura 29. Assim, embora um indivíduo se desloque entre os *pixels* de um *patch* específico na imagem original, ele permanecerá fixo no *pixel* mapeado na imagem de saída. Tal procedimento, embora implique em uma diminuição na resolução dos mapas de calor, contribui significativamente para a redução do tempo de processamento.

Figura 27 – *Bounding-boxes* e centroides em um frame.



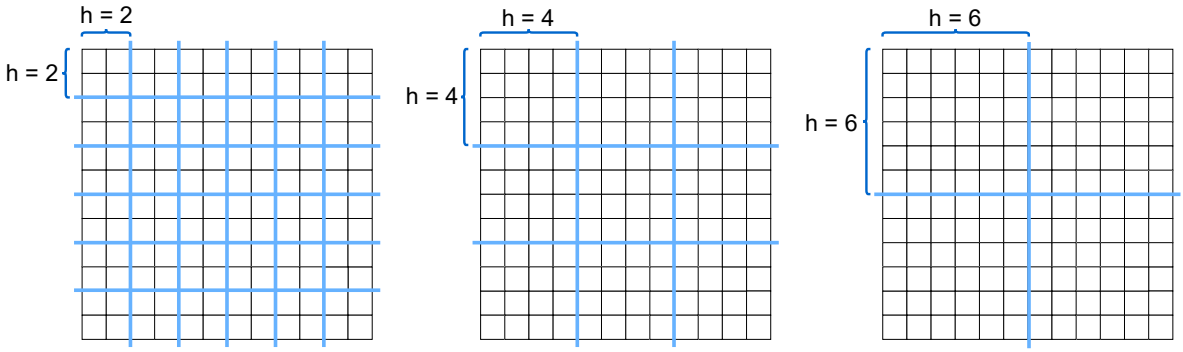
Fonte: elaborado pelo autor.

Para encontrar as coordenadas $(u', v')$ de um *patch* a partir das coordenadas $(u, v)$ de um *pixel* da imagem original, basta utilizar a Equação (26), na qual $\lceil \cdot \rceil$ denota a operação de aproximação para o próximo número inteiro.

$$(u', v') = \left( \left\lceil \frac{u}{h} \right\rceil , \left\lceil \frac{v}{h} \right\rceil \right) \tag{26}$$

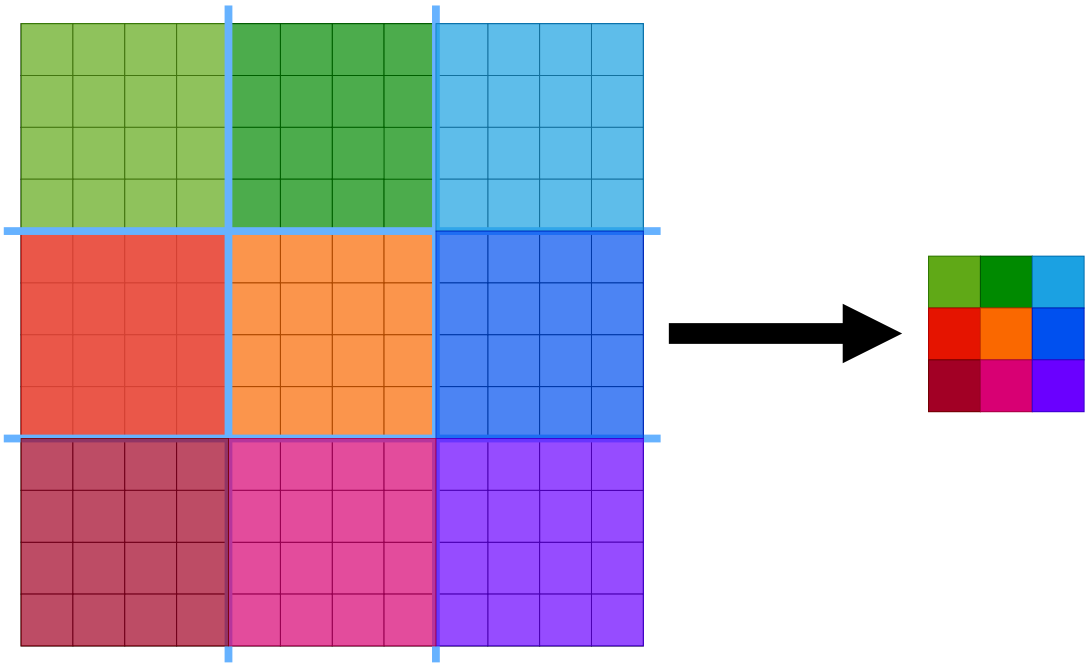
Como resultado dessa etapa, obtém-se uma lista de posições para cada indivíduo na imagem reduzida (resultante da divisão em *patches*). Ademais, define-se que cada objeto que armazena a posição de um indivíduo deve ser composto por três informações fundamentais: *frame* de entrada, *frame* de saída e coordenadas. A Figura 30 mostra a conversão entre as coordenadas de *pixel* e de *patch*, utilizando como base uma divisão com $h = 4$ (semelhante à observada na Figura 28).

Figura 28 – Divisão de uma imagem por *patches*.




Fonte: elaborado pelo autor.

Figura 29 – Mapeamento entre *patches* e *pixels*.

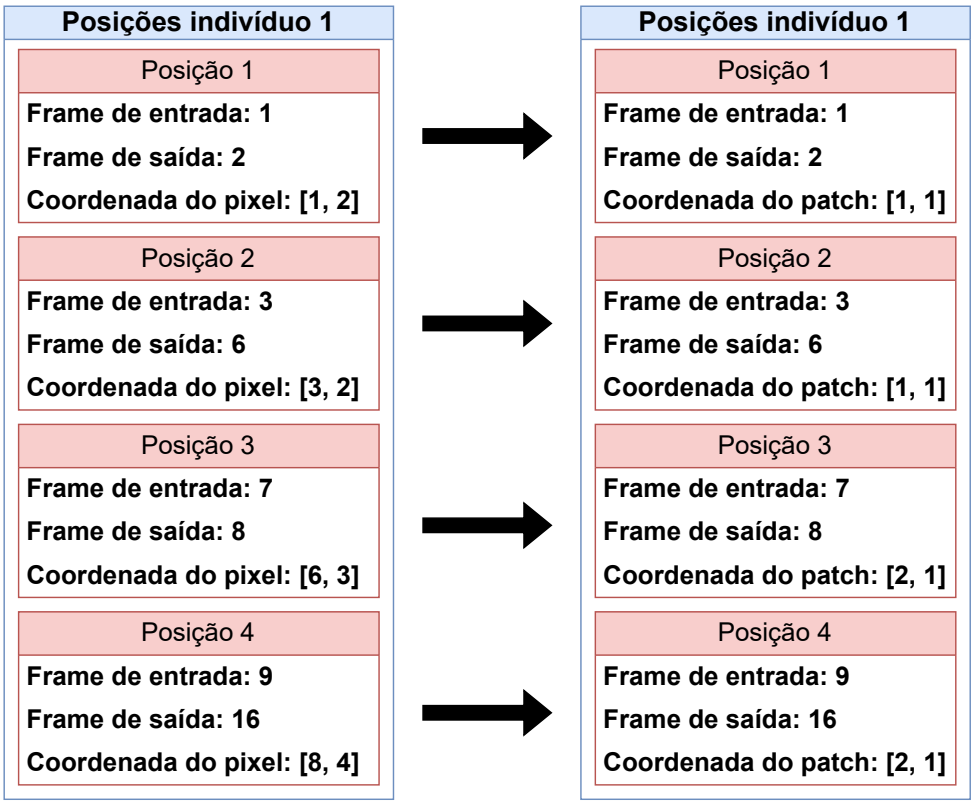


Fonte: elaborado pelo autor.

### 3.1.2 Trajetórias em cada patch

A identificação das trajetórias dos indivíduos constitui uma etapa subsequente à obtenção das posições (Seção 3.1.1). Uma trajetória pode ser definida como um conjunto de posições de um corpo, especificando as coordenadas e instantes de cada posição. Ademais, em uma cena específica, um indivíduo pode realizar uma trajetória contínua por diversos *patches* diferentes. Contudo, para o desenvolvimento do algoritmo de obtenção das fontes de calor, necessita-se obter as trajetórias do ponto de vista de cada *patch*. Desse modo, dada uma trajetória completa de um indivíduo, pode ser realizada a sua subdivisão conforme os

Figura 30 – Conversão entre posições com coordenadas de *pixels* e *patches*.




Fonte: elaborado pelo autor.

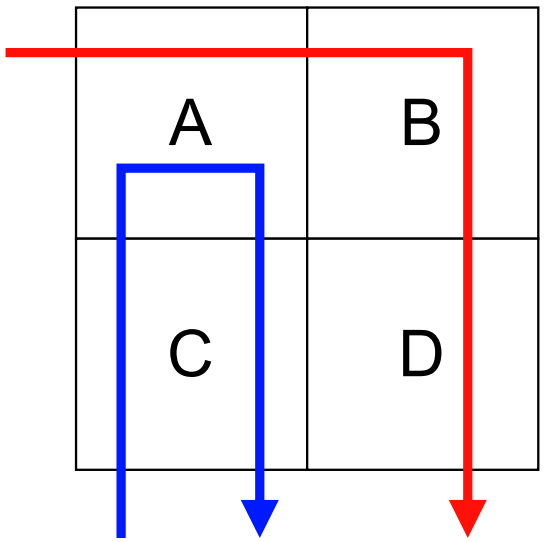
*patches* que e o indivíduo percorre, demarcando os *frames* de entrada e saída daquele *patch*. Desse modo, é importante observar que a saída e posterior reentrada de um indivíduo no mesmo *patch* geram duas trajetórias distintas.

Esta particularidade é ilustrada na Figura 31, na qual se nota que um indivíduo transita pelos *patches* A, B e D, entrando e saindo de cada um deles apenas uma vez, resultando em um total de três trajetórias. Em contraste, um segundo indivíduo, embora transite apenas pelos *patches* A e C, também registra três trajetórias, devido a reentrada no *patch* C. Por consequência, define-se que a mudança de *patch* por parte de um indivíduo serve como indicador para o início de uma nova trajetória.

A fim de automatizar o processo de obtenção das trajetórias a partir da lista de posições, desenvolveu-se o algoritmo apresentado na Figura 32. Especificamente, para cada indivíduo, inicia-se o processamento ao obter um elemento da sua fila de posições. Subsequentemente, comparam-se as coordenadas do seu *patch* com o *patch* anterior (considerado nulo na primeira iteração). Caso sejam diferentes, adiciona-se uma nova trajetória à lista de trajetórias desse indivíduo e atualiza-se a variável que armazena as coordenadas do *patch* anterior. Este procedimento repete-se até o término da fila de posições.

Ao final da execução deste algoritmo, a lista original de posições transforma-se

Figura 31 – *Patches* com múltiplas trajetórias.




Fonte: elaborado pelo autor.

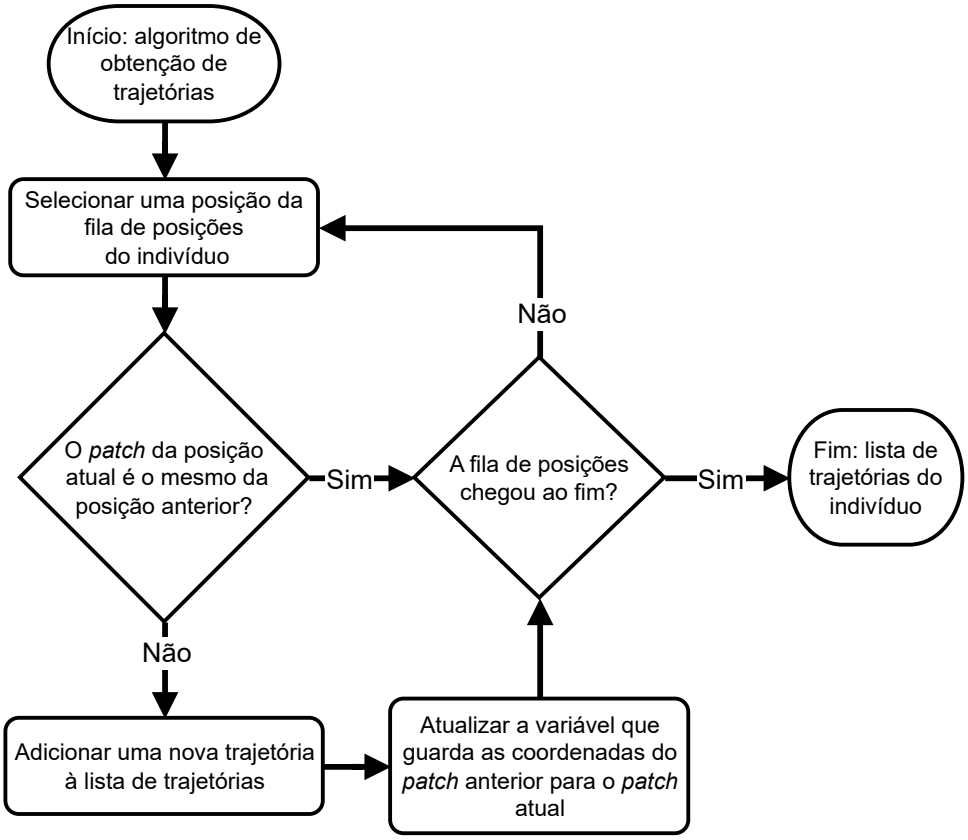
em uma lista de trajetórias. Ademais, é importante ressaltar que, durante esse processo, são armazenados os valores dos *frames* de entrada e saída para cada trajetória, conforme ilustrado na Figura 33. Estes dados são cruciais para as análises a serem realizadas na subsequente etapa, discutida na Seção 3.1.3.

### 3.1.3 Fontes de calor em cada frame

No algoritmo implementado para geração dos mapas de calor, cada indivíduo é considerado um emissor de calor, cujo deslocamento no ambiente resulta em um rastro térmico. Particularmente, um maior período de permanência em uma localização específica resulta em um acúmulo de energia térmica nessa coordenada. Inicialmente, o mapa de calor é instanciado apenas com valores nulos. Contudo, à medida que os indivíduos emergem e se movem no vídeo, esse mapa é atualizado com as respectivas fontes de calor calculadas. Vale destacar que apenas os *patches* que contam com trajetórias associadas a eles - isto é, já percorridos por algum indivíduo - contribuem como fontes de calor. Adicionalmente, conforme a teoria da termodinâmica, observa-se um fenômeno de decaimento temporal nas fontes de calor, caracterizado por uma redução exponencial da energia térmica ao longo do tempo. Em resumo, fontes de calor mais recentes tendem a exibir maiores níveis de energia térmica (LIN *et al.*, 2013).

Para efetuar o cálculo das fontes de calor, são empregadas algumas equações fornecidas por Lin *et al.* (2013). A Equação (27) permite o cálculo do calor total $E_i$ armazenado em um *patch* $i$, somando as contribuições térmicas $\overline{E_{i,j}}$ de todas as trajetórias $j$ que o interceptam até o *frame* atual. Observe que tal equação incorpora o efeito de decai-

Figura 32 – Algoritmo de obtenção das trajetórias de cada *patch*.



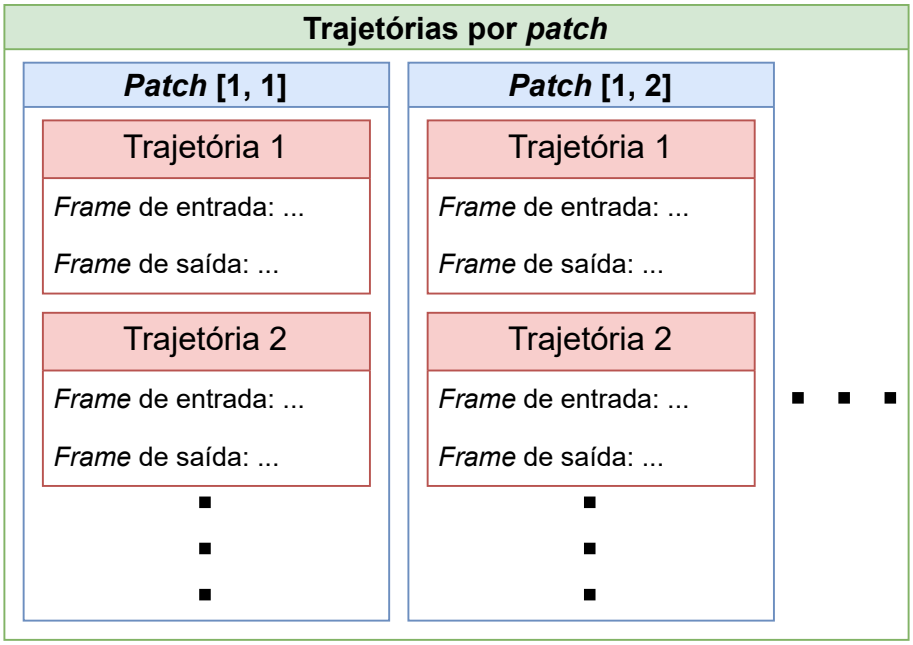

Fonte: elaborado pelo autor.

mento exponencial previamente mencionado, controlado pela constante $k_t$. Nesse contexto, quanto mais distante o *frame* atual estiver do *frame* de saída da trajetória, menor será a contribuição de calor no *patch* considerado.

$$E_i = \sum_j \overline{E_{i,j}} \cdot e^{-k_t(t_{atual} - t_{saida})} \tag{27}$$

Especificamente, a energia térmica acumulada $\overline{E_{i,j}}$ em um *patch* $i$ devido a uma trajetória específica $j$ é determinada através da Equação (28). Essa equação utiliza os instantes de tempo dos *frames* de entrada e saída, obtidas conforme descrito na Seção 3.1.2. Assim, percebe-se que trajetórias mais breves, nas quais os instantes $t_{entrada,j}$ e $t_{saida,j}$ são próximos, resultam em menores quantidades de energia térmica acumulada. Em contraste, quando $t_{saida,j}$ é substancialmente maior que $t_{entrada,j}$, a energia térmica tende a ser mais elevada, dada a maior permanência do objeto no *patch*. No entanto, neste trabalho, optou-se por uma modificação nesta equação no cálculo da energia térmica acumulada, resultando na Equação (29). Em tal equação a constante $C$ é considerada 1 e a constante de decaimento temporal $k_t$ é substituída pela constante de ganho de calor $k_e$. Tal ajuste mostrou-se relevante para desacoplar o cálculo de calor acumulado do seu respectivo

Figura 33 – Estrutura de dados para as trajetórias de cada *patch*.




Fonte: elaborado pelo autor.

decaimento, oferecendo assim mais graus de liberdade na geração dos mapas de calor.

$$\begin{aligned} \overline{E_{i,j}} &= \int_0^{t_{saida,j}-t_{entrada,j}} C \cdot e^{-k_t t}\, dt \\ &= \frac{C}{k_t}\left(1 - e^{-k_t(t_{saida,j}-t_{entrada,j})}\right) \end{aligned} \tag{28}$$

$$\overline{E_{i,j}} = \frac{1}{k_e}\left(1 - e^{-k_e(t_{saida,j}-t_{entrada,j})}\right) \tag{29}$$

A metodologia adotada visa calcular o mapa de calor para cada *frame* de forma individual e sequencial. Tal abordagem parte do pressuposto de que as trajetórias dos indivíduos estão organizadas de maneira sequencial, um atributo assegurado na fase de extração das trajetórias. Ademais, o algoritmo se fundamenta essencialmente nas Equações Equação (27) e Equação (29).

Dado este contexto, a lista de trajetórias de todos os indivíduos é sempre percorrida desde o início, visando calcular as fontes de calor em cada *frame*. Este procedimento é necessário porque o mapa de calor evolui temporalmente, e o *frame* atual é influenciado pelos *frames* anteriores.

O fluxograma contido na Figura 34 apresenta todos os passos necessários para calcular a contribuição de calor de um indivíduo para um determinado *patch* em um *frame*. De acordo com tal fluxograma, se o instante de tempo do *frame* atual estiver compreendido entre os instantes dos *frames* de entrada e de saída da trajetória atual, calcula-se a sua contribuição para o *patch* em questão, conforme estabelecido pela Equação (29). Isso

indica uma interação ativa do indivíduo com o *patch* no *frame* atual. Caso contrário, se o instante do *frame* atual for maior que o do *frame* de saída da trajetória, calcula-se a contribuição de calor e aplica-se o decaimento temporal especificado pela Equação (27). Tal lógica se faz necessária porque nessa situação o indivíduo já deixou o *patch*, implicando em um decaimento na sua contribuição térmica ao longo do tempo.

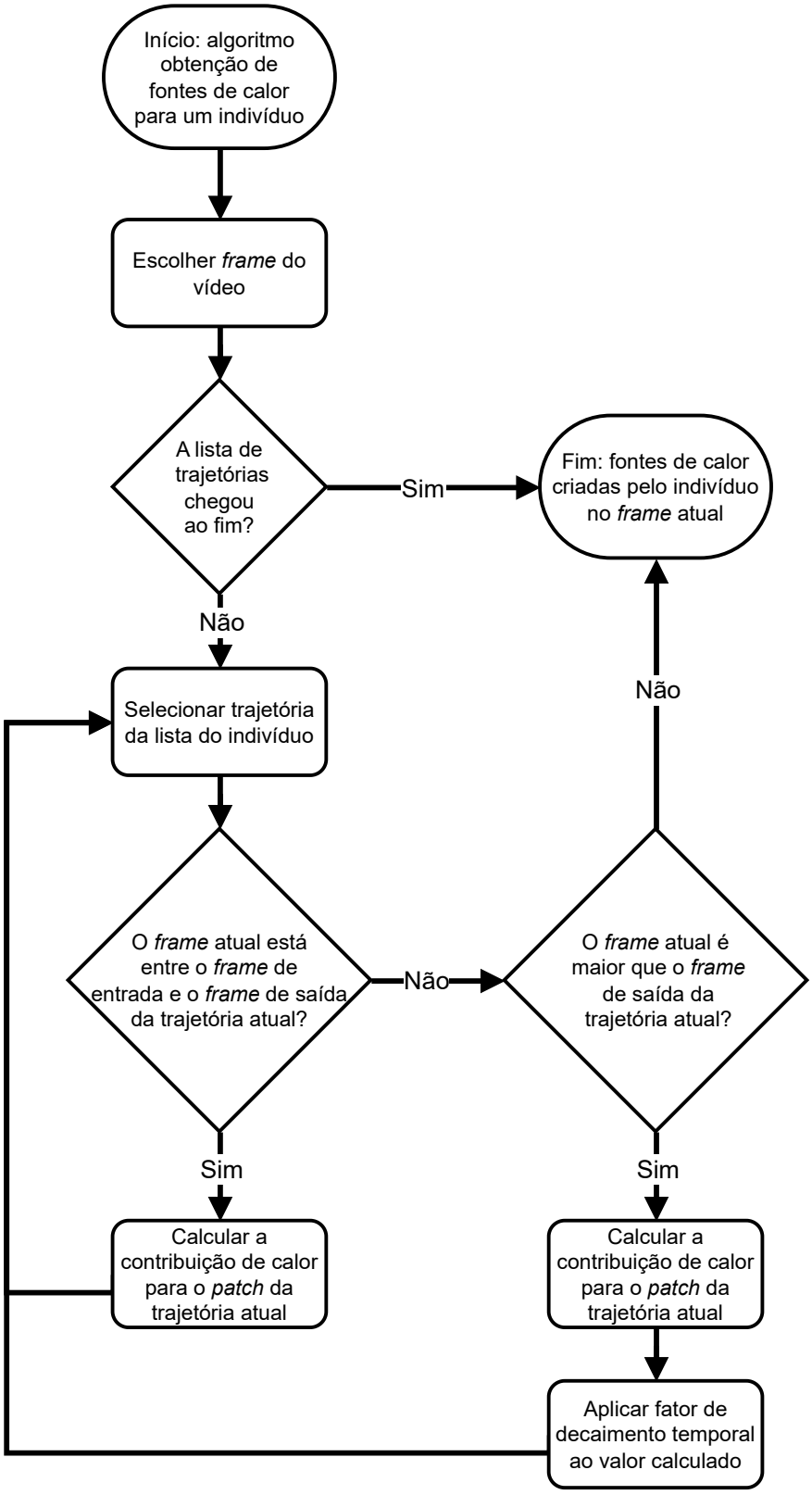
Entretanto, se ambas as condições acima não se verificarem, ou seja, o instante de tempo do *frame* atual for menor que o instante do *frame* de entrada da trajetória corrente, nenhum calor será adicionado ao *patch* atual, dado que o indivíduo ainda não executou tal trajetória. Todos os indivíduos são submetidos ao mesmo algoritmo, e suas contribuições são somadas nos respectivos *patches* do mapa de calor. Esta abordagem abrange cenários em que múltiplos indivíduos interagem com o mesmo *patch*. Assim, ao final do processo, obtém-se o mapa de calor completo contendo todas as fontes de calor para o *frame* em análise, ilustrado pela Figura 35.

Para ilustrar o funcionamento do algoritmo, uma situação hipotética é apresentada na Figura 36. Neste cenário, um mapa de calor $2 \times 2$ é considerado, onde dois indivíduos, A e B, se deslocam. O cálculo dos mapas de calor ocorre em sequência, iniciando no *frame* 1 e estendendo-se até o *frame* 18, com as constantes $k_t$ e $k_e$ unitárias. No intervalo do *frame* 1 ao *frame* 3, apenas o indivíduo A se encontra presente no mapa, ocupando o *patch* $[1, 1]$. A partir do *frame* 3, o indivíduo B entra em cena através do *patch* $[2, 1]$.

No *frame* 4, o indivíduo A se move para o *patch* $[1, 2]$, onde permanece até o *frame* 10. Simultaneamente, nos *frames* subsequentes, os indivíduos A e B convergem para o *patch* $[2, 2]$, sendo que A chega no *frame* 11 e B no *frame* 6. Ambos os indivíduos passam algum tempo neste *patch* antes de finalmente sair do mapa de calor.

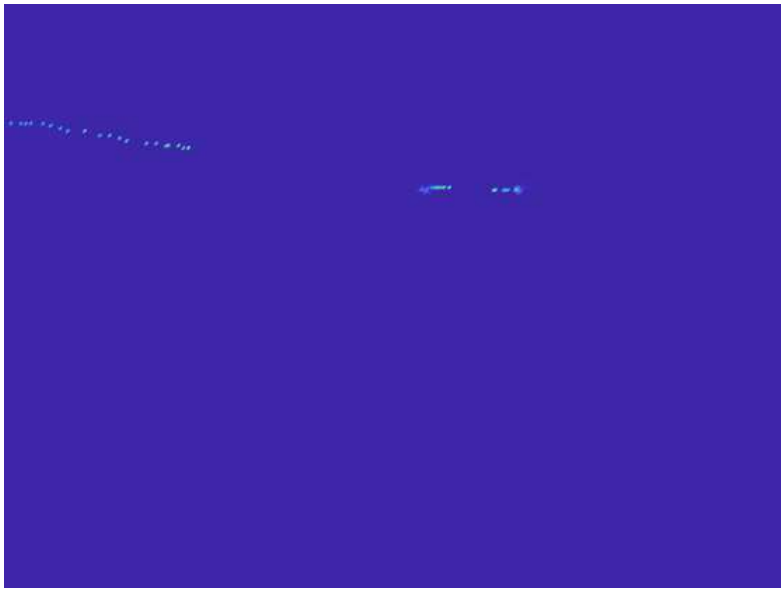
A progressão dessas interações *frame* a *frame* pode ser visualizada na Figura 37, na qual é possível observar o comportamento modelado pelo algoritmo. Fica evidente que quanto mais tempo um indivíduo permanece em um *patch*, maior é sua contribuição para o acúmulo de calor neste local. Além disso, nota-se que quando um indivíduo se retira de um *patch*, a contribuição de calor deste começa a decrescer exponencialmente nos *frames* subsequentes, conformando-se a lógica do algoritmo.

Figura 34 – Algoritmo para o cálculo das fontes de calor de cada indivíduo em um *frame*.



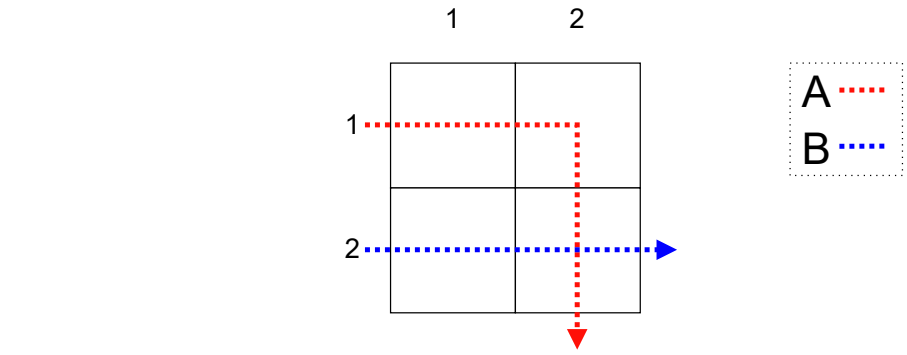

Fonte: elaborado pelo autor.

Figura 35 – Fontes de calor para um *frame*.



Fonte: elaborado pelo autor.

Figura 36 – Exemplo de trajetórias para 2 indivíduos.




**Trajetórias por *patch***

| ***Patch* [1, 1]** | ***Patch* [1, 2]** | ***Patch* [2, 1]** | ***Patch* [2, 2]** |
|---|---|---|---|
| Trajetória 1 | Trajetória 1 | Trajetória 1 | Trajetória 1 |
| *Frame* de entrada: 1 | *Frame* de entrada: 4 | *Frame* de entrada: 3 | *Frame* de entrada: 11 |
| *Frame* de saída: 3 | *Frame* de saída: 10 | *Frame* de saída: 5 | *Frame* de saída: 15 |
| Indivíduo: A | Indivíduo: A | Indivíduo: B | Indivíduo: A |
| | | | Trajetória 2 |
| | | | *Frame* de entrada: 6 |
| | | | *Frame* de saída: 13 |
| | | | Indivíduo: B |

Fonte: elaborado pelo autor.

Figura 37 – Fontes de calor resultantes do exemplo.

*Frame* 1

| | |
|---|---|
| 0,8647 | 0 |
| 0 | 0 |

*Frame* 4

| | |
|---|---|
| 0,3181 | 0,9975 |
| 0,8647 | 0 |

*Frame* 6

| | |
|---|---|
| 0,0430 | 0,9975 |
| 0,3181 | 0,9991 |

*Frame* 11

| | |
|---|---|
| 0,0002 | 0,3670 |
| 0,0021 | 1,9808 |

*Frame* 15

| | |
|---|---|
| 0 | 0,0067 |
| 0 | 1,1168 |

*Frame* 18

| | |
|---|---|
| 0 | 0,0003 |
| 0 | 0,0556 |

Fonte: elaborado pelo autor.

### 3.1.4 Difusão espacial do calor

Observando a Figura 35, percebe-se que o mapa de calor apresenta inicialmente poucos pontos isolados com valores não nulos, ilustrados pela tonalidade azul clara. A escassez de fontes de calor implica uma dificuldade na aplicação de metodologias de classificação da atividade do grupo presente na cena, uma vez que as variações nas trajetórias dos indivíduos dificultam identificar padrões consistentes.

Para mitigar as flutuações e promover uma distribuição mais uniforme de calor no mapa, realiza-se um processo de difusão térmica espacial. Nessa etapa, uma fração do calor originado nas fontes é distribuída aos *patches* vizinhos. Segundo a Equação (30), a quantificação do calor em um *patch* $i$ decorre da soma do calor proveniente de todas as $j$ fontes, ponderado por um fator exponencial (com expoente negativo), dependente da distância euclidiana entre os *patches* $i$ e $j$. Após a soma, o resultado é dividido pelo total $N$ de fontes de calor, analogamente a uma média ponderada. Destaca-se também a inclusão de um coeficiente $k_p$ (definido pelo projetista), responsável por ajustar a influência das fontes no *patch* em análise.

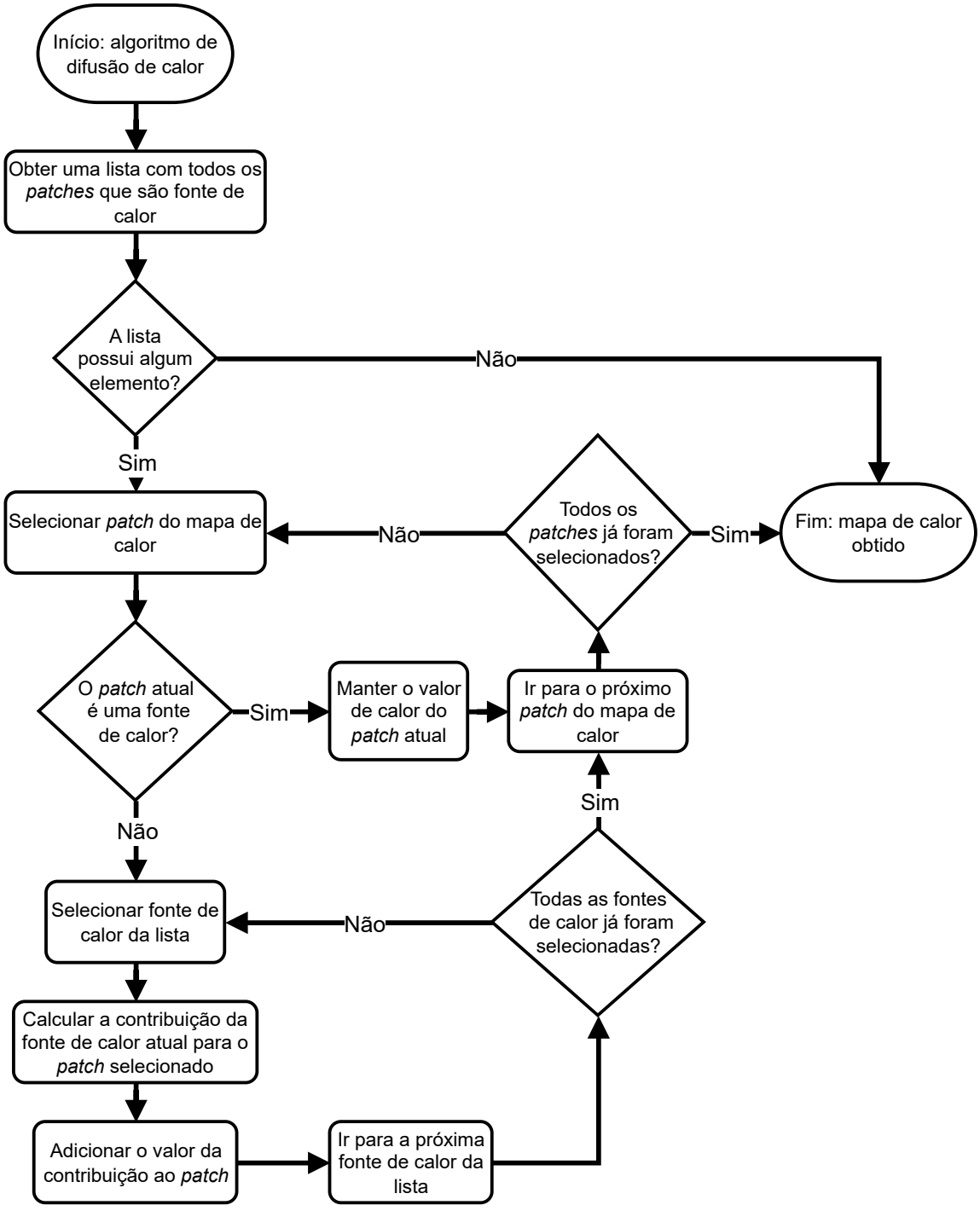
$$H_i = \frac{\sum_{j=1}^{N} E_j \cdot e^{-k_p \cdot d(i,j)}}{N} \tag{30}$$

Em termos gerais, conclui-se que a quantidade de calor recebida por um *patch* é diretamente proporcional a sua proximidade a uma fonte de calor. O procedimento para realizar a difusão térmica é detalhado na Figura 38, na qual são descritas as etapas para a obtenção do mapa de calor difundido. Inicia-se com a identificação dos *patches* que atuam como fontes de calor no mapa, isto é, aqueles com valores não nulos. Prossegue-se com a seleção do primeiro *patch* do mapa de calor. Caso este não seja uma fonte de calor, avalia-se a contribuição de cada fonte de calor ao *patch* em questão (percorrendo a lista de fontes de calor previamente identificadas). Entretanto, se o *patch* selecionado já for uma fonte, mantém-se o seu valor original. Este processo é repetido até que cada *patch* do mapa seja analisado e atribuído um valor correspondente.

Ao término, chega-se a um mapa de calor plenamente preenchido. A comparação do antes e depois da difusão é apresentada na Figura 39, evidenciando a propagação do calor a partir das fontes. Uma abordagem alternativa para visualizar mapas de calor pós-difusão envolve correlacioná-los com uma superfície tridimensional. Neste contexto, o eixo Z representa a quantidade de calor acumulado, conforme evidenciado na Figura 40.

Para ilustrar a aplicação do algoritmo, adota-se o exemplo hipotético na Figura 41, compreendendo um mapa de dimensões $4 \times 4$. Primeiramente, define-se o mapa de calor com três fontes na Figura 41a. Posteriormente, estima-se o calor no *patch* $[1, 1]$ com base nas três fontes e suas respectivas distâncias, conforme ilustrado na Figura 41b. Na sequência, analisa-se o *patch* $[1, 2]$, porém tendo em vista que o mesmo já constitui uma

Figura 38 – Algoritmo de difusão espacial dos mapas de calor.

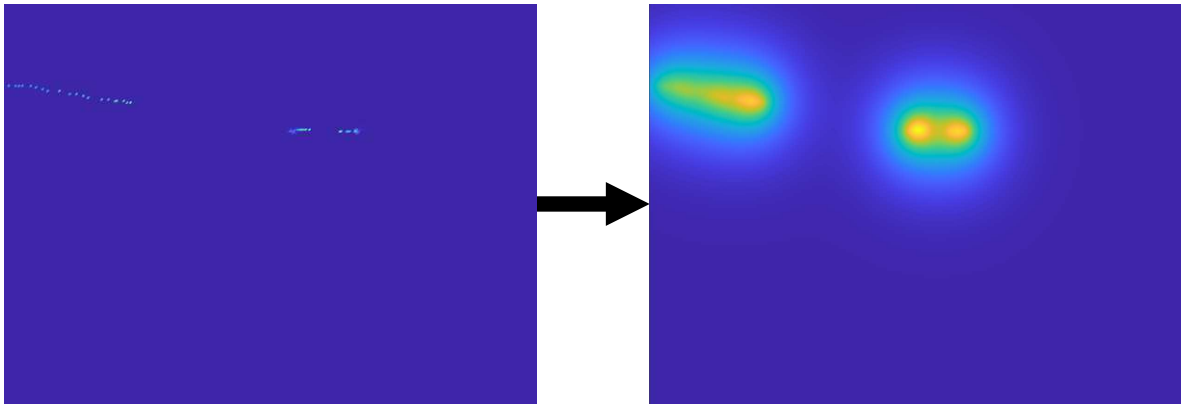



Fonte: elaborado pelo autor.

fonte de calor, seu valor é preservado, prosseguindo-se para o próximo *patch*. No cálculo do *patch* $[1, 3]$, aplica-se o mesmo método, considerando apenas as fontes iniciais, detalhado na Figura 41c. Executando este procedimento em todo o mapa, obtém-se o resultado na Figura 41d.

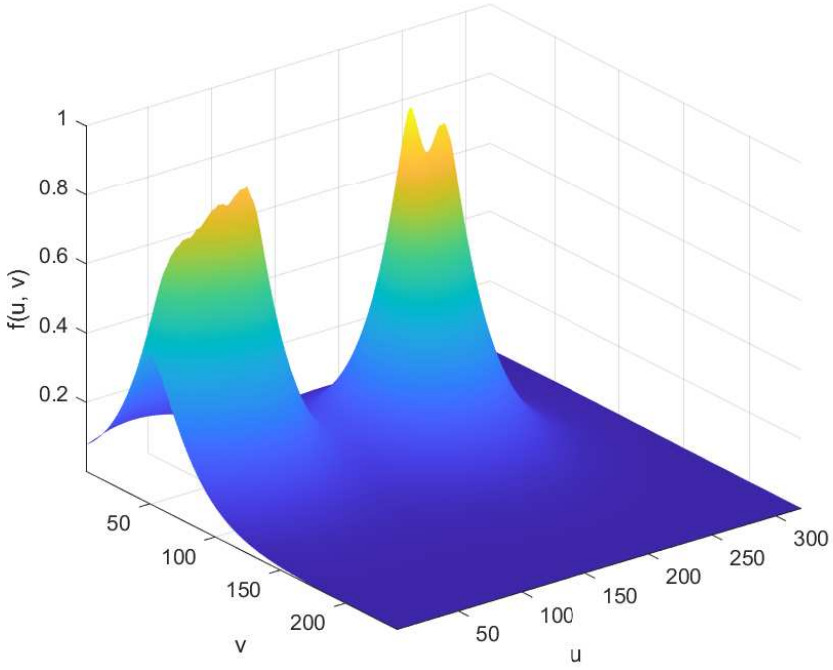
Consequentemente, após a realização da etapa de difusão térmica, procede-se à

Figura 39 – Mapa de calor após a difusão espacial.



Fonte: elaborado pelo autor.

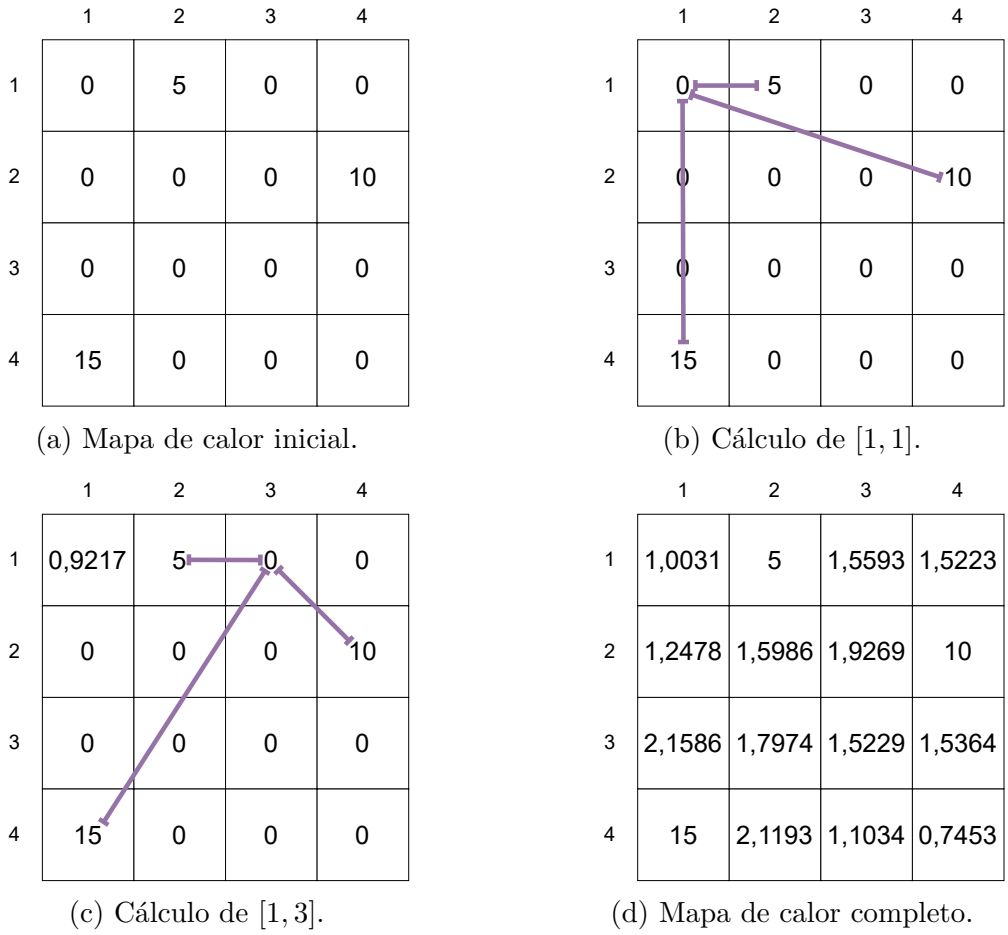
Figura 40 – Mapa de calor como superfície.




Fonte: elaborado pelo autor.

normalização dos valores dos *patches* em todo o mapa, realizando uma divisão pelo máximo valor registrado. Esta ação assegura que todos os valores estejam contidos no intervalo entre 0 e 1. Este procedimento de normalização é preponderante para se obter bom desempenho na etapa, na qual se efetua a classificação dos mapas de calor (Seção 3.2) por meio de redes neurais convolucionais.

Figura 41 – Exemplo do algoritmo de difusão de calor.



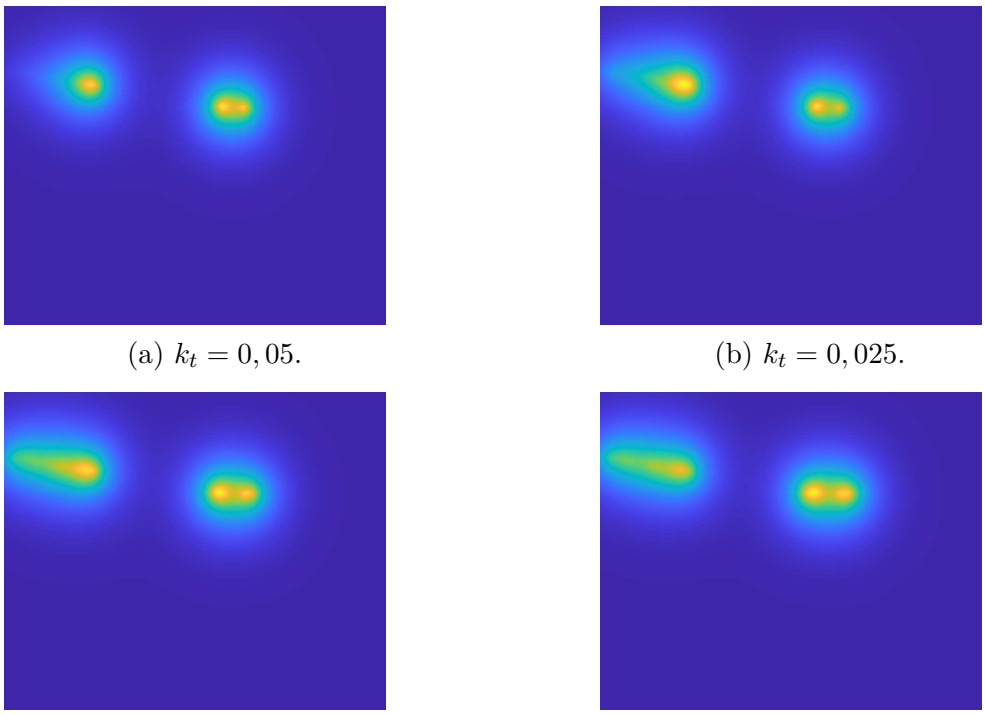

(a) Mapa de calor inicial. (b) Cálculo de $[1, 1]$.

(c) Cálculo de $[1, 3]$. (d) Mapa de calor completo.

Fonte: elaborado pelo autor.

Ademais, é pertinente examinar as influências dos parâmetros $k_t$ e $k_p$, utilizados nas Equações (27) e (30), respectivamente. Conforme ilustrado na Figura 42, identifica-se que $k_t$ é inversamente proporcional ao intervalo necessário para o declínio das fontes de calor, evidenciando que valores inferiores resultam em uma maior persistência das informações temporais das trajetórias dos indivíduos. De forma análoga, ao analisar a Figura 43, percebe-se que um valor reduzido de $k_p$ acarreta um aumento no fenômeno de difusão espacial, intensificando a transferência de calor entre os *patches*. Os valores de ambos os parâmetros são importantes para obter uma boa representação das atividades dos indivíduos, sendo essencial ajustá-los adequadamente para evitar a superposição de informações no mapa e, simultaneamente, reter a essência das informações temporais. Apesar dos mapas de calor serem normalmente apresentados como uma imagem colorida, com as cores transitando do azul ao amarelo, o que realmente se utiliza é uma representação em escala de cinza, conforme demonstrado na Figura 44, uma vez que um único canal de cor é suficiente para transmitir a quantificação do calor de cada *patch*.

Figura 42 – Mapas de calor com diferentes valores de decaimento temporal.

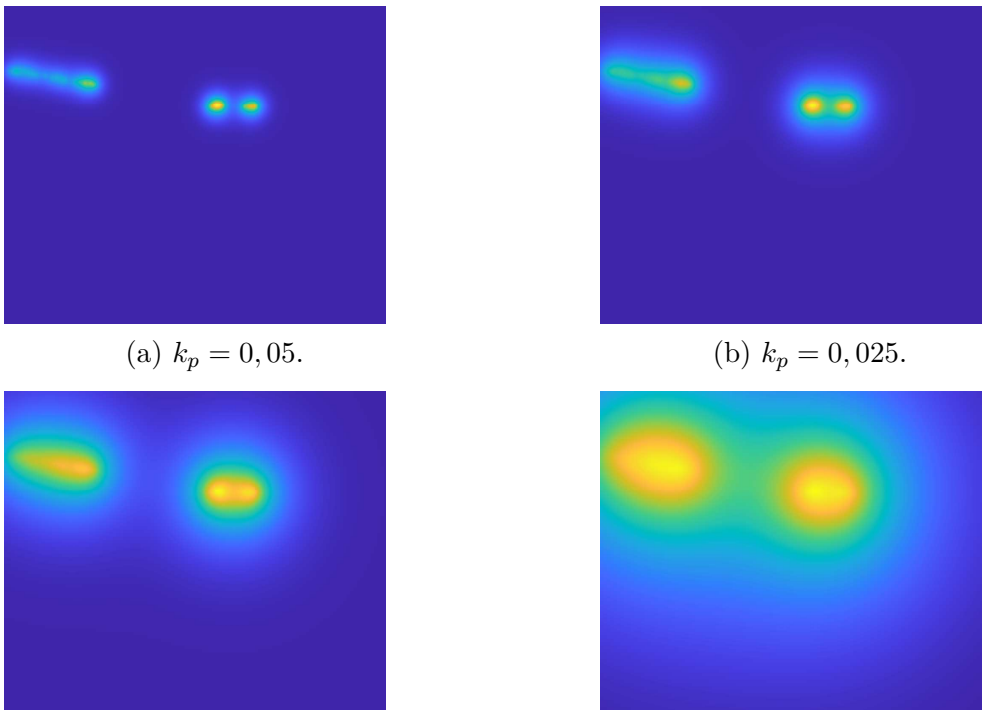


(a) $k_t = 0,05$. (b) $k_t = 0,025$.

(c) $k_t = 0,01$. (d) $k_t = 0,006$.

Fonte: elaborado pelo autor.

Figura 43 – Mapas de calor com diferentes valores de difusão espacial.

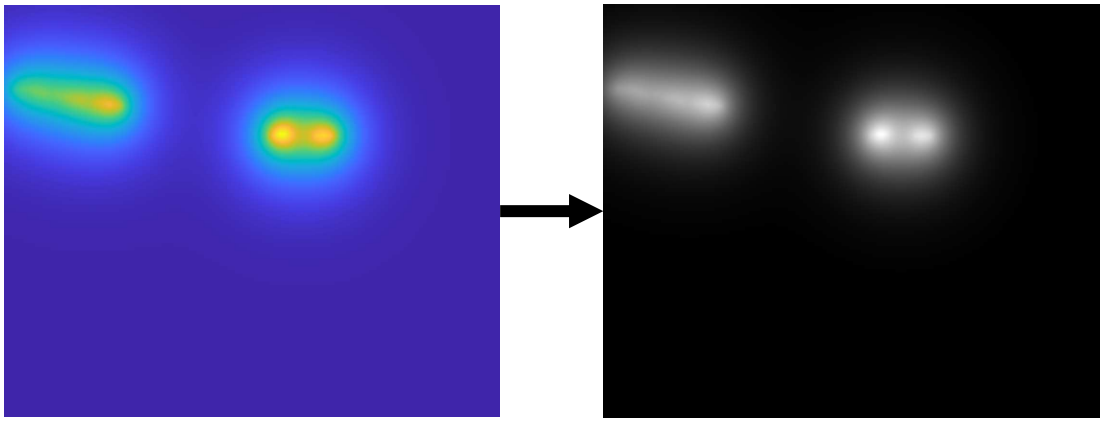


(a) $k_p = 0,05$. (b) $k_p = 0,025$.

(c) $k_p = 0,01$. (d) $k_p = 0,002$.

Fonte: elaborado pelo autor.

Figura 44 – Mapa de calor como imagem em escala de cinza.



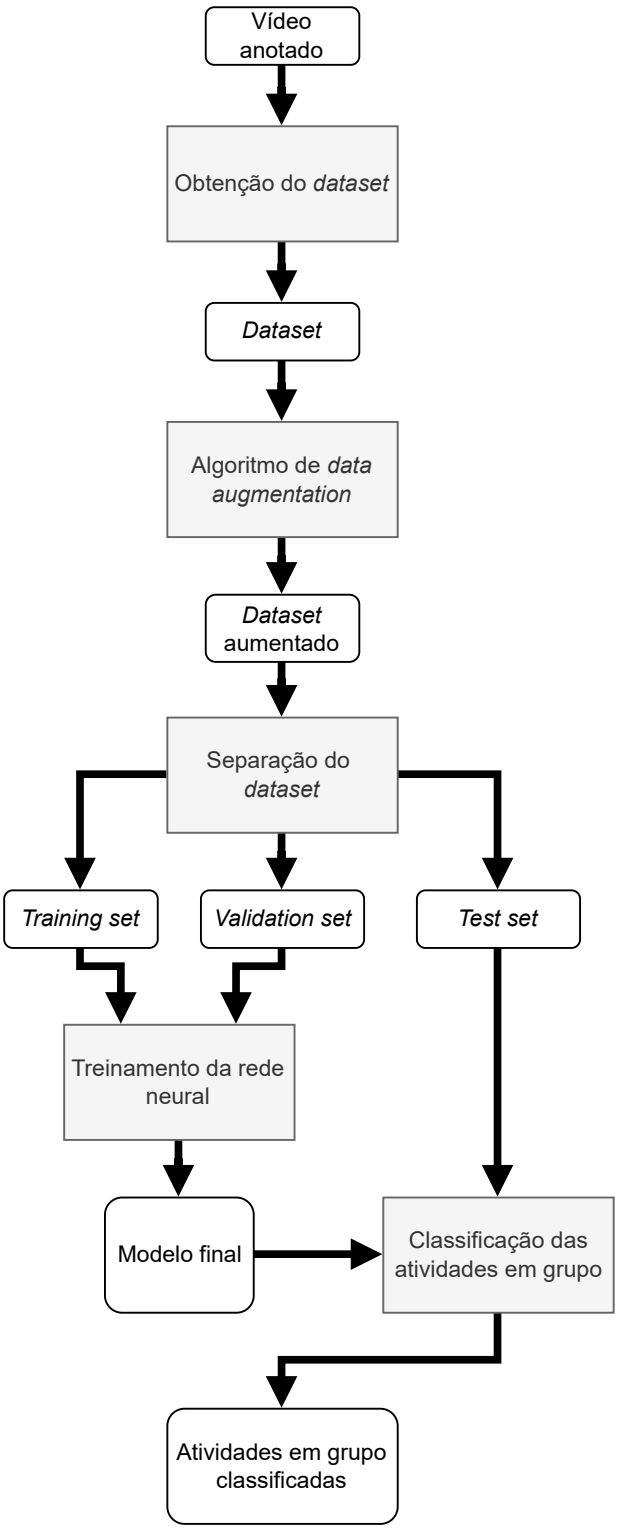
Fonte: elaborado pelo autor.

## 3.2 CLASSIFICAÇÃO DAS ATIVIDADES EM GRUPO

Neste trabalho o algoritmo de obtenção de mapas de calor (apresentado nas seções anteriores) é aplicado nos vídeos fornecidos por (BLUNSDEN; FISHER, 2010), visando a criação de um conjunto de dados abrangente de mapas de calor, representando diversas atividades em grupo. Como ilustrado na Figura 45, o conjunto de dados obtido é então submetido a uma etapa de *Data Augmentation*. Este processo visa expandir o número de amostras disponíveis, aumentando a diversidade e a representatividade do conjunto de dados, o que é crucial para melhora do desempenho do modelo de classificação. Após a ampliação do conjunto de dados, este é dividido em três subconjuntos distintos: treinamento, validação e teste, cada um desempenhando um papel específico no processo de treinamento e avaliação do modelo de classificação.

Em particular, como modelo de classificação dos mapas de calor, utiliza-se uma rede neural convolucional, a qual é treinada utilizando os conjuntos de treinamento e validação. Este processo é realizado com o auxílio da *Deep Learning Toolbox* do *software* MATLAB, que fornece as ferramentas necessárias para criar, treinar e manipular as redes neurais. Especificamente, diversos modelos de redes neurais, com variadas profundidades e custos computacionais, são treinados e avaliados, permitindo a realização de uma análise comparativa entre os modelos de classificação obtidos. Após o treinamento, o conjunto de teste é utilizado para verificar a precisão de cada classificador obtido. Este conjunto é particularmente importante, por ser composto por mapas de calor não utilizados durante o treinamento, oferecendo assim uma avaliação imparcial do desempenho do modelo em dados novos e desconhecidos.

Figura 45 – Etapas de elaboração e utilização do *dataset*.




Fonte: elaborado pelo autor.

### 3.2.1 Obtenção do *dataset* de mapas de calor

Para a obtenção de um conjunto de dados consistente de mapas de calor representando atividades em grupo, (BLUNSDEN; FISHER, 2010) fornece um arquivo com demarcações temporais de interações entre grupos de indivíduos. Este arquivo, estruturado como uma lista, possui campos que contém o início e o fim de cada interação (*frames* de início e fim) e o nome da atividade em grupo ocorrendo em tal intervalo temporal. As atividades disponibilizadas pelo *dataset* estão descritas no Quadro 1.

Quadro 1 – Descrições das atividades em grupo.

| **Atividade** | **Descrição** |
|---|---|
| Em grupo | Um grupo de indivíduos está parado |
| Se aproximando | Dois grupos de indivíduos se aproximam um do outro |
| Andando junto | Um grupo de indivíduos está andando junto |
| Se encontrando | Dois grupos de indivíduos se encontram em um lugar |
| Se dividindo | Um grupo de indivíduos se divide |
| Ignorando | Um grupo ignora a movimentação de outro grupo |
| Perseguindo | Um grupo persegue outro grupo |
| Lutando | Dois grupos de indivíduos lutam um contra o outro |
| Correndo junto | Um grupo de indivíduos está correndo junto |
| Seguindo | Um grupo de indivíduos está sendo seguido por outro grupo |

Fonte: (BLUNSDEN; FISHER, 2010).

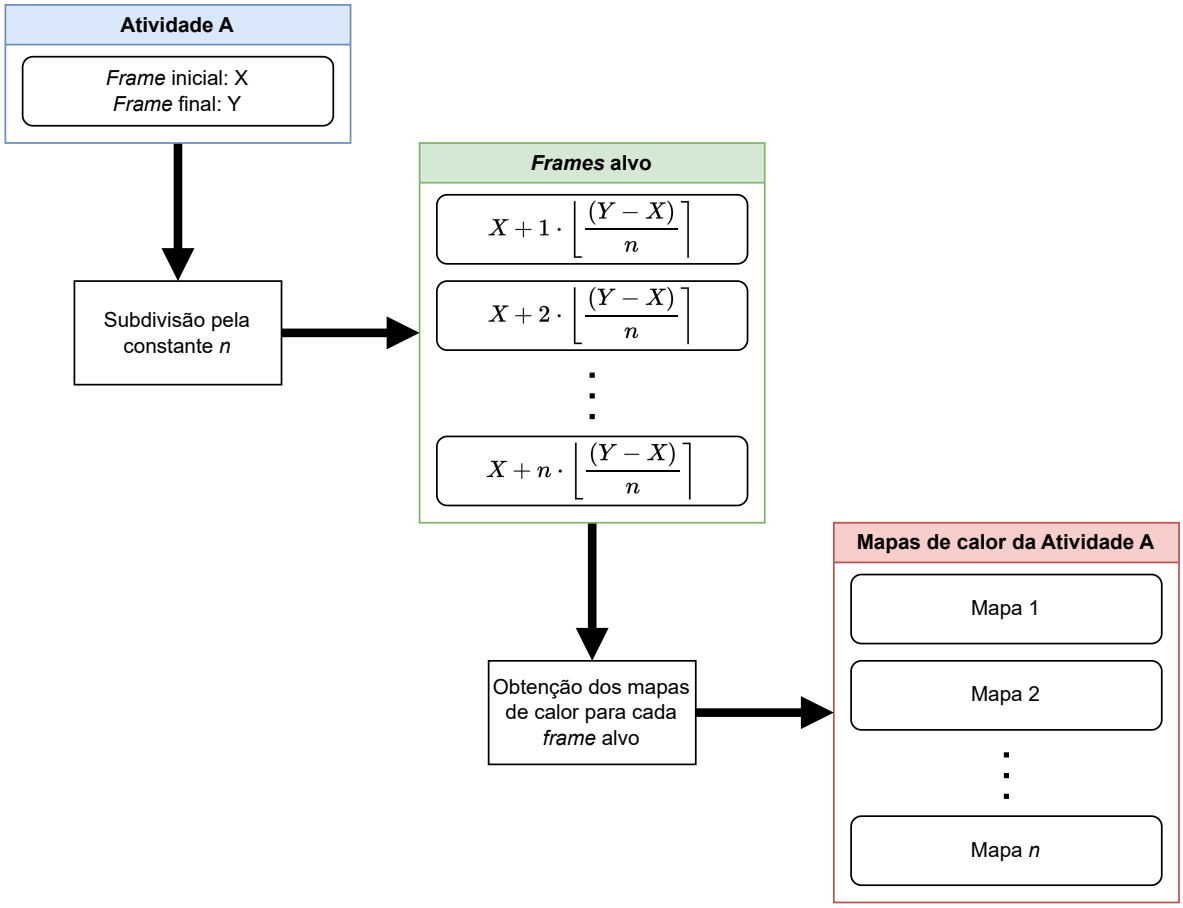
Dado que cada atividade etiquetada é representada por uma sequência de *frames*, surge a necessidade de estabelecer uma regra para decidir quais desses *frames* serão selecionados como *frames*-alvo para a geração dos mapas de calor, de modo a não poluir o *dataset* e permitir uma representação completa da atividade em grupo selecionada. Ademais, é importante reforçar que o algoritmo desenvolvido anteriormente consegue gerar um mapa de calor para um *frame* específico, considerando as atividades que ocorreram nos *frames* anteriores. Assim, visando aprimorar o posterior processo de treinamento do modelo de classificação, deve-se obter um conjunto de dados com baixa redundância e alta variabilidade de informações. Por isso, a metodologia adotada se baseia em dividir o intervalo total de uma atividade em $n$ subintervalos de igual tamanho, com cada subintervalo correspondendo a um *frame*-alvo específico, que originará um mapa de calor. A determinação do *frame*-alvo $f_i$ para cada subintervalo é realizada conforme a Equação (31), na qual $\lfloor \cdot \rceil$ representa o arredondamento para o número inteiro mais próximo, e $i$ é um número inteiro incrementado até o valor de $n$. Em tal equação, $f_{inicial}$ e $f_{final}$ correspondem, respectivamente, aos índices dos *frames* inicial e final de cada atividade.

$$f_i = f_{inicial} + i \cdot \left\lfloor \frac{f_{final} - f_{inicial}}{n} \right\rceil \tag{31}$$

A Figura 46 ilustra esse processo. Observe que após definir os *frames* inicial e final de uma atividade $A$, o intervalo é subdividido de maneira uniforme, resultando em uma lista

de *frames*-alvo para a atividade em questão. Cada *frame*-alvo passa então pelo algoritmo de geração de mapa de calor, criando as imagens que servirão para o treinamento da rede neural. Esta metodologia é necessária porque captura de maneira sucinta a evolução da movimentação dos indivíduos, detalhando as transições nos mapas de calor para a atividade que está ocorrendo nos *frames*. A Figura 47 exemplifica essa abordagem ao dividir uma atividade do tipo "se aproximando" em quatro intervalos regulares, captando o início, meio e fim da movimentação.

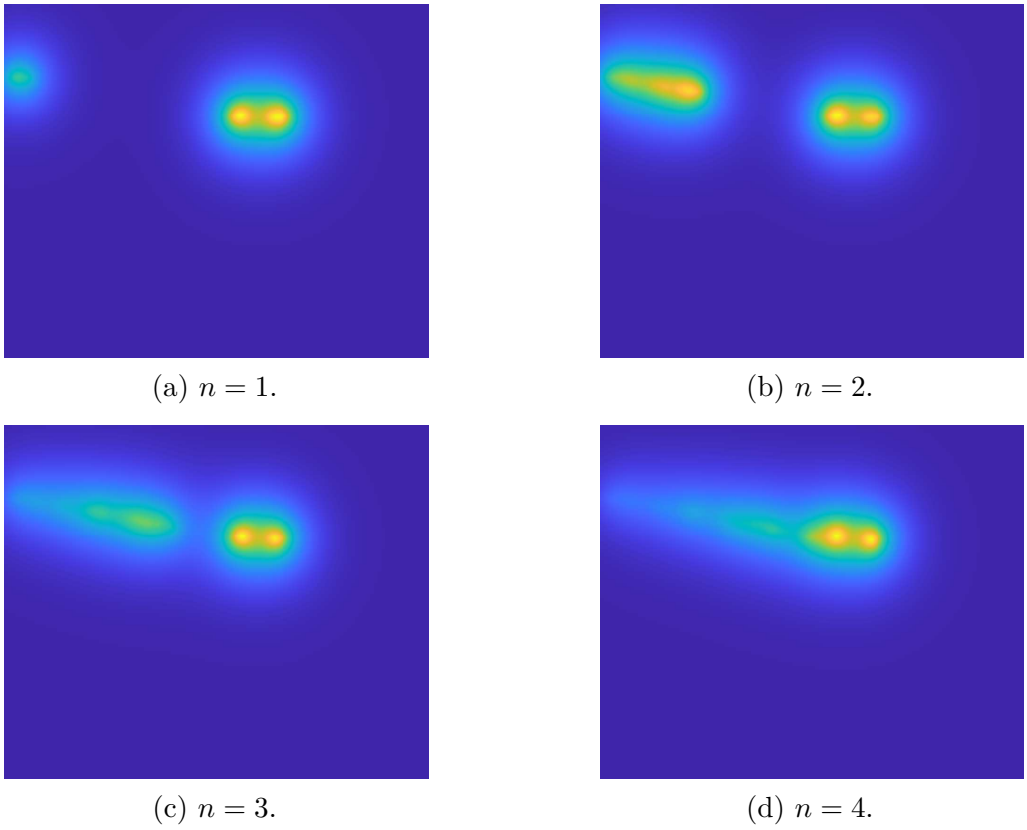
Figura 46 – Algoritmo de obtenção dos mapas de calor de uma atividade em grupo.




Fonte: elaborado pelo autor.

Após a definição da metodologia para a formação do *dataset* de atividades em grupo, emergem alguns tópicos que demandam atenção. Nota-se que algumas atividades fornecidas por (BLUNSDEN; FISHER, 2010) apresentam um número limitado de amostras, como as atividades "ignorando" e "se encontrando". Em consequência disso, com base em alguns testes prévios, decidiu-se excluir tais atividades do *dataset*, porque não fornecem um volume suficiente de dados para garantir uma classificação confiável. Além disso, identifica-se uma ambiguidade significativa em algumas atividades, particularmente em "andando junto" e "correndo junto", onde os mapas de calor gerados são muito semelhantes devido

Figura 47 – Mapas de calor dos *frames* alvo de uma atividade.



(a) $n = 1$. (b) $n = 2$.

(c) $n = 3$. (d) $n = 4$.

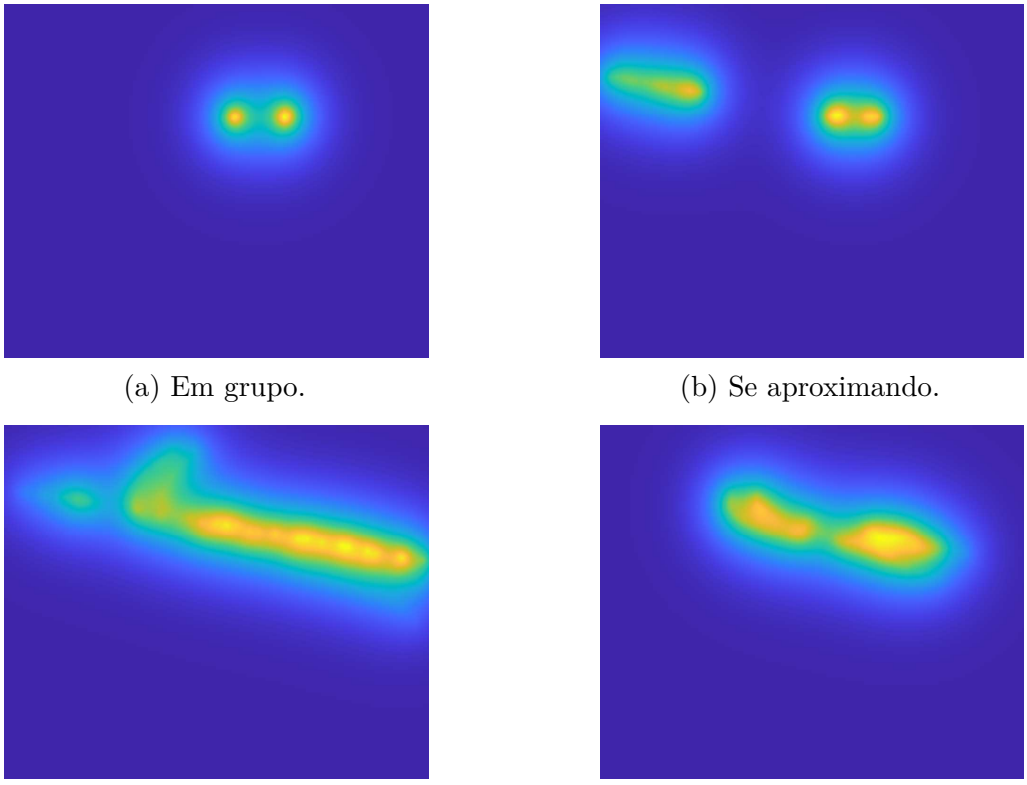
Fonte: elaborado pelo autor.

à resolução temporal adotada, dificultando a distinção precisa entre elas. Adicionalmente, a possibilidade de sobreposição de atividades, onde mais de uma ação é representada no mesmo *frame*, apresenta um desafio adicional, levando à eliminação de algumas classes para reduzir a probabilidade de erros decorrentes dessa sobreposição.

No contexto deste trabalho, conforme ilustrado na Figura 48, escolheram-se quatro classes de atividades para a classificação, a saber: “em grupo”, “se aproximando”, “andando junto” e “se dividindo”. Essa seleção melhora a precisão da classificação, priorizando atividades com diferenças mais claras em seus padrões de movimento. Posteriormente, foram gerados os mapas de calor para todas as atividades selecionadas, resultando no *dataset* inicial que foi posteriormente expandido utilizando a técnica de *Data Augmentation* (descrita na Seção 3.2.2). Contudo, antes de prosseguir para a próxima fase de processamento, realizou-se uma inspeção manual nos mapas de calor obtidos, com o propósito de excluir quaisquer anomalias, assegurando a integridade e a qualidade do *dataset* que serviu como base para o treinamento do modelo de classificação.

No contexto deste trabalho, conforme ilustrado na Figura 48, foram escolhidas quatro classes de atividades para a classificação: “em grupo”, “se aproximando”, “andando junto” e “se dividindo”. Esta seleção visou melhorar a precisão da classificação, enfatizando atividades com diferenças mais claras em seus padrões de movimento. Para a geração dos

Figura 48 – Mapas de calor das atividades selecionadas.



(a) Em grupo. (b) Se aproximando.

(c) Andando junto. (d) Se dividindo.

Fonte: elaborado pelo autor.

mapas de calor dessas atividades, adotaram-se os parâmetros: $h = 2$, $k_t = 0{,}006$, $k_e = 2$ e $k_p = 0{,}07$. Os mapas de calor resultantes formaram o *dataset* inicial, que posteriormente foi expandido utilizando a técnica de *Data Augmentation* (conforme descrito na Seção 3.2.2). Antes de prosseguir para a fase subsequente de processamento, realizou-se uma inspeção manual nos mapas de calor gerados, visando identificar e remover quaisquer anomalias. Este procedimento assegurou a integridade e a qualidade do *dataset*, essencial para o treinamento eficaz do modelo de classificação.

### 3.2.2 Criação do algoritmo de *Data Augmentation*

Ao adotar uma constante de subdivisão $n = 5$, gerou-se um *dataset* inicial composto por aproximadamente 500 mapas de calor. No entanto, constatou-se que essa quantidade não era suficiente para obter redes neurais com desempenho adequado. Uma primeira tentativa de melhoria envolveu o uso da funcionalidade de *augmented datastore* da *toolbox* do MATLAB, que proporcionou uma ligeira melhoria nos resultados, mas ainda insuficiente para alcançar resultados satisfatórios. Assim, surgiu a necessidade de desenvolver um algoritmo específico de *Data Augmentation* para gerar mapas de calor sintéticos, visando aprimorar o treinamento das redes neurais.

Este algoritmo de *Data Augmentation* utilizou predominantemente transformações

geométricas de rotação, escalamento, cisalhamento e espelhamento (conforme descrito na Seção 2.2.3), em conjunto com alteração de contraste e posterização (Seções 2.2.1.2 e 2.2.1.5).

Como estratégia adotada na etapa de Data Augmentation, definiu-se que entre a lista de operações disponíveis, nem todas eram aplicadas a cada mapa de calor original, com a seleção das operações realizada de forma aleatória, ampliando dessa forma a variabilidade do *dataset*.

No que se refere às transformações geométricas, estabeleceram-se limites máximos e mínimos para rotação, translação (horizontal e vertical) e cisalhamento (horizontal e vertical). Em cada iteração do algoritmo, um valor aleatório dentro desses intervalos era escolhido, gerando assim imagens sintéticas distintas. Para as operações de espelhamento, utilizou-se uma variável binária aleatória para definir a ocorrência ou não do espelhamento no eixo especificado. Para a alteração de contraste, a constante multiplicativa foi selecionada aleatoriamente (em um intervalo previamente definido). Já na posterização, foi escolhido aleatoriamente o número de níveis de cinza da imagem de saída (entre valores também previamente definidos).

É importante ressaltar que todos os parâmetros do algoritmo foram definidos em um intervalo razoável, assegurando que a imagem sintética gerada não se distanciasse excessivamente da imagem original, mas apenas introduzisse uma variação moderada. Por exemplo, o mapa de calor original em escala de cinza mostrado na Figura 49 pode originar diversos mapas de calor sintéticos, como ilustrado na Figura 50.

Considerando as diferenças no número de amostras entre as classes, adotaram-se fatores de aumento variados para cada atividade, a saber: um aumento de sete vezes para “se aproximando” e “em grupo”, enquanto para “se separando” e “andando junto” utilizaram-se fatores de treze e oito vezes, respectivamente. Essa abordagem uniformizou a quantidade de amostras em cada classe, evitando que a classificação de uma predomine sobre as demais.

### 3.2.3 Treinamento das redes neurais

Após a obtenção do *dataset* completo, proveniente das etapas de extração de mapas de calor e *Data Augmentation*, procedeu-se com o treinamento das redes neurais. Antes de iniciar o treinamento, contudo, foi necessário separar o *dataset* total em três conjuntos distintos: treinamento, validação e teste. A estratégia adotada envolveu a utilização das amostras sintéticas, geradas pelo algoritmo de *Data Augmentation* para o treinamento e a validação da rede. Consequentemente, o *dataset* de teste é composto majoritariamente de amostras reais, com uma quantidade limitada de amostras sintéticas, garantindo que as amostras reais não fossem processadas pela rede neural durante o treinamento.

Dentro da *Deep Learning Toolbox* do MATLAB, estabeleceu-se uma proporção de 25% para o *dataset* de validação e os 75% restantes para o treinamento. Adicionalmente,

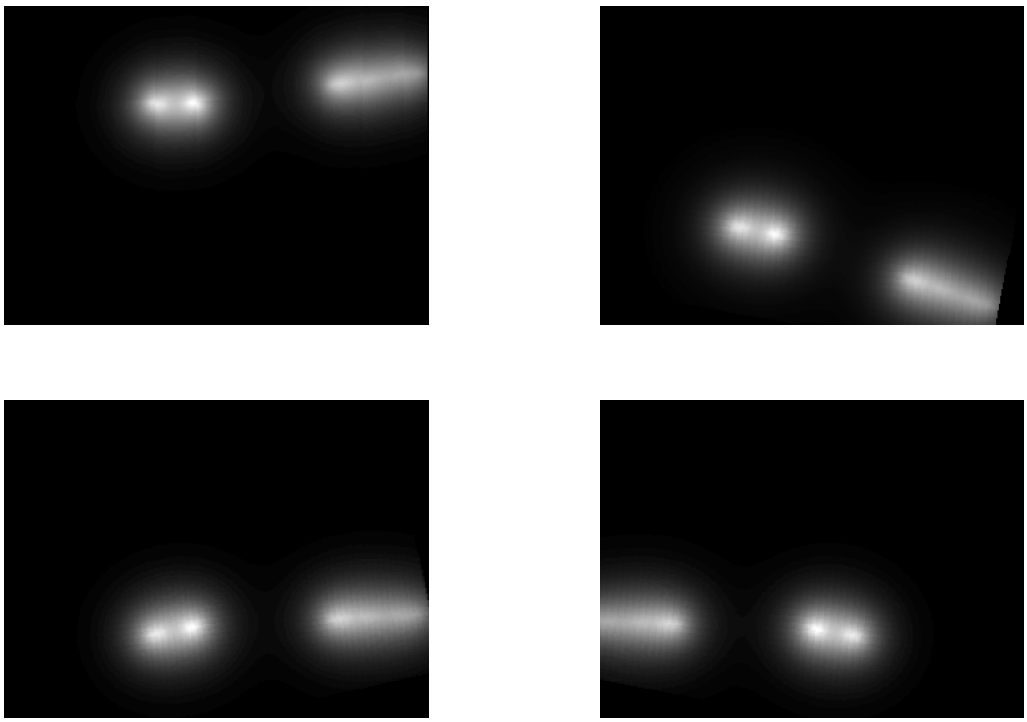
Figura 49 – Mapa de calor original.

Fonte: elaborado pelo autor.

empregou-se a ferramenta de *Data Augmentation* interna da *toolbox* para aprimorar o processo de treinamento das redes, gerando modelos de classificação mais robustos. Após a conclusão do treinamento, o modelo da rede neural pode ser exportado para uso no ambiente de desenvolvimento do MATLAB, permitindo a classificação do *dataset* de teste em uma etapa subsequente de avaliação da acurácia do modelo de classificação.

Vale destacar que a *toolbox* utilizada oferece suporte à aplicação de técnicas de *Transfer Learning* para diversas arquiteturas de redes neurais já previamente treinadas. Estas arquiteturas são exploradas de maneira mais detalhada no Capítulo 4, onde é demonstrada a eficácia de diferentes arquiteturas de redes neurais no contexto de reconhecimento de atividades em grupo. Observa-se que a organização dos *datasets* em diferentes grupos e a utilização de técnicas avançadas de treinamento e validação são fundamentais para garantir robustez e capacidade de generalização do modelo final de classificação de atividades em grupo.

Figura 50 – Mapas de calor sintéticos.



Fonte: elaborado pelo autor.

# 4 RESULTADOS

Para a classificação das atividades em grupo representadas nos mapas de calor, empregaram-se redes neurais convolucionais pré-treinadas, disponíveis na *Deep Learning Toolbox* do MATLAB. As arquiteturas selecionadas para este estudo incluem a *AlexNet*, *SqueezeNet*, *GoogLeNet*, *ResNet-18* e *ResNet-50*. Inicialmente, treinou-se cada rede com dois *datasets* distintos: o primeiro compreendendo três classes de atividades, a saber: "em grupo", "se dividindo" e "se aproximando"; e o segundo incorporando uma atividade adicional "andando junto". Desse modo, objetivou-se compreender a mudança na dinâmica com a adição de uma classe extra. Após experimentos iniciais, definiram-se os hiperparâmetros para o treinamento conforme a Tabela 1. Para o tamanho de *minibatch* e o número de épocas escolhidos, o processo de treinamento demandou aproximadamente 1500 iterações (de atualização dos parâmetros da rede) para o *dataset* com três atividades e 2100 iterações para o *dataset* com quatro atividades.

A avaliação do desempenho das redes neurais na classificação dos mapas de calor realizou-se por meio da análise da acurácia total nos conjuntos de validação e teste, bem como pelas acurácias específicas para cada tipo de atividade. Para uma avaliação ainda mais abrangente, também foram determinadas matrizes de confusão para o conjunto de teste, visando identificar quais atividades em grupo foram confundidas. Além disso, uma análise detalhada da evolução da função custo em cada processo de treinamento permitiu uma avaliação criteriosa sobre a otimização dos parâmetros gerais do sistema.

Tabela 1 – Hiperparâmetros de treinamento.

| **Hiperparâmetro** | **Valor** |
|---|---|
| *Learning rate* inicial | 0,0001 |
| Frequência de validação | 50 |
| Máximo de épocas | 300 |
| *Minibatch size* | 128 |

Fonte: elaborado pelo autor.

## 4.1 CLASSIFICAÇÃO COM 3 CLASSES DE ATIVIDADES EM GRUPO

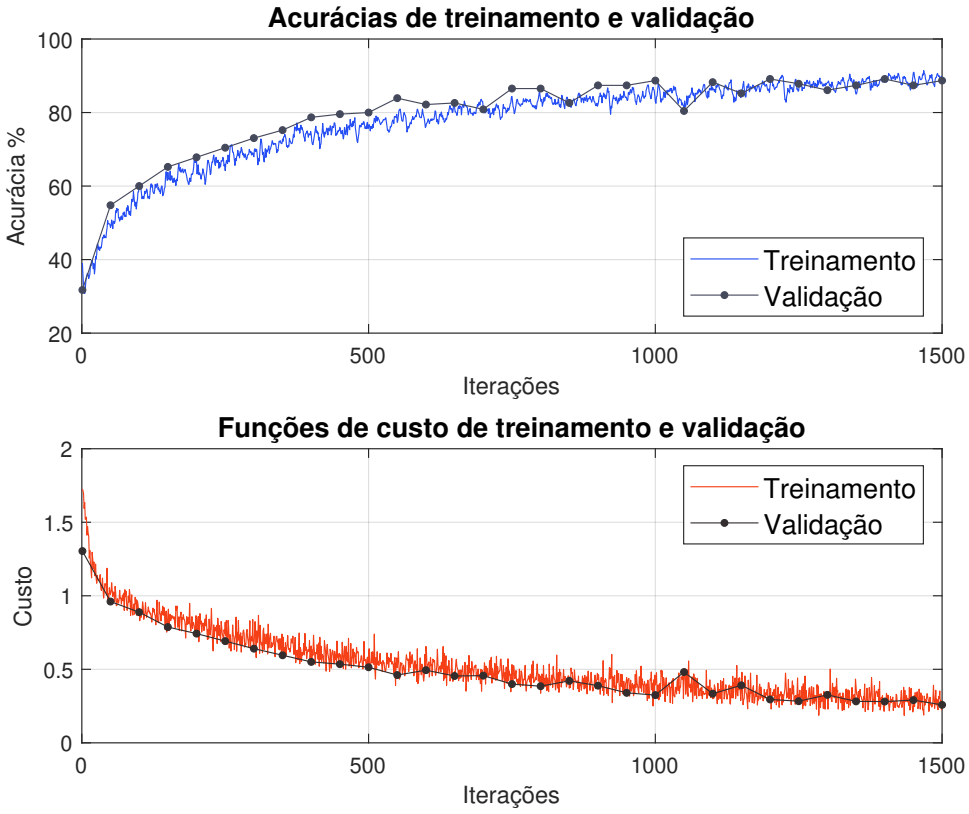
Nesta seção são apresentados os gráficos e métricas de desempenho originados dos processos de treinamento e classificação com o *dataset* que possui 3 classes de atividades em grupo.

### 4.1.1 *AlexNet*

Observando a Figura 51, pode-se notar que durante o processo de treinamento da *AlexNet* as acurácias dos conjuntos de treinamento e validação mantêm-se bastante próximas entre si, com a acurácia de validação apresentando um ligeiro aumento nas

iterações iniciais. Em relação à evolução da função custo, como ilustrado na mesma figura, é perceptível que ocorre alguma flutuação entre as iterações, mas ela permanece em uma faixa aceitável, não possuindo deformações acentuadas que invalidem o processo de treinamento.

Figura 51 – Curvas de treinamento e validação com 3 classes de atividades em grupo para a *AlexNet*.




Fonte: elaborado pelo autor.

Ao analisar as métricas de desempenho destacadas na Tabela 2, revela-se um decréscimo de aproximadamente 4,3% na acurácia final do conjunto de validação em comparação com a acurácia final do conjunto de teste. Além disso, vale ressaltar que, entre as três atividades em grupo avaliadas, apenas a classificação da atividade “se aproximando” apresenta uma acurácia final inferior a 80%, embora ainda esteja próxima desse limite e não denote uma redução substancial na acurácia da classificação. A análise da matriz de confusão, conforme indicada na Tabela 3, também corrobora essas conclusões, demonstrando que a *AlexNet*, no contexto desta aplicação específica, só apresenta uma confusão significativa ao classificar erroneamente atividades do tipo “se aproximando” como atividades do tipo “em grupo” e “se dividindo”.

Tabela 2 – Métricas de desempenho para a *AlexNet* com 3 classes de atividades em grupo.

| **Métricas** | **Valores** |
|---|---|
| Acurácia final de validação | 88,70% |
| Custo final de validação | 0,2581 |
| Acurácia final de teste | 84,46% |
| Acurácia “em grupo” | 92,47% |
| Acurácia “se aproximando” | 78,02% |
| Acurácia “se dividindo” | 82,09% |

Fonte: elaborado pelo autor.

Tabela 3 – Matriz de confusão para a *AlexNet* com 3 classes de atividades em grupo (conjunto de teste).

| | | **Atividades preditas** | | |
|---|---|---|---|---|
| | | **Em grupo** | **Se aproximando** | **Se dividindo** |
| **Atividades reais** | **Em grupo** | 172 | 13 | 1 |
| | **Se aproximando** | 21 | 142 | 19 |
| | **Se dividindo** | 9 | 15 | 110 |

Fonte: elaborado pelo autor.

### 4.1.2 *SqueezeNet*

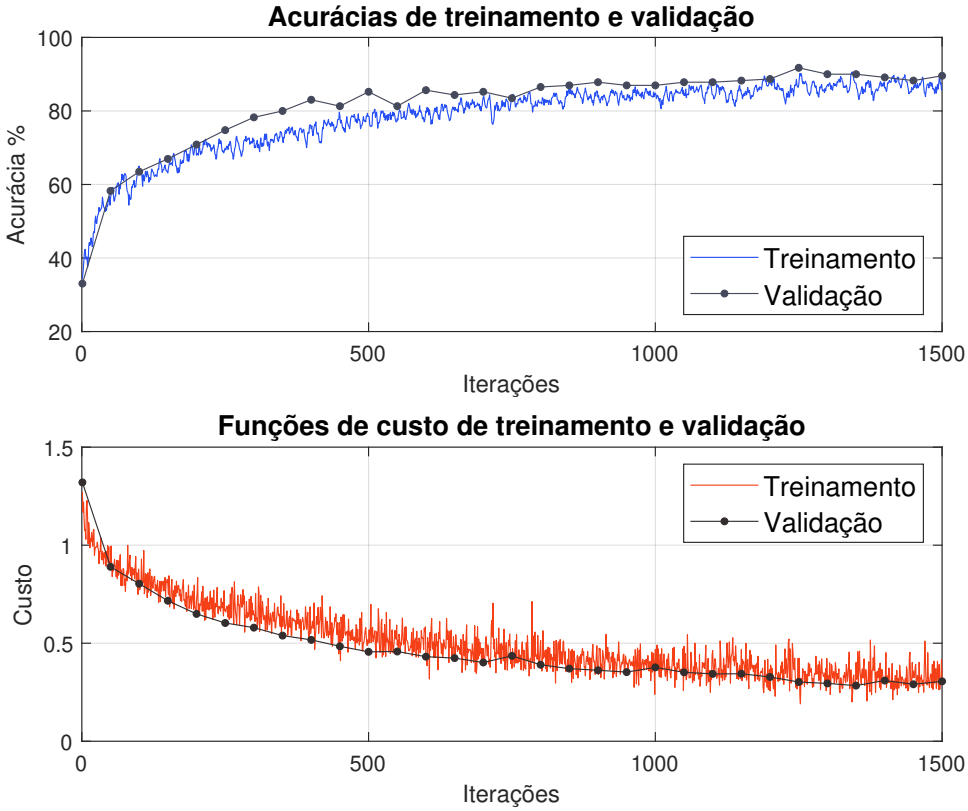
Os resultados obtidos com a utilização da *SqueezeNet* para classificação de atividades em grupo, conforme apresentados na Figura 52, destacam que a acurácia do conjunto de validação superou consistentemente a acurácia do conjunto de treinamento ao longo de quase todo o processo. Tal fato sugere não haver *overfitting* durante o processo de treinamento. Adicionalmente, a evolução temporal da função de custo de treinamento exibiu uma variação acentuada quando comparada com a *AlexNet*. Como evidenciado na Tabela 4, a função de custo finalizou em um valor ligeiramente maior.

Tabela 4 – Métricas de desempenho para a *SqueezeNet* com 3 classes de atividades em grupo.

| **Métricas** | **Valores** |
|---|---|
| Acurácia final de validação | 89,57% |
| Custo final de validação | 0,3053 |
| Acurácia final de teste | 86,45% |
| Acurácia “em grupo” | 93,01% |
| Acurácia “se aproximando” | 75,82% |
| Acurácia “se dividindo” | 91,79% |

Fonte: elaborado pelo autor.

Figura 52 – Curvas de treinamento e validação com 3 classes de atividades para a *SqueezeNet*.




Fonte: elaborado pelo autor.

Quanto às métricas de desempenho, a *SqueezeNet* alcançou resultados ótimos, com a acurácia de validação e teste ultrapassando os 85%. Entretanto, ao analisar na Tabela 4, nota-se que o desempenho de classificação da atividade "se aproximando", apresenta uma acurácia inferior em comparação às demais atividades. Esta discrepância, mais acentuada do que em análises anteriores, permite questionamentos sobre a sensibilidade do modelo a certas dinâmicas de grupo. Apesar dessa variação, a Tabela 5 revela que não ocorreram confusões significativas na classificação. A matriz de confusão demonstra que a maioria das classificações se concentrou corretamente na diagonal, indicando uma alta taxa de acertos nas classificações efetuadas pela *SqueezeNet*.

Tabela 5 – Matriz de confusão para a *SqueezeNet* com 3 classes de atividades em grupo.

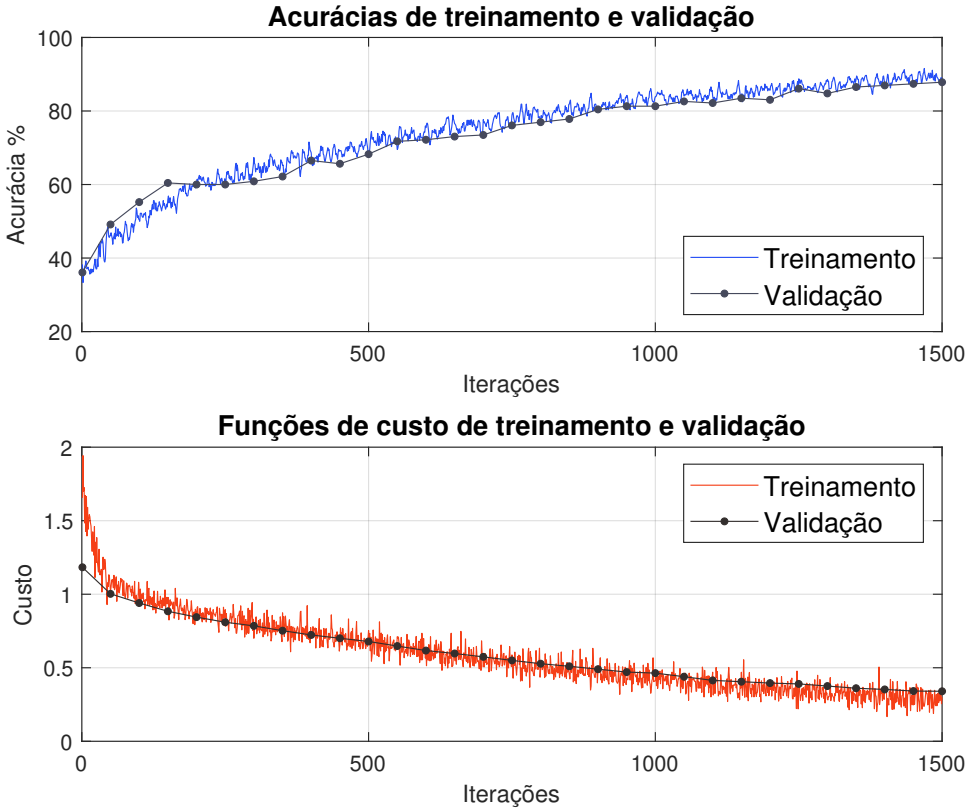
| | | **Atividades preditas** | | |
|---|---|---|---|---|
| | | **Em grupo** | **Se aproximando** | **Se dividindo** |
| **Atividades reais** | **Em grupo** | 173 | 10 | 3 |
| | **Se aproximando** | 26 | 138 | 18 |
| | **Se dividindo** | 4 | 7 | 123 |

Fonte: elaborado pelo autor.

### 4.1.3 *GoogLeNet*

Conforme ilustrado na Figura 53, os gráficos referentes ao processo de treinamento da *GoogLeNet* mostram-se um pouco mais estáveis e em comparação à *AlexNet* e à *SqueezeNet*. Em relação às curvas de acurácia, estas mantiveram-se consistentes, registrando valores finais superiores a 80%, uma observação corroborada pela Tabela 6. Esse resultado demonstra a eficácia da *GoogLeNet* na tarefa de classificação proposta.

Figura 53 – Curvas de treinamento e validação com 3 classes de atividades para a *GoogLeNet*.




Fonte: elaborado pelo autor.

Tabela 6 – Métricas de desempenho para a *GoogLeNet* com 3 classes de atividades em grupo.

| **Métricas** | **Valores** |
|---|---|
| Acurácia final de validação | 87,83% |
| Custo final de validação | 0,3401 |
| Acurácia final de teste | 83,07% |
| Acurácia “em grupo” | 93,01% |
| Acurácia “se aproximando” | 74,18% |
| Acurácia “se dividindo” | 81,34% |

Fonte: elaborado pelo autor.

Conforme os resultados apresentados na Tabela 6, tem-se que a atividade "se aproximando" foi a classificada com a menor acurácia no conjunto de teste, uma tendência semelhante observada nas outras redes. Além disso, a matriz de confusão, apresentada na Tabela 7, exibe um padrão análogo ao das matrizes anteriores. Embora haja uma confusão um pouco mais acentuada na atividade "se aproximando", a matriz ainda revela uma assertividade considerável nas demais classificações, reforçando a competência da *GoogLeNet* em discernir entre as três atividades em grupo analisadas.

Tabela 7 – Matriz de confusão para a *GoogLeNet* com 3 classes de atividades em grupo.

| | | **Atividades preditas** | | |
|---|---|---|---|---|
| | | **Em grupo** | **Se aproximando** | **Se dividindo** |
| **Atividades reais** | **Em grupo** | 173 | 11 | 2 |
| | **Se aproximando** | 27 | 135 | 20 |
| | **Se dividindo** | 8 | 17 | 109 |

Fonte: elaborado pelo autor.

### 4.1.4 *ResNet-18*

Observa-se, através da Figura 54, que as curvas de treinamento da *ResNet-18* para a classificação de mapas de calor de atividades em grupo exibem comportamentos similares aos das outras redes neurais avaliadas. Esta similaridade se estende às métricas de desempenho, conforme indicado na Tabela 8. Especificamente, constata-se que a acurácia de validação e a função custo da *ResNet-18* apresentam, respectivamente, uma eficácia satisfatória e um baixo erro de classificação total.

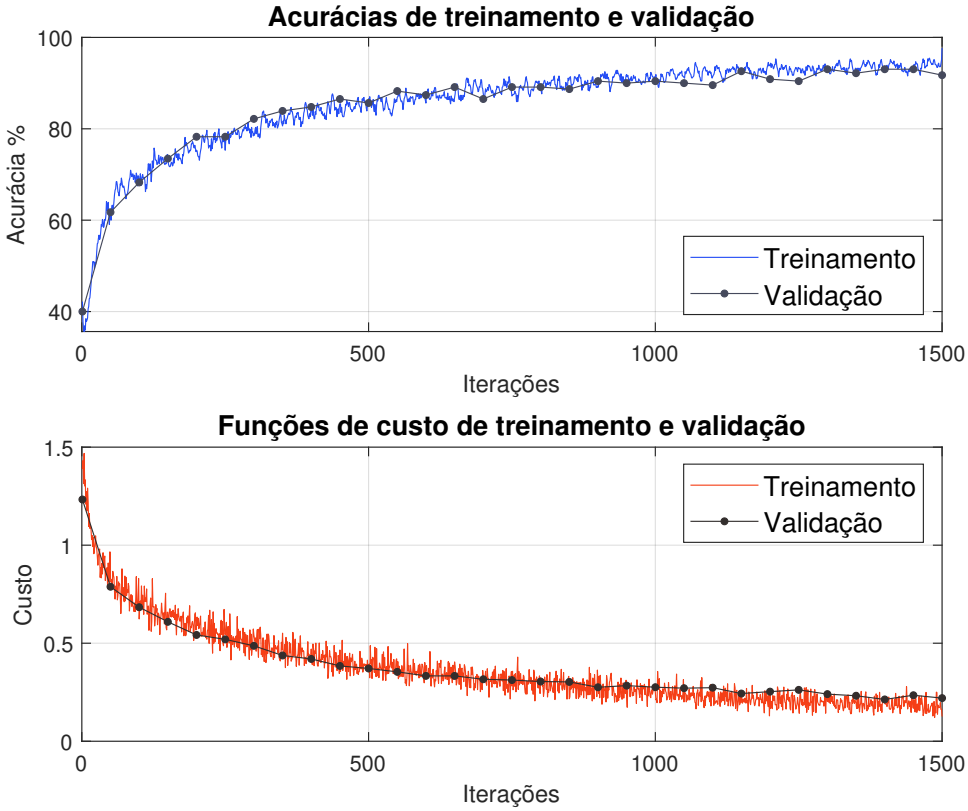
Tabela 8 – Métricas de desempenho para a *ResNet-18* com 3 classes de atividades em grupo.

| **Métricas** | **Valores** |
|---|---|
| Acurácia final de validação | 88,26% |
| Custo final de validação | 0,2491 |
| Acurácia final de teste | 76,10% |
| Acurácia "em grupo" | 51,61% |
| Acurácia "se aproximando" | 98,35% |
| Acurácia "se dividindo" | 79,85% |

Fonte: elaborado pelo autor.

No entanto, ao analisar a acurácia final no conjunto de teste, percebe-se uma queda notável de 12,16% em comparação à acurácia de validação, sugerindo uma possível limitação na generalização da *ResNet-18* no contexto específico de classificação de mapas

Figura 54 – Curvas de treinamento e validação com 3 classes de atividades para a *ResNet-18*.




Fonte: elaborado pelo autor.

de calor. Além disso, verifica-se que as acurácias das atividades variam significativamente, destacando-se a atividade "em grupo" com um valor de apenas 51,61%. Este resultado é consideravelmente baixo e poderia implicar em desafios na implementação prática do modelo. A Tabela 9 ilustra a causa dessa baixa acurácia: o classificador inverte a tendência de confusão das redes anteriores, frequentemente categorizando incorretamente atividades do tipo "em grupo" como "se aproximando", o que evidencia dificuldades na diferenciação destas categorias pelo modelo.

### 4.1.5 *ResNet-50*

Durante o treinamento da *ResNet-50* para classificação de mapas de calor de atividades em grupo, identificou-se uma possível ocorrência de *overfitting*. Esta constatação baseia-se na observação de que a curva de acurácia de validação permaneceu consistentemente abaixo da acurácia de treinamento ao longo de todo o processo, conforme demonstrado na Figura 55. Apesar disso, a *ResNet-50* alcançou acurácias finais para os conjuntos de validação e teste superiores às obtidas pela *ResNet-18*, como evidenciado na Tabela 10. Além disso, se observa que a classificação para atividades do tipo "em grupo"

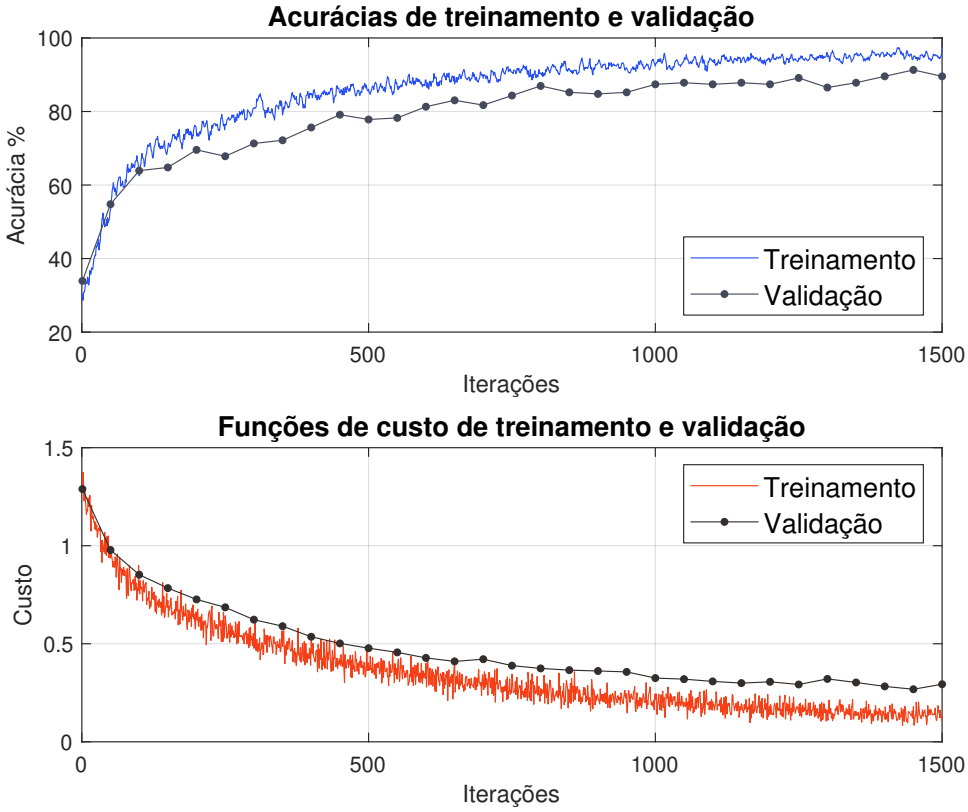
Tabela 9 – Matriz de confusão para a *ResNet-18* com 3 classes de atividades em grupo.

| | | **Atividades preditas** | | |
|---|---|---|---|---|
| | | **Em grupo** | **Se aproximando** | **Se dividindo** |
| **Atividades reais** | **Em grupo** | 96 | 80 | 10 |
| | **Se aproximando** | 1 | 179 | 2 |
| | **Se dividindo** | 2 | 25 | 107 |

Fonte: elaborado pelo autor.

apresentou uma acurácia relativamente baixa para o conjunto de teste, não ultrapassando 70%. Adicionalmente, embora em menor grau, a Tabela 11 revela que persiste uma confusão na distinção entre as atividades "em grupo" e "se aproximando", indicando uma dificuldade da arquitetura *ResNet* em diferenciar estas atividades.

Figura 55 – Curvas de treinamento e validação com 3 classes de atividades para a *ResNet-50*.




Fonte: elaborado pelo autor.

Tabela 10 – Métricas de desempenho para a *ResNet-50* com 3 classes de atividades em grupo.

| **Métricas** | **Valores** |
|---|---|
| Acurácia final de validação | 92,17% |
| Custo final de validação | 0,2264 |
| Acurácia final de teste | 80,68% |
| Acurácia “em grupo” | 67,74% |
| Acurácia “se aproximando” | 95,60% |
| Acurácia “se dividindo” | 78,36% |

Fonte: elaborado pelo autor.

Tabela 11 – Matriz de confusão para a *ResNet-50* com 3 classes de atividades em grupo.

| | | **Atividades preditas** | | |
|---|---|---|---|---|
| | | **Em grupo** | **Se aproximando** | **Se dividindo** |
| **Atividades reais** | **Em grupo** | 126 | 59 | 1 |
| | **Se aproximando** | 5 | 174 | 3 |
| | **Se dividindo** | 3 | 26 | 105 |

Fonte: elaborado pelo autor.

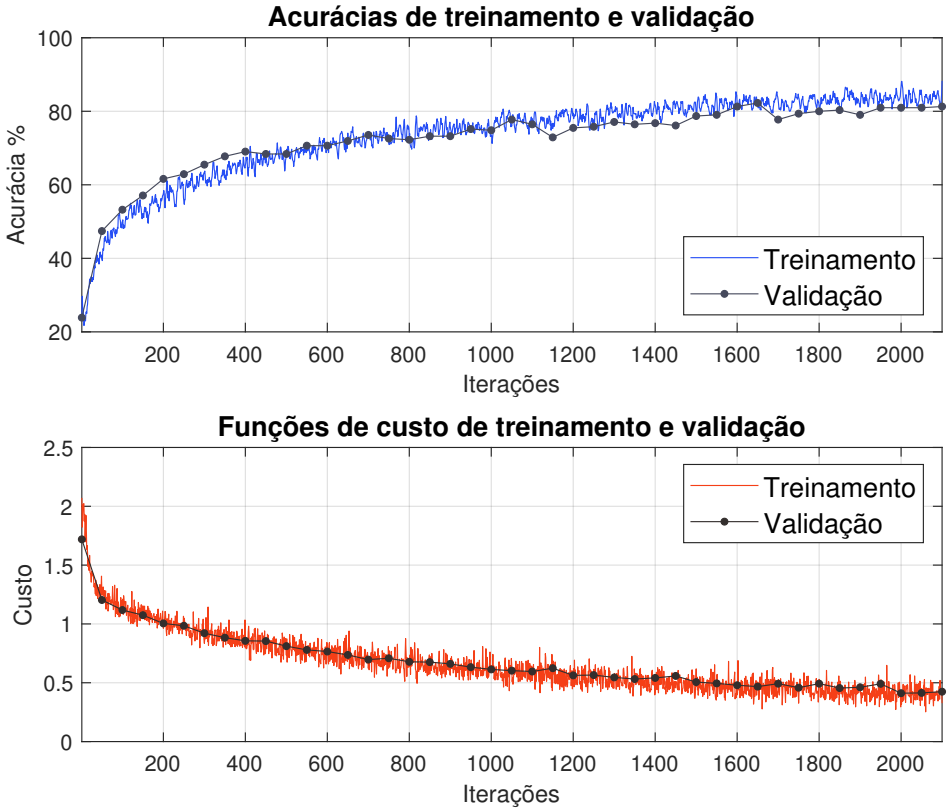
## 4.2 CLASSIFICAÇÃO COM 4 CLASSES DE ATIVIDADES EM GRUPO

Nesta seção são discutidos os resultados da etapa de treinamento e de classificação das redes neurais com o *dataset* que possui 4 classes de atividades em grupo.

### 4.2.1 *AlexNet*

Ao treinar a *AlexNet* com um *dataset* composto por 4 classes de atividades em grupo, observou-se o início de *overfitting*, especialmente ao analisar os pontos-finais da curva de acurácia para o conjunto de validação, conforme ilustrado na Figura 56. Nesta figura, nota-se uma discrepância entre as curvas de treinamento e validação, embora seja uma diferença relativamente pequena. Ademais, um aumento na função custo (em relação ao treinamento anterior com apenas 3 classes de atividades) também é notado, como evidenciado na Tabela 12, onde o custo final alcançou 0,4239. Este valor é sensivelmente mais elevado em comparação com a métrica correspondente no treinamento que utilizou o *dataset* com apenas 3 classes de atividades, sugerindo um aumento na complexidade do problema com a inclusão de uma atividade adicional.

Figura 56 – Curvas de treinamento e validação com 4 classes de atividades para a *AlexNet*.




Fonte: elaborado pelo autor.

Tabela 12 – Métricas de desempenho para a *AlexNet* com 4 classes de atividades em grupo.

| **Métricas** | **Valores** |
|---|---|
| Acurácia final de validação | 81,29% |
| Custo final de validação | 0,4239 |
| Acurácia final de teste | 76,96% |
| Acurácia "em grupo" | 93,55% |
| Acurácia "se aproximando" | 62,09% |
| Acurácia "se dividindo" | 76,87% |
| Acurácia "andando junto" | 74,69% |

Fonte: elaborado pelo autor.

Os resultados indicam também que a *AlexNet* enfrenta maiores dificuldades para classificar atividades do tipo "se aproximando", alcançando uma acurácia de apenas 62,09%. Ao examinar a matriz de confusão na Tabela 13, constata-se que essa baixa acurácia deriva da confusão dessa atividade com as atividades "se dividindo" e "andando junto". Essa confusão nas classificações contribui significativamente para a redução da acurácia geral, ressaltando os desafios enfrentados pela arquitetura da *AlexNet* em diferenciar com precisão entre esses tipos específicos essas classes de atividades em grupo.

Tabela 13 – Matriz de confusão para a *AlexNet* com 4 classes de atividades em grupo.

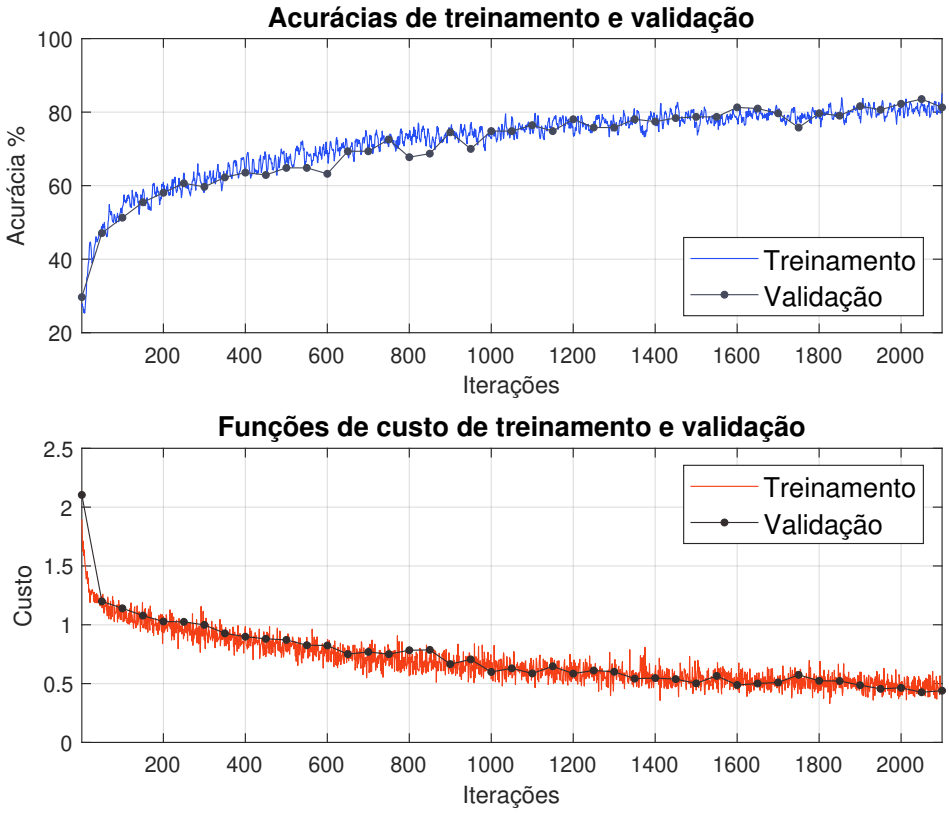
| | | **Atividades preditas** | | | |
|---|---|---|---|---|---|
| | | **Em grupo** | **Se aproximando** | **Se dividindo** | **Andando junto** |
| **Atividades reais** | **Em grupo** | 174 | 9 | 2 | 1 |
| | **Se aproximando** | 15 | 113 | 26 | 28 |
| | **Se dividindo** | 9 | 7 | 103 | 15 |
| | **Andando junto** | 10 | 9 | 22 | 121 |

Fonte: elaborado pelo autor.

### 4.2.2 *SqueezeNet*

A análise dos resultados obtidos com a *SqueezeNet* para a classificação de mapas de calor de 4 classes de atividades em grupo revela um desempenho muito similar ao da *AlexNet*, porém com uma vantagem: a ausência de indícios de *overfitting*, conforme demonstrado na Figura 57. Esta figura também indica um aumento na função custo em comparação ao treinamento realizado com o *dataset* que contém 3 classes de atividades em grupo. As métricas de desempenho, como apresentadas na Tabela 14, são comparáveis às da *AlexNet*, com a *SqueezeNet* atingindo uma acurácia total no conjunto de teste ligeiramente superior. Entretanto, enfrenta-se novamente um desafio na classificação das atividades do tipo "se aproximando" no conjunto de teste, que, conforme a Tabela 15, são agora mais frequentemente confundidas com as atividades "em grupo" e "andando junto".

Figura 57 – Curvas de treinamento e validação com 4 classes de atividades para a *SqueezeNet*.




Fonte: elaborado pelo autor.

Tabela 14 – Métricas de desempenho para a SqueezeNet com 4 classes de atividades em grupo.

| **Métricas** | **Valores** |
|---|---|
| Acurácia final de validação | 81,29% |
| Custo final de validação | 0,4397 |
| Acurácia final de teste | 79,82% |
| Acurácia “em grupo” | 91,40% |
| Acurácia “se aproximando” | 63,74% |
| Acurácia “se dividindo” | 79,85% |
| Acurácia “andando junto” | 84,57% |

Fonte: elaborado pelo autor.

Tabela 15 – Matriz de confusão para a *SqueezeNet* com 4 classes de atividades em grupo.

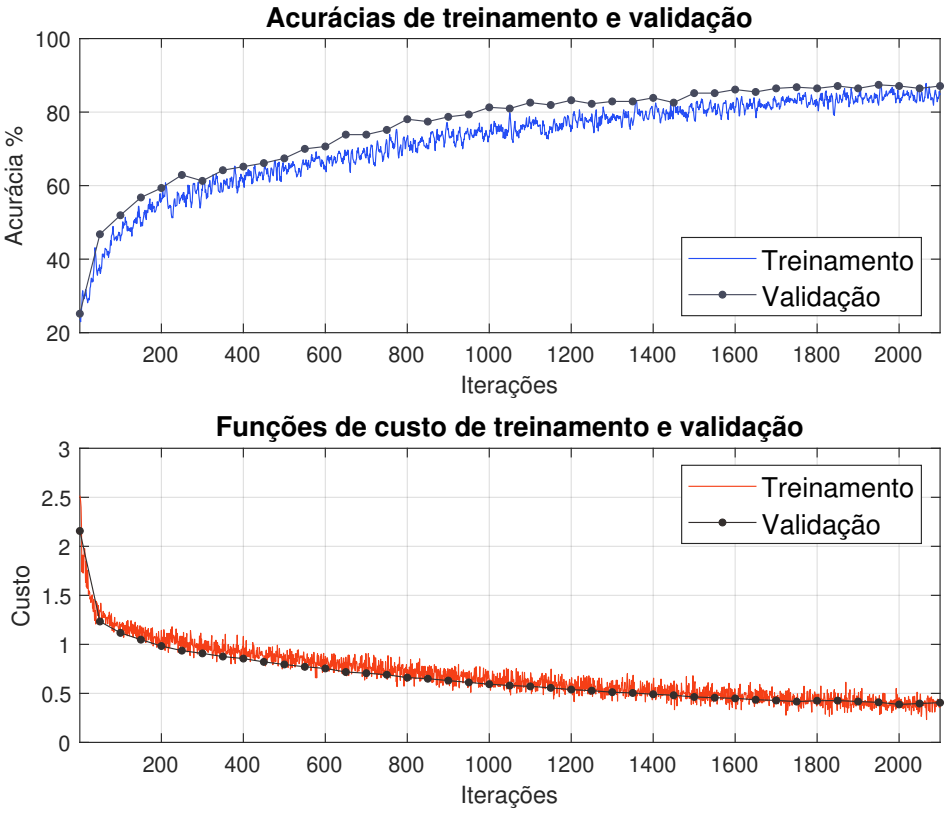
| | | **Atividades preditas** | | | |
|---|---|---|---|---|---|
| | | **Em grupo** | **Se aproximando** | **Se dividindo** | **Andando junto** |
| **Atividades reais** | **Em grupo** | 170 | 11 | 2 | 3 |
| | **Se aproximando** | 20 | 116 | 15 | 31 |
| | **Se dividindo** | 3 | 7 | 107 | 17 |
| | **Andando junto** | 1 | 12 | 12 | 137 |

Fonte: elaborado pelo autor.

### 4.2.3 *GoogLeNet*

Para a rede GoogLeNet treinada com o *dataset* que contém 4 classes de atividades em grupo, observa-se conforme ilustrado na Figura 58 que a acurácia do conjunto de validação supera a do conjunto de treinamento, um indicativo de boa generalização do modelo. As métricas de desempenho, detalhadas na Tabela 16, indicam que a acurácia final no conjunto de teste ultrapassa 80%. No entanto, a atividade "se aproximando" continua apresentando uma acurácia reduzida. Ademais, a Tabela 17 revela uma confusão consistente entre as atividades "se aproximando" e "andando junto", uma tendência já observada nas redes neurais anteriores, evidenciando a dificuldade na classificação dessas atividades.

Figura 58 – Curvas de treinamento e validação com 4 classes de atividades para a *GoogLeNet*.




Fonte: elaborado pelo autor.

Tabela 16 – Métricas de desempenho para a *GoogLeNet* com 4 classes de atividades em grupo.

| **Métricas** | **Valores** |
|---|---|
| Acurácia final de validação | 87,10% |
| Custo final de validação | 0,4050 |
| Acurácia final de teste | 80,72% |
| Acurácia “em grupo” | 93,01% |
| Acurácia “se aproximando” | 64,29% |
| Acurácia “se dividindo” | 82,09% |
| Acurácia “andando junto” | 83,95% |

Fonte: elaborado pelo autor.

Tabela 17 – Matriz de confusão para a *GoogLeNet* com 4 classes de atividades em grupo.

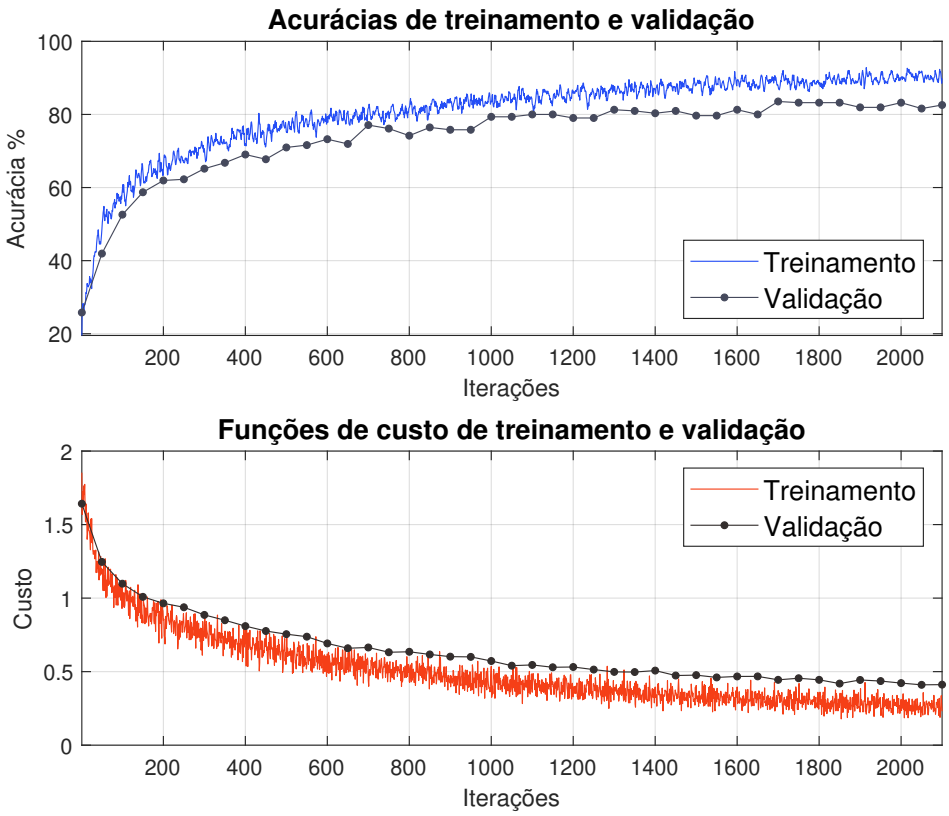
| | | **Atividades preditas** | | | |
|---|---|---|---|---|---|
| | | **Em grupo** | **Se aproximando** | **Se dividindo** | **Andando junto** |
| **Atividades reais** | **Em grupo** | 173 | 10 | 1 | 2 |
| | **Se aproximando** | 18 | 117 | 16 | 31 |
| | **Se dividindo** | 3 | 5 | 110 | 16 |
| | **Andando junto** | 0 | 12 | 14 | 136 |

Fonte: elaborado pelo autor.

### 4.2.4 *ResNet-18*

Na aplicação da *ResNet-18* para a classificação das 4 classes de atividades em grupo, observou-se a tendência ao *overfitting*, uma característica já identificada no uso dessa arquitetura para a classificação de 3 classes de atividades em grupo. Essa inclinação ao *overfitting* é evidenciada pela discrepância nas curvas de treinamento e validação, como ilustra a Figura 59. A menor acurácia e o maior custo no conjunto de validação, em comparação ao conjunto de treinamento, sugerem que a rede neural está memorizando os mapas de calor do conjunto de treinamento ao invés de aprender suas características essenciais.

Figura 59 – Curvas de treinamento e validação com 4 classes de atividades para a *ResNet-18*.




Fonte: elaborado pelo autor.

Apesar da presença de *overfitting*, as métricas de desempenho, descritas na Tabela 18, apresentam resultados promissores em comparação com outras redes neurais. Embora a acurácia final no conjunto de teste esteja abaixo de 80%, todas as atividades em grupo registraram acurácias individuais superiores a 70%, um contraste em relação às demais arquiteturas. A Tabela 19 revela que a maior confusão na classificação ocorre com atividades do tipo “em grupo”, sendo frequentemente classificada erroneamente como “se aproximando”.

Tabela 18 – Métricas de desempenho para a *ResNet-18* com 4 classes de atividades em grupo.

| **Métricas** | **Valores** |
|---|---|
| Acurácia final de validação | 87,10% |
| Custo final de validação | 0,3491 |
| Acurácia final de teste | 77,86% |
| Acurácia “em grupo” | 72,04% |
| Acurácia “se aproximando” | 82,42% |
| Acurácia “se dividindo” | 75,37% |
| Acurácia “andando junto” | 81,48% |

Fonte: elaborado pelo autor.

Tabela 19 – Matriz de confusão para a *ResNet-18* com 4 classes de atividades em grupo.

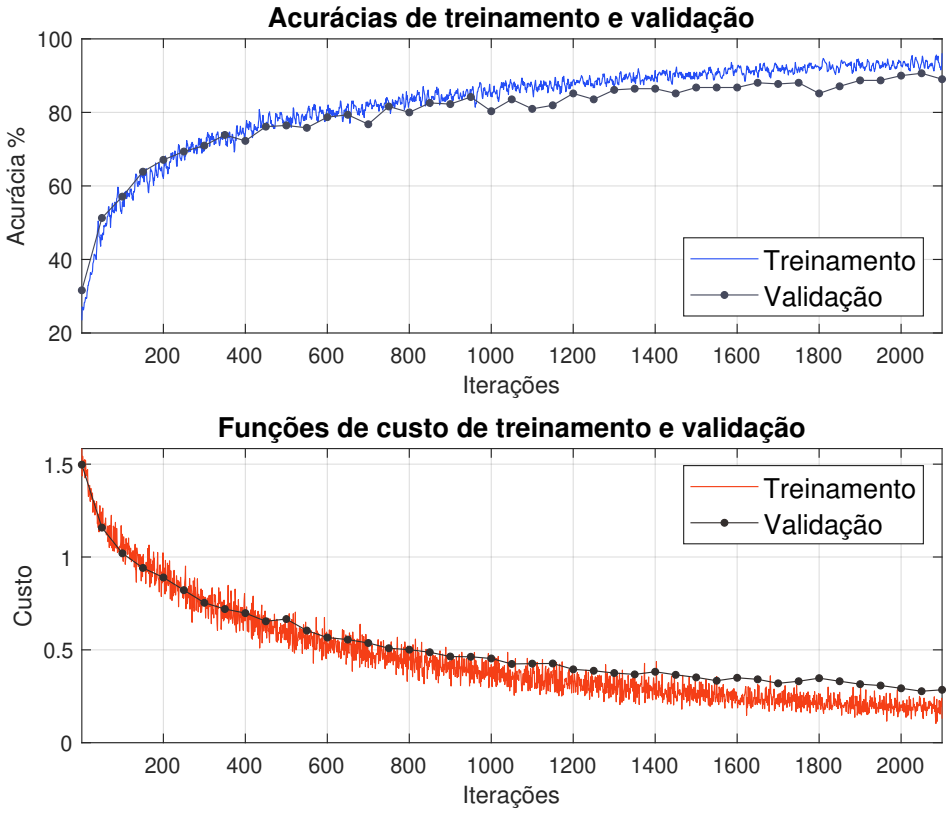
| | | **Atividades preditas** | | | |
|---|---|---|---|---|---|
| | | **Em grupo** | **Se aproximando** | **Se dividindo** | **Andando junto** |
| **Atividades reais** | **Em grupo** | 134 | 47 | 3 | 2 |
| | **Se aproximando** | 5 | 150 | 1 | 26 |
| | **Se dividindo** | 2 | 19 | 101 | 12 |
| | **Andando junto** | 2 | 14 | 14 | 132 |

Fonte: elaborado pelo autor.

### 4.2.5 *ResNet-50*

Ao analisar os resultados do treinamento e classificação com a *ResNet-50* para mapas de calor de 4 classes de atividades em grupo, constata-se, através da Figura 60, uma menor incidência de *overfitting* em comparação à *ResNet-18*. Além disso, a Tabela 20 indica que a acurácia final no conjunto de teste da *ResNet-50* supera a da *ResNet-18*, também contando com todas as atividades em grupo alcançando acurácias individuais acima de 70%

Figura 60 – Curvas de treinamento e validação com 4 classes de atividades para a *ResNet-50*.




Fonte: elaborado pelo autor.

Outro aspecto relevante é que a função custo durante o processo de treinamento da *ResNet-50* atinge o valor final mais baixo para a classificação das 4 classes de atividades em grupo, aproximando-se do valor alcançado no treinamento com o *dataset* que contém 3 classes de atividades. A Tabela 21, assim como a Tabela 19, revela uma maior confusão na classificação das atividades do tipo "em grupo", as quais são erroneamente categorizadas como "se aproximando" em alguns casos. Esta tendência sinaliza um desafio na diferenciação precisa dessas categorias específicas de atividades em grupo pela arquitetura *ResNet*.

Tabela 20 – Métricas de desempenho para a *ResNet-50* com 4 classes de atividades em grupo.

| **Métricas** | **Valores** |
|---|---|
| Acurácia final de validação | 86,77% |
| Custo final de validação | 0,2464 |
| Acurácia final de teste | 82,38% |
| Acurácia “em grupo” | 70,43% |
| Acurácia “se aproximando” | 88,46% |
| Acurácia “se dividindo” | 92,54% |
| Acurácia “andando junto” | 80,86% |

Fonte: elaborado pelo autor.

Tabela 21 – Matriz de confusão para a *ResNet-50* com 4 classes de atividades em grupo.

| | | **Atividades preditas** | | | |
|---|---|---|---|---|---|
| | | **Em grupo** | **Se aproximando** | **Se dividindo** | **Andando junto** |
| **Atividades reais** | **Em grupo** | 131 | 53 | 0 | 2 |
| | **Se aproximando** | 4 | 161 | 5 | 12 |
| | **Se dividindo** | 0 | 4 | 124 | 6 |
| | **Andando junto** | 0 | 19 | 12 | 131 |

Fonte: elaborado pelo autor.

## 4.3 ANÁLISE DOS RESULTADOS

A análise dos resultados das classificações obtidas nas diversas arquiteturas de redes neurais empregadas revela tendências importantes. Especificamente, na arquitetura *ResNet*, observa-se um aumento no fenômeno de *overfitting*, especialmente quando o treinamento é realizado com 4 classes de atividades de grupo. Apesar disso, as variantes da *ResNet* demonstraram uma acurácia superior no conjunto de teste, um indicativo de sua eficácia na tarefa de classificação. No entanto, um desafio persistente na *ResNet* é a classificação de atividades do tipo "em grupo", onde a acurácia foi consistentemente menor em relação as outras classes de atividades em grupo. Além disso, para todas as arquiteturas é observado um incremento no valor final da função custo ao passar da classificação de 3 para 4 atividades, sinalizando uma complexidade adicional na tarefa.

No *ranking* de acurácias para o conjunto de teste com 3 classes de atividades em grupo, conforme apresentado na Tabela 22, a *SqueezeNet* obteve a melhor desempenho, alcançando uma acurácia de 86,45%. Este resultado destaca a eficiência da *SqueezeNet* em termos de parâmetros e capacidade computacional, sugerindo que redes mais leves podem ser eficazes em tarefas específicas. A *AlexNet*, com uma arquitetura menos complexa, seguiu de perto com 84,46%, demonstrando robustez apesar de sua simplicidade. Por outro lado, as redes *GoogLeNet*, *ResNet-50* e *ResNet-18*, com arquiteturas mais profundas e complexas, apresentaram acurácias inferiores de 83,07%, 80,68% e 76,10%, respectivamente. Esses resultados indicam que, para este conjunto de dados e tarefa específica, a complexidade da rede não é necessariamente proporcional à acurácia alcançada.

Tabela 22 – *Ranking* de acurácias para 3 classes de atividades em grupo.

| **Posição** | **Arquitetura** | **Acurácia final (conjunto de teste)** |
|---|---|---|
| 1 | SqueezeNet | 86,45% |
| 2 | AlexNet | 84,46% |
| 3 | GoogLeNet | 83,07% |
| 4 | ResNet-50 | 80,68% |
| 5 | ResNet-18 | 76,10% |

Fonte: elaborado pelo autor.

Em contraste com o cenário de 3 classes, a Tabela 23 revela um quadro diferente para a classificação de 4 classes de atividades em grupo. A *ResNet-50*, uma rede de arquitetura mais complexa, liderou com uma acurácia de 82,38%, sugerindo que o aumento na complexidade da tarefa pode se beneficiar de redes mais sofisticadas. A *GoogLeNet* e a *SqueezeNet*, com arquiteturas otimizadas, obtiveram acurácias de 80,72% e 79,82%, respectivamente, exibindo um desempenho consistente, mas ligeiramente inferior à *ResNet-50*. As redes *ResNet-18* e *AlexNet*, com acurácias de 77,86% e 76,96%, respectivamente, mostraram que para tarefas com maior número de classes, as redes com arquiteturas menos

complexas podem não ser tão eficientes. Essa tendência sugere que a complexidade da rede neural pode ter um papel mais significativo em cenários de classificação mais desafiadores.

Tabela 23 – *Ranking* de acurácias para 4 classes de atividades em grupo.

| **Posição** | **Arquitetura** | **Acurácia final (conjunto de teste)** |
|---|---|---|
| 1 | ResNet-50 | 82,38% |
| 2 | GoogLeNet | 80,72% |
| 3 | SqueezeNet | 79,82% |
| 4 | ResNet-18 | 77,86% |
| 5 | AlexNet | 76,96% |

Fonte: elaborado pelo autor.

# 5 CONCLUSÃO

Este trabalho apresentou o desenvolvimento de um sistema de identificação de atividades em grupo baseado na análise de mapas de calor, gerados a partir do processamento de informações das trajetórias dos indivíduos do grupo analisado.

Especificamente, no Capítulo 1 foi apresentada uma breve revisão bibliográfica de trabalhos publicados na literatura que abordam a identificação de atividades humanas. O Capítulo 2 apresentou conceitos teóricos relacionados com o algoritmo proposto, incluindo processamento digital de imagens e redes neurais. No Capítulo 3, foi detalhado o processo de geração de mapas de calor de atividades em grupo e o uso de redes neurais convolucionais para a sua classificação. Por fim, o Capítulo 4 apresentou os resultados da classificação de atividades em grupo usando redes neurais treinadas com 3 e 4 classes de atividades.

No decorrer deste trabalho, alcançaram-se resultados promissores na implementação e treinamento de várias redes neurais, utilizando um conjunto de mapas de calor gerados a partir do processamento da posição de cada indivíduo em *frame* específicos. Esses mapas foram elaborados aplicando-se técnicas do processamento digital de imagens e princípios da transferência de calor. Notavelmente, as redes neurais treinadas demonstraram eficácia na tarefa de classificação, conforme evidenciado pela avaliação criteriosa de suas acurácias, cumprindo assim os objetivos inicialmente delineados no Capítulo 1.

Em relação aos resultados obtidos, constatou-se que as diferentes redes neurais apresentaram uma capacidade variada de processar *datasets* com três e quatro classes de atividades distintas. Para o conjunto com três atividades, a maioria dos classificadores alcançou acurácias superiores a 80%, indicando um potencial significativo para aplicações práticas. Por outro lado, no caso do conjunto com quatro atividades, observou-se uma ligeira redução nas acurácias, embora os valores ainda se mantivessem próximos à acurácia de 80%.

Entre os desafios enfrentados, destacou-se a necessidade de adaptar a equação de cálculo do calor para conferir maior flexibilidade ao algoritmo. Além disso, enfrentou-se o obstáculo de um *dataset* de tamanho limitado, com algumas atividades representadas por um número reduzido de amostras. A seleção de atividades em grupo com características não redundantes também se revelou uma tarefa desafiadora.

Ao comparar o algoritmo implementado com outros da área identificação de atividade humana, este trabalho adotou uma metodologia diferenciada, tratando uniformemente atividades em grupo e atividades individuais, uma abordagem que diverge das propostas por (SHU *et al.*, 2021) e (CHANG; ZHENG; ZHANG, J., 2015). Essa estratégia particular oferece outra perspectiva no que diz respeito ao reconhecimento de atividades humanas.

As implicações práticas e teóricas do trabalho podem contribuir à área de Visão Computacional, particularmente no desenvolvimento de algoritmos aplicáveis no reconheci-

mento de atividades humanas. Embora o algoritmo desenvolvido neste estudo se baseie nos princípios propostos por (LIN *et al.*, 2013), um diferencial significativo foi a incorporação de redes neurais para a etapa de classificação. Além disso, detalhes adicionais de implementação foram cuidadosamente explorados, enriquecendo o trabalho e possibilitando a aplicação do algoritmo em cenários reais com maior eficácia.

Para futuras pesquisas, sugere-se a refatoração do algoritmo de geração dos mapas de calor, a fim de otimizar o desempenho computacional. Além disso, considerando que as acurácias das redes neurais se aproximaram de 80% na classificação de 4 atividades, torna-se relevante desenvolver uma metodologia para a remoção de *outliers*. Esta abordagem poderia incluir a aplicação de filtros para suavizar variações abruptas e atípicas nas atividades reconhecidas em um intervalo de *frames*, assegurando que os resultados reflitam com maior precisão as tendências gerais das atividades em grupo. Por fim, recomenda-se a utilização de *datasets* mais abrangentes, com uma diversidade maior de atividades, para ampliar a robustez e a aplicabilidade dos algoritmos desenvolvidos.

# REFERÊNCIAS

AGGARWAL, Charu C. **Neural Networks and Deep Learning**: A Textbook. 1. ed. New York: Springer., 2018.

BLUNSDEN, S. J.; FISHER, R.B. The BEHAVE video dataset: ground truthed video for multi-person behavior classification. *In*: ANNALS of the BMVA. [*S.l.*: *s.n.*], 2010. v. 4, p. 1–12.

BORGNAKKE, Claus; SONNTAG, Richard E. **Fundamentos da Termodinâmica**. 8. ed. São Paulo: Blucher, 2013.

BURGER, Wilhelm; BURGE, Mark J. **Principles of Digital Image Processing**: Fundamental Techniques. 1. ed. Washington, DC: Springer, 2009.

CHANG, Xiaobin; ZHENG, Wei-Shi; ZHANG, Jianguo. Learning Person–Person Interaction in Collective Activity Recognition. **IEEE Transactions on Image Processing**, v. 24, n. 6, p. 1905–1918, 2015.

CORKE, Peter. **Robotics, Vision and Control**: Fundamental Algorithms in MATLAB. 1. ed. Heidelberg: Springer, 2011.

CS231N. **Convolutional Networks - CS231n Convolutional Neural Networks for Visual Recognition**. [*S.l.*: *s.n.*], 2023. https://cs231n.github.io/convolutional-networks/. Acessado em: 03 de dez. de 2023.

D'SA, Ashwin Geet; PRASAD, B. G. A Survey on Vision Based Activity Recognition, its Applications and Challenges. *In*: 2019 Second International Conference on Advanced Computational and Communication Paradigms (ICACCP). [*S.l.*: *s.n.*], 2019. P. 1–8.

GONZALEZ, Rafael C.; WOODS, Richard E. **Processamento Digital de Imagens**. 3. ed. São Paulo: Pearson, 2009.

GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep Learning**. 1. ed. Cambridge: MIT Press., 2016.

HE, Kaiming; ZHANG, Xiangyu; REN, Shaoqing; SUN, Jian. Deep Residual Learning for Image Recognition. *In*: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [*S.l.*: *s.n.*], 2016. P. 770–778.

IANDOLA, Forrest N.; HAN, Song; MOSKEWICZ, Matthew W.; ASHRAF, Khalid; DALLY, William J.; KEUTZER, Kurt. **SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size**. [*S.l.*: *s.n.*], 2016. arXiv: 1602.07360 [cs.CV].

KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. ImageNet Classification with Deep Convolutional Neural Networks. *In*: PEREIRA, F.; BURGES, C.J.; BOTTOU, L.; WEINBERGER, K.Q. (Ed.). **Advances in Neural Information Processing Systems**. [*S.l.*]: Curran Associates, Inc., 2012. v. 25.

LIN, Weiyao; CHU, Hang; WU, Jianxin; SHENG, Bin; CHEN, Zhenzhong. A Heat-Map-Based Algorithm for Recognizing Group Activities in Videos. IEEE Transactions On Circuits *And* Systems For Video Technology, v. 23, n. 11, p. 1980–1992, 2013.

MACKAY, David J. C. **Information Theory, Inference, and Learning Algorithms**. 4. ed. Cambridge: Cambridge University Press., 2005.

MATHWORKS. **Pretrained Deep Neural Networks**. [*S.l.*: *s.n.*], 2023. Disponível em: https://www.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networksáÿčtml. Acessado em: 09 de nov. de 2023.

MATHWORKS. **Pretrained Deep Neural Networks**. [*S.l.*: *s.n.*], 2023. Disponível em: https://www.mathworks.com/help/deeplearning/ug/build-networks-with-deep-network-designeráÿčtml. Acessado em: 09 de nov. de 2023.

MAXWELL, James Clerk. **Theory of Heat**. 4. ed. Cambridge: Longmans, Greens, e CO., 1904.

MITCHELL, Tom M. **Machine Learning**. 1. ed. New York: McGraw-Hill., 1997.

NIXON, Mark S.; AGUADO, Alberto S. **Feature Extraction and Image Processing**. 1. ed. Oxford: Newnes, 2002.

SHU, Xiangbo; ZHANG, Liyan; SUN, Yunlian; TANG, Jinhui. Host–Parasite: Graph LSTM-in-LSTM for Group Activity Recognition. **IEEE Transactions on Neural Networks and Learning Systems**, v. 32, n. 2, p. 663–674, 2021.

SZEGEDY, Christian; WEI, Liu; YANGQING, Jia; SERMANET, Pierre; REED, Scott; ANGUELOV, Dragomir; ERHAN, Dumitru; VANHOUCKE, Vincent; RABINOVICH, Andrew. Going deeper with convolutions. *In*: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). [*S.l.*: *s.n.*], 2015. P. 1–9.

SZELISKI, Richard. **Computer Vision**: Algorithms and Applications. 1. ed. Heidelberg: Springer, 2010.

WANG, Chuanchuan; MOHAMED, Ahmad Sufril Azlan. Attention Relational Network for Skeleton-Based Group Activity Recognition. **IEEE Access**, v. 11, p. 129230–129239, 2023.

ZHANG, Xuguang; YU, Qinan; YU, Hui. Physics Inspired Methods for Crowd Video Surveillance and Analysis: A Survey. **IEEE Access**, v. 6, p. 66816–66830, 2018.