UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

# Segmentação de vídeos em cenas utilizando detecção de objetos

**Nicole Zukowski Luduvice**

Trabalho de Conclusão de Curso
MBA em Inteligência Artificial e Big Data

ICMC USP
SÃO CARLOS

**UNIVERSIDADE DE SÃO PAULO**
**Instituto de Ciências Matemáticas e de Computação**

# Segmentação de vídeos em cenas utilizando detecção de objetos

***Nicole Zukowski Luduvice***

***Nicole Zukowski Luduvice***

USP - São Carlos

2023

Nicole Zukowski Luduvice

# Segmentação de vídeos em cenas utilizando detecção de objetos

Trabalho de conclusão de curso apresentado ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial

Orientador: Prof. Dr. Rudinei Goularte

USP - São Carlos

2023

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

L94s

Luduvice, Nicole
Segmentação de vídeos em cenas utilizando detecção de objetos / Nicole Luduvice; orientador Rudinei Goularte. -- São Carlos, 2023.
50 p.

Trabalho de conclusão de curso (MBA em Inteligência Artificial e Big Data) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2023.

1. Segmentação de vídeo. 2. Extração de características. 3. YOLO. I. Goularte, Rudinei, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

# AGRADECIMENTOS

Gostaria de agradecer ao meu noivo, Victor Zanin, que esteve comigo durante todo o processo, me apoiando, auxiliando e incentivando sempre. Você tornou esse processo muito mais tranquilo.

Também agradeço aos meus pais cujo investimento contínuo na minha educação foi essencial para realização deste trabalho. Obrigada pelos conselhos e apoio incondicional.

Agradeço ao Prof. Dr. Rudinei Goularte que, com muita paciência, me orientou no projeto, sem ele nada teria sido possível.

# RESUMO

LUDUVICE, N. Z. **Segmentação de vídeos em cenas utilizando detecção de objetos**. 2023. 52 f. Trabalho de conclusão de curso (MBA em Inteligência Artificial e Big Data) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2020.

A pesquisa consiste em examinar o impacto da segmentação de vídeos em cenas através da mudança na forma tradicional de realizar a extração de características. Dado o crescente impacto dos vídeos na sociedade contemporânea, o aumento exponencial de conteúdo audiovisual demanda abordagens inovadoras, sendo a segmentação em cenas uma estratégia essencial para organização dessas mídias. Esta pesquisa busca atuar no desenvolvimento de melhorias na eficácia de segmentação de vídeo. Usualmente, durante a extração de características, os frames completos dos vídeos são usados. O processo proposto neste trabalho, aborda o uso de YOLO (You Only Look Once), uma arquitetura de rede neural convolucional, para detectar regiões de interesse na imagem. Essas áreas são então extraídas da imagem e utilizadas para extração de características, dando base para uma subsequente segmentação de vídeo. O método foi aplicado e métricas de cobertura e transbordamento são calculadas para a análise de desempenho da detecção de cenas em vídeos. Os resultados indicam que, apesar da proximidade de valores e treinamento bem-sucedido do modelo, a inclusão do modelo YOLO no *pipeline* da segmentação não aumenta a eficácia do modelo ao classificar adequadamente os conjuntos de cenas. A comparação com abordagens convencionais indicou que, em determinados contextos, o método proposto pode diminuir a complexidade das estruturas visuais. Este estudo busca explorar estratégias complementares para superar as lacunas identificadas na segmentação de vídeos e destaca a complexidade persistente dessa tarefa e a importância de abordagens mais sofisticadas para enfrentar os desafios inerentes à segmentação em cenas.

Palavras-chave: Segmentação de vídeos em cena; Extração de características; YOLO; Vídeos digitais; Cenas; Frames; Arquitetura de redes neurais.

# ABSTRACT

LUDUVICE, N. Z. Segmentation of videos into scenes using object detection. 2023. 52 f. Final paper (MBA in Artificial Intelligence and Big Data) – Institute of Mathematical and Computer Sciences, University of São Paulo, São Carlos, 2020.

The research consists of examining the impact of segmenting videos in scenes through the change in the traditional way of performing feature extraction. Given the growing impact of videos in contemporary society, the exponential increase in audiovisual content demands innovative approaches, and segmentation into scenes is an essential strategy for organizing these media. This research seeks to work on the development of improvements in the effectiveness of video segmentation. Usually, during feature extraction, the full frames of the videos are used. The process proposed in this work addresses the use of YOLO (You Only Look Once), a convolutional neural network architecture, to detect regions of interest in the image. These areas are then extracted from the image and used for feature extraction, providing the basis for subsequent video segmentation. The method has been applied and coverage as well as spillover metrics are calculated for the performance analysis of scene detection in videos. The results indicate that, despite the proximity of values and successful training of the model, the inclusion of the YOLO model in the segmentation pipeline does not increase the effectiveness of the model by properly classifying the sets of scenes. The comparison with conventional approaches indicated that, in certain contexts, the proposed method can decrease the complexity of visual structures. This study seeks to explore complementary strategies to overcome the gaps identified in video segmentation and highlights the persistent complexity of this task and the importance of more sophisticated approaches to address the challenges inherent in scene segmentation.

Keywords: On-scene video segmentation; Feature extraction; YOLO; Digital videos; Scenes; Frames; Neural network architecture.

SUMÁRIO

# SUMÁRIO

# 1 INTRODUÇÃO

## 1.1 Contextualização

Os vídeos estão cada vez mais presentes no dia a dia do ser humano e são amplamente utilizados em diversas áreas, como lazer, educação, publicidade, pesquisa, entre outros. O seu crescimento tem sido impulsionado por uma combinação de fatores tecnológicos e sociais, inclusive pela fácil absorção de informação de forma rápida, capacidade de transmitir informações de forma clara e objetiva, grande variedade de entretenimento e avanço das tecnologias de transmissão e armazenamento de dados (TERRA.COM.BR, 2021). Com o aumento acelerado da produção de vídeos, a automação torna-se fundamental para organizar e analisar todo esse conteúdo.

Apenas no YouTube, por exemplo, foram adicionadas mais de 500 horas de vídeo a cada minuto em abril de 2022 (CECI, 2023). O YouTube Shorts, um serviço do YouTube lançado globalmente em junho de 2021 para subir vídeos com duração máxima de 1 minuto, ultrapassou 50 bilhões de visualizações diárias em fevereiro de 2023 (CECI, 2023).

Este expressivo volume dificulta na organização e análise dos vídeos. Segmentar os vídeos em pequenas partes facilita este processo, contudo, a forma como são segmentados pode influenciar nos resultados das análises (KISHI, 2020), tornando, assim, importante levar em consideração a estrutura semântica do vídeo.

Dado este cenário, a segmentação de vídeos em cenas se destaca. As cenas são trechos dos vídeos geralmente separadas com base em critérios como mudança brusca de cenário, iluminação e, principalmente, assunto. Em geral, uma mudança de cena se dá quando existe uma quebra na continuidade de pensamento semântico, criando uma troca de assuntos (FONSECA, 2006).

A partir do reconhecimento estrutural de um vídeo, é possível realizar processamentos em seu conteúdo que auxiliam a organização, indexação e recuperação de informações. Uma etapa de extrema importância para a tarefa de segmentação é a extração de características relevantes dos vídeos (KISHI, 2020). Antes de realizar a segmentação de um vídeo, é necessário efetuar um pré-processamento, identificando os principais atributos de um vídeo para utilizá-los em algoritmos de segmentação. Existem diversas técnicas que podem ser utilizadas para realizar essa extração, cada uma explorando um atributo em particular.

A extração de características é um processo que envolve a identificação e a codificação de padrões ou informações úteis dos dados de entrada, com o objetivo de representá-los em um formato mais adequado para a análise posterior (DUDA, R. O.; HART, P. E.; STORK, D. G., 2020). No caso de vídeos, essas características podem ser extraídas de diferentes atributos, tais como visuais, sonoros e temporais dos vídeos, como frequências, amplitude, duração e mudanças de cenas (KISHI, 2020).

Geralmente as técnicas utilizadas para extração de dados visuais consistem em, primeiramente, reduzir o excesso de informações redundantes ao selecionar um subconjunto de imagens representativas dos vídeos, para então utilizá-las como entrada na etapa de extração de características. Há diferentes técnicas para seleção de atributos importantes de um vídeo, com diferentes níveis de complexidade, desde técnicas simples como histogramas de cor até técnicas que envolvem aprendizado de máquinas e processamento de imagem e vídeo mais avançados. Estas técnicas, usualmente, resultam em vetores numéricos de características representando algum aspecto do atributo analisado (AWAD; HASSABALLAH, 2016).

## 1.2 Motivação

Por conta da grande utilização diária de vídeos, por parte de diversos usuários, a separação de vídeos em cenas se torna uma possibilidade interessante para melhorar a usabilidade desses materiais, permitindo aos usuários encontrar informações específicas com maior facilidade. A utilização de técnicas de inteligência artificial tem se mostrado uma solução eficaz para automatizar o processo de segmentação de vídeos (TROJAHN, 2019;). Ainda assim, a segmentação de vídeos é um desafio complexo, longe de ser um problema trivial ou perto de estar resolvido (ROBUST VIDEO, 2017).

Uma forma de aumentar a eficácia do modelo, de modo que a segmentação seja mais eficiente, é melhorar o método de extração de características. Se as características extraídas não forem de grande relevância ou não possuírem informações suficientes, o modelo pode apresentar dificuldades para reconhecer padrões importantes nos dados.

Os métodos de extração de caraterísticas se dividem em duas categorias, globais e locais. Métodos globais utilizam toda a imagem para criar um vetor de características. Por outro lado, os métodos locais identificam pontos de interesse na imagem e, desses pontos, são extraídos os vetores de característica (AWAD; HASSABALLAH, 2016). Ou seja, a extração de características locais busca encontrar, geralmente em toda a imagem, estruturas como

arestas e pontos diferentes de sua vizinhança, localizando, assim, estruturas altamente descritivas da imagem (LOWE, 2004).

Contudo, muitas informações presentes nas imagens são irrelevantes para a análise semântica das cenas. Desta forma, pode ser vantajoso selecionar regiões mais relevantes e significativas na imagem. O plano de fundo de uma imagem, por exemplo, não apresenta uma informação que agrega ao conteúdo da imagem na maioria das vezes.

Neste contexto, modelos de redes neurais, como o YOLO, abreviação para *You Only Look Once,* ou em português, "você olha apenas uma vez", podem ser utilizados para identificar e selecionar objetos de interesse em vídeo (DIVVALA, et al, 2015), permitindo determinar certa região em uma imagem com um objeto correspondente no intuito de extrair apenas as características de regiões que, de fato, representam informações visuais relevantes de interesse.

## 1.3 Objetivo

O objetivo deste trabalho é propor um método que identifica, em imagens, regiões de interesse representativas de um vídeo, de modo que tais regiões sejam as únicas utilizadas para extração de características. Espera-se verificar se o método contribui para a melhoria da eficácia em tarefas de segmentação de vídeo.

# 2. FUNDAMENTAÇÃO TEÓRICA E TRABALHOS RELACIONADOS

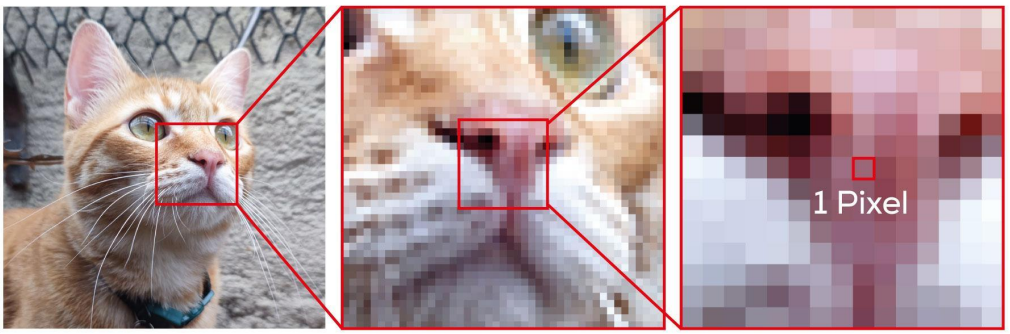
Neste serão abordados os principais conceitos e terminologias para uma melhor compreensão do projeto, bem como, serão apresentados trabalhos encontrados na literatura relacionados ao tema central da monografia.

## 2.1 Vídeo Digital

Segundo a perspectiva teórica de Blanken, Blok, Feng e Vries (2007), vídeos digitais podem ser definidos como uma sequência de imagens estáticas entrepostas em uma determinada velocidade, ou taxa, trazendo uma sensação de movimentação contínua. A maioria dos vídeos incluem elementos visuais e sonoros, como imagens em movimento e efeitos audiovisuais.

Cada imagem estática de um vídeo é chamada de quadro, ou *frame*, comumente utilizado em inglês. Em uma imagem digital, os quadros são formados por uma matriz bidimensional de valores. Os elementos da matriz são chamados de pixels e possuem informações sobre sua cor e intensidade. A quantidade de pixels determina a resolução e qualidade da imagem.

Figura 1 - Representação de um pixel em uma imagem

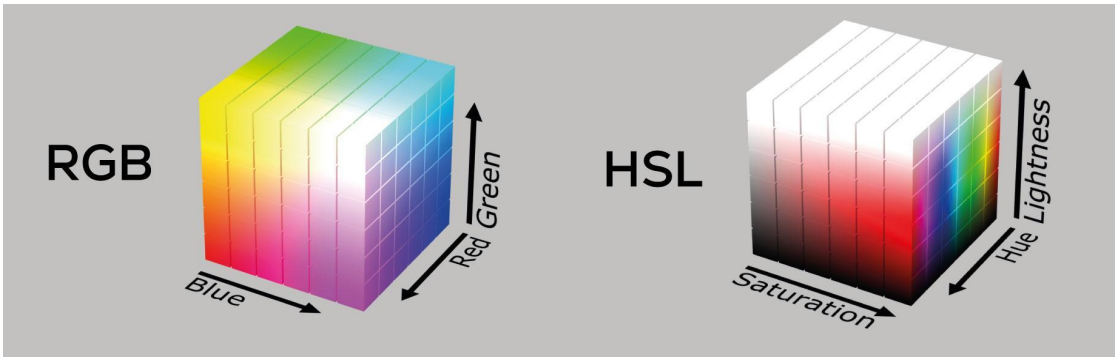



Fonte: De autoria própria (2023)

Um sistema de representação de pixels comumente utilizado é o RGB (Red-Green-Blue ou Vermelho-Verde-Azul), com três bytes. Cada byte indica a intensidade e variação de cor entre 0 e 255 correspondente a cada cor do espectro RGB. Além do padrão de cores RGB, existem outros padrões muito utilizados na área audiovisual, como Hexadecima,

HSL, por exemplo. O hexadecimal possui valores de 0 até 9 e de A até F, mais voltados para interação em websites. O padrão HSL, comumente utilizado em vídeos e suas edições por conta do seu processamento de cores de forma mais humanizada, é representado por *Hue* (tom), *Saturation* (saturação) e *Lightness* (luminosidade), com valores de 0 a 100.

Figura 2 - Diferença entre modos de cor RGB e HSL




Fonte: Datumizer (2008)

Os vídeos podem ser estruturados, semanticamente, de forma hierárquica, subdivididos entre quadros, tomadas e cenas. Começando pela menor medida, os quadros, que podem ser definidos como imagens estáticas únicas que, quando apresentados um após o outro em determinada velocidade, dão a ilusão de movimentação.

Na visão de Koprinska (2001), uma tomada é um conjunto de quadros estáticos, de uma mesma câmera. São segmentados com base na continuidade de informações dos quadros adjacentes, mantendo uma continuação visual dos elementos da tela. O encerramento e troca de tomadas pode ser definido por diversos fatores, tais como mudança de ângulo da câmera, desaparecimento de elementos ou até modificações expressivas de cores em objetos.

A partir de um conjunto de tomadas adjacentes semanticamente relacionadas tem-se uma cena, ou *story units* (BOLLE; YEO; YEUNG, 1998). Segundo Fonseca (2006), o agrupamento de tomadas em cenas é um julgamento subjetivo para dar correlação para a semântica e devem ser realizadas análises além do âmbito visual.

Por fim, o vídeo pode ser definido como uma unidade a partir do conjunto de todas as cenas presentes. Como ressalva temos que: apesar de amplamente adotado, a segmentação hierárquica apresentada não é a única utilizada. Dependendo do objetivo do vídeo, segmentações diversas podem ser apresentadas, como segmentações por notícias em vídeos de reportagens jornalísticas ou até mesmo em tópicos para vídeo-aulas, entre outras (SAWAI; TAKAHASHI; DEUGUICHI; MURASE, 2011).

Figura 3 - Diferença ilustrativa entre vídeo, cenas, tomadas e quadros


Fonte: Adaptada de Trojahn (2019)

## 2.2 Processo de Segmentação

A segmentação em cenas é um processo crucial para a análise e compreensão de vídeos. De modo geral, o processo de segmentação em cenas compreende três etapas principais e sequenciais: Pré-processamento, Segmentação e Pós-processamento.

Primeiramente, o pré-processamento se dá pela criação de uma representação adequada dos vídeos. Nesta fase é realizada a extração de características relevantes dos frames, que após análise, são utilizados para identificar a transição de cenas.

A segunda fase, Segmentação, tem o propósito de identificar ou agrupar todas as tomadas que contabilizam a mesma cena. Para isso, diversas técnicas podem ser aplicadas, incluindo abordagens que se baseiam exclusivamente em informações visuais dos vídeos como aquelas com uma combinação de modalidades, informações sonoras e visuais, por exemplo. A pesquisa irá focar em técnicas de segmentação visuais, abrangendo desde análise de histogramas, correlação em grafos e até redes neurais.

Em resumo, a segmentação em cenas é um processo complexo e fundamental para diversas áreas que contemplem análise de vídeo. É composta por etapas de pré-processamento, segmentação e pós-processamento, que utilizam diversas técnicas para obter uma segmentação automática precisa e eficiente. A escolha das técnicas adequadas para cada etapa pode trazer resultados significativos na qualidade da segmentação obtida.

### 2.2.1 Técnica baseada em grafos

Dentre as técnicas de segmentação em cenas atualmente encontradas na literatura, destaca-se a utilização de grafos, que são normalmente agrupados em subgrafos não conectados. Cada subgrafo representa uma cena. O STG (*scene transition graph*) proposto por Yeung, Yeo e Liu (1998) é uma das principais técnicas fundamentadas em grafos. Nesse caso, cada nó representa um conjunto de tomadas e suas arestas são conectadas caso elas façam parte da mesma cena.

A distância de cossenos entre as tomadas é calculada para realizar o agrupamento, sem ignorar o aspecto temporal das tomadas. A inclusão desse aspecto é essencial para evitar que tomadas distantes, mesmo que similares visualmente, não sejam agrupadas. O algoritmo implementa essa abordagem ao considerar uma distância infinita entre tomadas que estão temporalmente separadas por uma margem superior a ω.

Além de ω, a técnica depende do parâmetro δ, que é utilizada como critério de parada do agrupamento. As tomadas são agrupadas até que a distância entre todos os grupos de tomadas seja maior que δ. O STG é descrito com mais detalhes no Algoritmo 1.

---

**Algoritmo 1** - STG (grafo de transição de cenas)

---

Os argumentos de entrada são: Anotação das transições entre tomadas (***T***), histogramas para cada tomada (***H***), parâmetro de restrição temporal ω e o parâmetro de parada de agrupamento δ.

E para a saída: Uma lista ***B***, com os índices das tomadas que representam o começo e o fim de cada cena.

1. Calcule a distância do cosseno entre os histogramas das tomadas. Se as tomadas forem temporalmente mais distantes que ω, substitua seu valor por ∞
2. Enquanto a distância entre grupos for menor que δ:
    a. Agrupe a tomada ao grupo com menor distância. A distância entre dois grupos, compostos por mais de uma tomada, é calculada como a maior distância entre as tomadas que os constituem.
3. Atribua os grupos resultantes a uma lista ***C.***
4. Crie um grafo ***G*** em que cada vértice é um elemento da lista ***C.***
5. Insira arestas entre os vértices se alguma tomada de um vértice é adjacente a alguma tomada de outro vértice.
6. Encontre e remova todas as arestas de corte em ***G*** usando a busca em profundidade. Adicione os subgrafos resultantes ao conjunto ***R.***

7. Crie uma lista vazia ***B.***
8. Para cada grafo $r \in R$**.**
    a. pegue o índice da primeira e última tomada ***T*** representada por ***R.***
    b. adicione ***T*** a lista ***B.***

---

## 2.3 Extração de Características

De acordo com Duda, Hart e Stork (2000), a extração de características é um processo que se aplica para reduzir o grande volume de dados de um vídeo, retirando informações não importantes e redundantes. As características podem ser classificadas em três diferentes níveis semânticos: alto, médio e baixo (MARQUES; FURHT, 2002; MARTINET; SAYAD,2012).

Primeiramente, as extrações de baixo nível não possuem qualquer significado semântico e são informações extraídas diretamente de dados crus, sem nenhum refinamento ou conhecimento externo, como é o caso de um histograma de cor ou SIFT, que serão abordados nas seções seguintes. Ao extrair características de baixo nível geralmente é necessário usar medidas de dissimilaridade. A medida de dissimilaridade tem como objetivo medir a diferença entre vetores de características distintas.

As características de médio nível semântico, possuem um certo tipo de refinamento em relação às características de baixo nível. Como por exemplo, *bag-of-visual-words* (conjunto, pacote ou saco de palavras visuais), no qual após a extração das características da imagem, é realizado um processo de agrupamento para formar representações de padrões mais complexos e significativos.

Características de alto nível, por sua vez, possuem um significado semântico mais complexo e compreensíveis pelos seres humanos, como rótulos classificando o que está presente em uma imagem.

A extração de características pode ser realizada em diferentes modalidades, como visual, aural e textual. Este estudo foca na extração de características visuais. Os extratores de características visuais, podem ser divididos em globais e locais.

As subseções seguintes descrevem descritores de características muito comuns na literatura.

### 2.3.1 Histograma de cor

Histogramas de cor são gráficos que representam a distribuição de cores presentes em imagens ou vídeos. São comumente utilizados para realizar comparações entre diferentes imagens ou vídeos.

O gráfico sumariza as intensidades das cores presentes em uma imagem e retrata a quantidade de pixels de cada cor. O eixo horizontal do histograma representa as faixas de valores de cores e o eixo vertical representa sua frequência.

O histograma de cor pode ser usado com qualquer representação de cor. A Figura 4, abaixo, utiliza o RGB para representar as cores dos pixels, um dos sistemas mais conhecidos.

Figura 4 - Representação histograma de cor RGB com separação de canais


Fonte: De autoria própria (2023)

A utilização prática do histograma de cor pode ser vasta. Em um vídeo, por exemplo, é possível detectar mudanças na cena ao comparar o histograma de cor de cada quadro do vídeo, procurando por diferenças nos histogramas. Quando existe uma grande diferença, se tem um bom indicativo de mudança de cena.

### 2.3.2 Scale Invariant Feature Transform (SIFT)

O SIFT é um método de extração de características locais muito popular que busca identificar pontos-chave descritivos em imagens. Possui atributos de invariância à escala e à rotação, onde os pontos-chave conseguem manter sua robustez independente da variação desses aspectos na imagem, ou seja, possibilita a identificação, com maior facilidade, de uma

mesma imagem, independentemente de pequenas variações de cor rotação e escala. O SIFT é utilizado para imagens preto e branco e o CSIFT (color SIFT), proposto por Burghoust e Geusebroek (2009) é uma generalização do SIFT que incorpora informações sobre a cor da imagem.

O extrator SIFT consiste de algumas etapas, a saber:

1. Criar o "espaço de escala"

Para lidar com a invariância de escala, são criados vários tamanhos diferentes da mesma imagem. As diferentes escalas das figuras, são chamadas oitavas, e o número de oitavas depende do tamanho da imagem original. Cada oitava possui metade do tamanho da imagem anterior.

As imagens são progressivamente desfocadas para cada oitava por meio de convoluções gaussianas.

$$L(x,\ y,\ \sigma)\ =\ G(x,\ y,\ \sigma)\ *\ I(x,y)$$

$$G(x,\ y,\ \sigma)\ =\ \frac{1}{2\pi\sigma^2}e^{-(x^2+y^2)/2\sigma^2}$$

Em que G é o operador gaussiano de desfoque, I é a imagem, ***x*** e ***y*** são as coordenadas da imagem e σ é o operador de desfoque. Quanto maior seu valor, mais desfocada será a imagem.

Figura 5 - Representação de oitavas com desfoque gaussiano aplicado

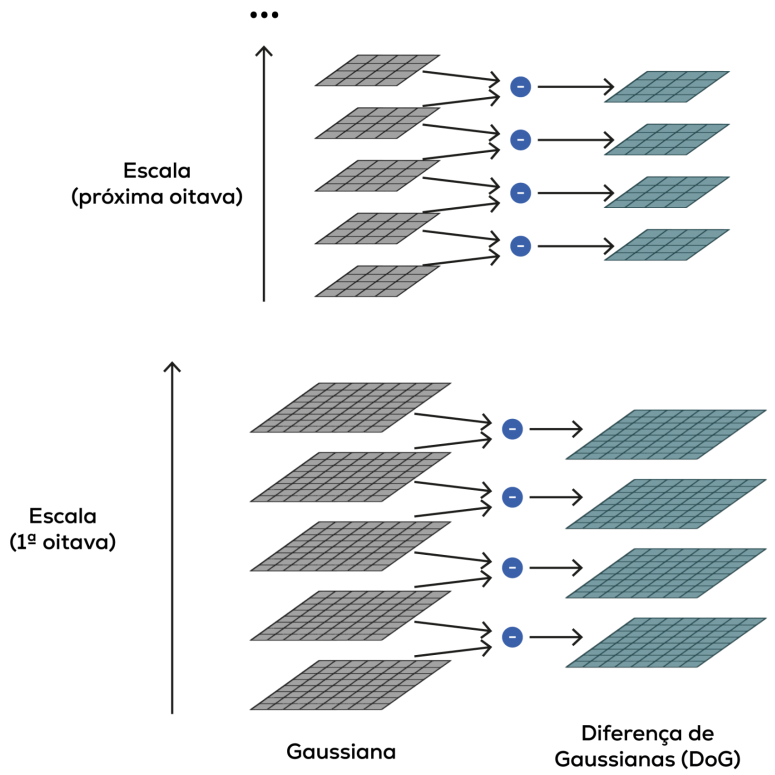

Fonte: De autoria própria (2023)

2. Localizar possíveis *keypoints*

A partir da geração de novas imagens, por meio de diferenças de gaussianas (DoG) o ruído das figuras é reduzido. O operador nada mais é do que uma subtração pixel por pixel de duas imagens da mesma escala com desfoque diferente.

Em seguida, para identificar os pontos-chave, é realizada uma busca pelos máximos e mínimos locais nessa imagem DoG. Procura-se para cada pixel o maior valor entre seus 26 pixels vizinhos, esses valores serão candidatos a pontos-chave. Cada pixel é comparado com os seus 8 pixels adjacentes da mesma imagem mais os 9 pixels da imagem anterior a essa (com mais desfoque) e 9 pixels da imagem posterior a essa (com menos desfoque), todas da mesma escala.

Figura 6 - Representação ilustrativa da aplicação do DoG para localização de *keypoints*



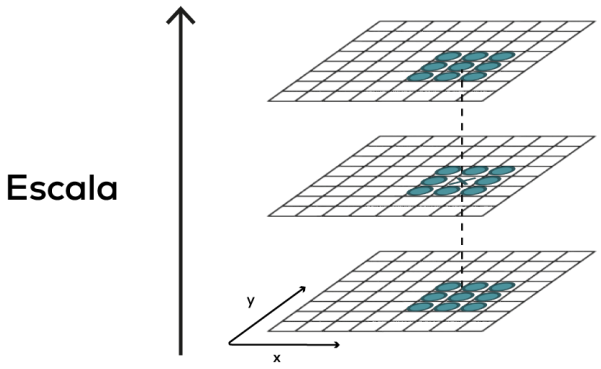

Fonte: Adaptada de Lowe (2004)

3. Filtrar *keypoints*

Os pontos-chave gerados pelos máximos e mínimos locais são muitos, desta forma, é necessário filtrar apenas os pontos mais relevantes.

Para isso, são eliminados pontos-chave com pouco contraste e filtrados os pontos próximos às bordas.

Figura 7 - Representação ilustrativa da filtragem de pontos-chave



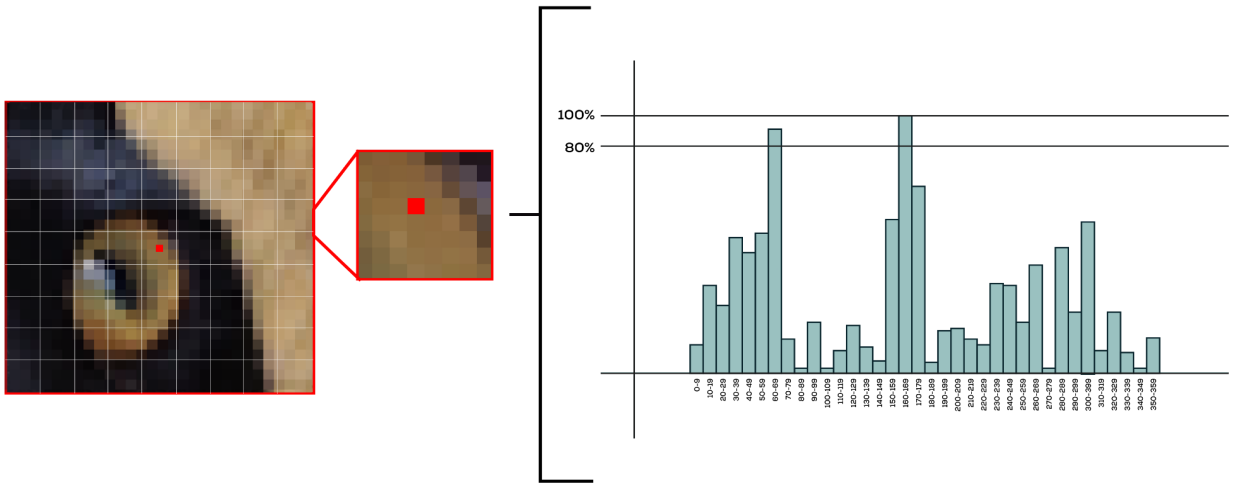

Fonte: Adaptada de Lowe (2004)

4. Torná-los invariantes à rotação

Após definir os pontos-chave da imagem, é realizado um processo para torná-los invariantes à rotação. Isso acontece ao calcular a orientação de cada ponto-chave.

Para determinar a orientação de um ponto-chave é calculada a rotação e a magnitude de cada pixel vizinho ao ponto chave. Em seguida, é gerado um histograma de 36 barras, cuja orientação está no eixo x e o valor do eixo **y** é proporcional à magnitude do gradiente. A orientação representa a direção do pixel e a magnitude representa a intensidade do pixel. O pico do histograma e todo valor de ***y*** acima de 80% é considerado para ser a orientação do ponto-chave.

Figura 8 - Representação ilustrativa da filtragem de pontos-chave




Fonte: Adaptada de Deepanshu Tyagi (2019)

5. Construção dos descritores

A última etapa é a criação dos descritores, onde é gerada uma janela de 16x16 pixels ao redor de cada *keypoint* (ponto-chave). Esse bloco é subdividido em uma janela de 4x4, resultando em 16 sub-blocos. Em seguida, um histograma de orientações e magnitudes como descrito na etapa 4 é criado, com 8 barras para cada sub-bloco. Os histogramas são concatenados, totalizando um vetor de 128 posições (16 blocos x 8 barras).

Finalmente é realizada a diferença entre este vetor normalizado e a orientação do ponto-chave obtido na etapa 4.

### 2.3.3 Bag of visual words (BoVW)

O BoVW é um extrator de características de médio nível semântico. Ele é uma adaptação do famoso *bag-of-words*, utilizado em tarefas de processamento de linguagem natural que cria um dicionário de palavras para representar o texto.

No caso do BoVW, o dicionário visual é formado a partir de partes das imagens, o que considera a localização espacial das características visuais da imagem, que proporciona um entendimento semântico maior da imagem.

São extraídas as informações das imagens por um método de extração de características locais, como o SIFT. Os vetores de características resultantes são agrupados em $N$ clusters, em que $N$ é o tamanho do dicionário. Como o extrator é local, cada vetor

representa um pedacinho da imagem. Dessa forma, um cluster contém partes da imagem similares entre si. Os centróides de cada cluster são coletados para representar as palavras visuais do dicionário.

Após a formação do dicionário de palavras visuais, é gerado um histograma de frequências. Este histograma pode ser usado para realizar comparações entre imagens ou até para ser utilizado como pré-processamento para, por exemplo, um algoritmo de segmentação.

## 2.4 Aprendizagem profunda

A aprendizagem profunda, também conhecida como *deep learning*, é uma subárea da inteligência artificial que se baseia em redes neurais artificiais (RNA) com várias camadas não lineares. O RNA foi inspirado pelo funcionamento do cérebro humano, com neurônios interconectados (DEEP LEARNING BOOK, 2018).

As redes neurais artificiais possuem uma camada de entrada, uma ou mais camadas escondidas e uma cada de saída, todas interconectadas de forma que a saída de uma camada é a entrada da próxima camada.

Cada camada é composta por um conjunto de neurônios que estão associados a determinado peso. No treinamento, esses pesos são adaptados e otimizados para que possam aprender a mapear as saídas desejadas. Para que uma RNA consiga aprender a realizar a tarefa desejada, é fundamental que haja muitos dados rotulados. Dados rotulados são aqueles que possuem as saídas corretas já especificadas. Vídeos para fazer segmentação em cenas, por exemplo, necessitam que as cenas já possuam o limite de cada cena detalhada. Dessa forma o modelo consegue entender como um vídeo é separado em cenas observando exemplos de vídeos e minimizando os erros entre a resposta desejada e a resposta obtida pelo modelo.

Atualmente, as redes neurais profundas têm sido utilizadas amplamente com considerável sucesso em diversos cenários, como classificação de imagem e vídeos, reconhecimento de fala e processamento de linguagem natural. Algumas vantagens da aprendizagem profunda é a capacidade de aprender características complexas dos dados e a generalização que permite produzir previsões para dados nunca antes vistos pelo modelo.

### 2.4.1 Redes convolucionais

As redes neurais convolucionais (CNNs) são redes neurais de múltiplas camadas muito utilizadas para reconhecer padrões de imagens e vídeos. A arquitetura da rede leva em

consideração a estrutura espacial das imagens (DEEP LEARNING BOOK, 2018). Analisar os pixels de uma imagem sem levar em consideração os pixels vizinhos não revela muito sobre a imagem em si. As CNNs utilizam matrizes de filtros para analisar a dependência espacial das imagens, conseguindo extrair características como bordas e formas, permitindo que a rede identifique objetos e padrões em uma imagem.

Os principais tipos de camadas de uma CNN são as camadas de convolução, camadas de *pooling* (agrupamento) e camadas completamente conectadas. As camadas de convolução processam os dados em forma de matriz. Elas possuem filtros bidimensionais de tamanhos variados que, aplicada a convolução à imagem de entrada, é capaz de extrair informações relevantes a respeito da imagem. Na convolução, o filtro é deslizado na imagem de entrada, calculando o produto interno entre a região da imagem e o filtro. O resultado é uma nova matriz que representa a convolução da entrada com o filtro.

As camadas de *pooling* são usadas para reduzir a dimensionalidade e aumentar a velocidade da computação. Realizar *pooling* ajuda a fazer a representação aproximadamente invariante a pequenas translações da entrada. A invariância à translação significa que, se a entrada for transladada em uma pequena quantidade, os valores da maioria das saídas agrupadas não mudam (GOODFELLOW; BENGIO; COURVILLE, 2016). A camada de *pooling* aplica uma operação de redução na imagem de entrada para produzir uma saída de menor dimensão. Várias operações de redução podem ser realizadas, as principais são selecionar o valor médio (*average pooling*) e selecionar o valor máximo (*max pooling*).

Já as camadas completamente conectadas são as camadas finais da CNN e são usadas para produzir os resultados de saída do modelo (DEEP LEARNING BOOK, 2018). Diferente das camadas citadas acima, a camada completamente conectada não recebe uma matriz, mas sim um vetor com as características relevantes da imagem obtida pelas camadas anteriores e retorna o resultado final.

## 2.5 You only look once (YOLO)

You only look once, conhecido como YOLO, é um modelo que usa redes convolucionais para detecção de objetos em tempo real (REDMON; DIVVALA; GIRSHICK; FARHADI, 2015). A detecção de objetos é uma área de visão computacional que visa identificar e localizar objetos de forma automática. O objetivo é identificar a presença de objetos específicos em uma imagem e marcar as suas posições com caixas delimitadoras.

Diferente de algoritmos que dividem o problema em dois estágios, onde primeiramente identificam as regiões por caixas delimitadoras e depois classificam os objetos, o YOLO “olha apenas uma vez” para a imagem a fim de delimitar e classificar os objetos de uma só vez. Isso o torna mais rápido e possibilita a detecção de objetos em vídeos também.

Primeiramente, a imagem é dividida em uma grade ***SxS***, em que ***S*** é um número qualquer. As células da grade são responsáveis por identificar os objetos próximos a sua região. Se o centro do objeto estiver contido em uma célula, essa célula será encarregada de detectar esse objeto, contudo, nem todas as células possuem algum objeto associado a ela.

Cada célula da imagem produz um conjunto de caixas delimitadoras e um escore refletindo o quão confiante o YOLO está de que a caixa delimitadora contém um objeto e a classe que o modelo previu seja, de fato, a classe do objeto. Se não existir nenhum objeto na célula, é esperado que o escore esteja próximo de zero.

Esse processo produz muitas caixas delimitadoras. O valor do escore é então utilizado para filtrar apenas as caixas que realmente possuem algum objeto relevante na imagem.

O algoritmo YOLO é capaz de detectar vários objetos em uma única imagem, mesmo que eles estejam parcialmente obscurecidos ou sobrepostos. Além disso, ele é rápido e preciso. Por isso, é amplamente utilizado em aplicações de visão computacional e aprendizado de máquina.

## 2.6 Trabalhos relacionados

Pesquisas envolvendo técnicas de segmentação de vídeo em cenas começaram a partir da década de 90. Desde então, foram desenvolvidos inúmeros trabalhos discutindo diversos aspectos da segmentação temporal de vídeo em cenas (KENDER; YEO, 1998).

Alguns autores estudaram quais as métricas apropriadas para a tarefa de segmentação. Vendrig (2002), por exemplo, aborda duas novas métricas, a Cobertura e o Transbordamento, para a avaliação de segmentação em cenas pois menciona que as métricas de Precisão e Abrangência não são adequadas para avaliar a segmentação em cenas.

Outros autores focaram em desenvolver novos métodos de segmentação automática de vídeo em cenas, como é o caso de Torjahn (2019) que utiliza características visuais e aurais dos vídeos e propõe um modelo que consiste em combinar redes convolucionais e recorrentes profundas.

Independente do foco da pesquisa, o pré-processamento das imagens dos vídeos é uma etapa indispensável para realização do estudo. Nas escritas de Vendrig e Worring, *Systematic*

*evaluation of logical story unit segmentation* (2002), os histogramas de cores foram empregados para extração de características. Para isso, foi utilizado o HSL para criar histogramas bidimensionais na parte cromática do espaço de cores da imagem, e histogramas de intensidade para a parte acromática.

No trabalho de Torjahn (2019) foram extraídas características convolucionais *(ConvFeat) e Color-SIFT (CSIFT)* na modalidade visual, além de extrair características aurais e textuais por meio do MFCC (SAHIDULLAH; SAHA, 2012) e *Bag of words*, respectivamente.

Os autores em *A semantic-based video scene segmentation using a deep neural network,* Ji, Hooshyar, Kim e Lim (2018), utilizaram textos produzidos a partir de uma variação das redes neurais recorrentes, LSTM, a fim de gerar textos descrevendo as imagens-chave. Foi, então, analisada a semelhança semântica entre tomadas a partir de combinação dos textos gerados e de histogramas de cor.

Quatro elementos foram considerados importantes na constituição de uma cena na pesquisa *A Local-to-Global Approach to Multi-modal Movie Scene Segmentation.* (2020), a saber, lugar, elenco, ação e áudio. Foi utilizado 1) *ResNet50* pré-treinada em imagens-chave para obter características de lugar, 2) *Faster-RCNN e ResNet50* pré-treinadas para detectar instâncias de elenco, 3) *TSN* pré-treinada para obter características de ação, 4) *NaverNet* pré-treinada para separar fala e som de fundo, e STFT para obter características de áudio.

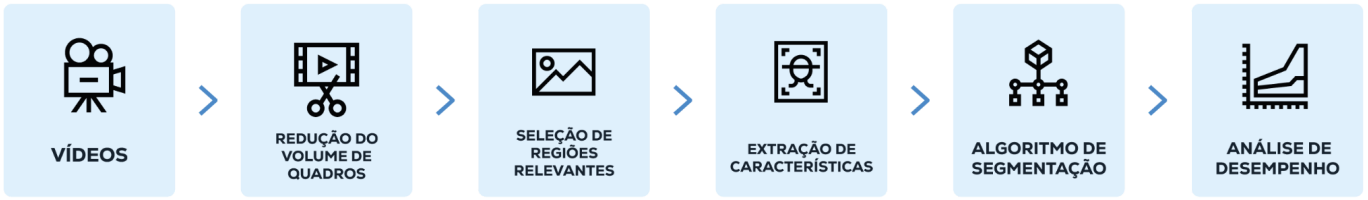
Na publicação *Movie scene segmentation using object detection and set theory* (2019), a extração de características foi realizada através de detecção de objetos. Os objetos de interesse são identificados a partir de redes neurais convolucionais YOLO, tendo como saída o conjunto de objetos encontrados na imagem.

Observe que este último, é similar ao estudo em questão, ao utilizar o YOLO na etapa de extração de características. Contudo, as abordagens diferem na análise da saída do YOLO em cada caso. No contexto desta pesquisa, a classe dos objetos identificados pelo YOLO não é importante para a detecção de cenas. Procura-se apenas pelas regiões importantes da imagem, isto é, o conteúdo dentro das caixas delimitadoras encontradas pelo YOLO.

# 3. METODOLOGIA

Este capítulo apresenta a metodologia utilizada no trabalho. A Figura 9, representa um resumo dos passos para desenvolvimento do projeto. Observa-se que o método proposto é composto por 6 principais etapas que serão descritas nas seções seguintes.

Figura 9 - Representação do processo de metodologia da fase inicial até final




Fonte: A Autora (2023)

## 3.1 Vídeos

Para realizar a segmentação de vídeos em cenas é necessário ter acesso a um conjunto de vídeos relevantes. Anotações indicando o momento da transição entre as tomadas e cenas nos vídeos são essenciais para o treinamento e validação dos modelos.

A disponibilidade de dados abertos com anotações apropriadas para segmentação de vídeos em cenas é limitada. O Dataset *BBC Planet Earth* (Baraldi; Grana; Cucchiara, 2015) foi utilizado no estudo, uma vez que é uma base de dados confiável que contém anotações adequadas para a tarefa de segmentação. A base possui onze documentários sobre diferentes *habitats* do Planeta Terra. Cada episódio contém aproximadamente 48 minutos, com um total de quase 9 horas de vídeo. Cada vídeo possui 2 arquivos de anotações - um com as anotações sobre as cenas e outro com anotações sobre as tomadas. As anotações possuem um formato em que cada linha representa uma tomada/cena com o número inicial e final dos quadros como mostrado na figura abaixo.

Figura 10 - Formato das anotações de cenas e tomadas dos vídeos da BBC

| begin | end |
|---|---|
| 1 | 1 |
| 2 | 9 |
| 10 | 15 |
| 16 | 23 |
| 24 | 27 |
| 28 | 38 |
| 39 | 44 |
| 45 | 49 |
| 50 | 52 |

Fonte: BBC (2023)

A base de vídeos é a entrada inicial para o processo de redução de quadros-chave discutido na seção 3.2.

### 3.2 Redução do volume de quadros

Os vídeos contêm um alto volume de informações visuais redundantes, o que pode resultar em uma enorme quantidade de dados a serem processados sem necessidade. A fim de lidar com essa redundância e reduzir a complexidade computacional, é comumente realizada uma seleção de quadros que representam o conteúdo de um segmento de vídeo (BARALDI, L.; GRANA, C.; CUCCHIARA, 2015; TROJAHN, 2019). Tal processo consiste em separar o vídeo em segmentos e reduzir seu conteúdo ao selecionar um subconjunto representativo de quadros que capturam as principais características visuais do fragmento. Estes quadros são chamados de quadros-chave.

O método utilizado considera a distância entre histogramas de cor dos quadros da mesma tomada para selecionar os quadros-chave como descrito no Algoritmo 1 (KISHI, 2019). Os quadros selecionados para cada tomada possuem o maior nível de similaridade com os quadros restantes e com pouca similaridade entre si.

Quadros com intersecção maior que 0.6 foram considerados similares. Uma análise empírica, em que números entre 0.3 e 0.7 foram testados a fim de definir o valor do parâmetro. Este valor ajuda a definir o número de quadros-chave por tomada.. O número de quadros-chave por segmento deve ser grande o suficiente para representar adequadamente as

informações relevantes dos segmentos e pequeno o suficiente para não sobrecarregar o processamento dos vídeos.

---

**Algoritmo 2:** Seleção de quadros-chave

---

Entrada: Quadros de entrada Q, anotação das tomadas T.
Saída: A lista QC, contendo o índice dos quadros-chave selecionados.
Para cada tomada:

1. Crie uma lista, **H**, com histograma de cor para cada quadro.
2. Calcule distância (intersecção) entre os frames e guarde em uma matriz mxm **D**.
3. Para cada histograma, conte quantos outros histogramas são similares a ele (D[i][j] > 0.6) e salve os valores no vetor **S**.
4. Enquanto:
    a. maxIndex <- argmax(S),
    b. maxValue <- S[maxIndex] - maior valor de S,
    c. candidato <- H[maxIndex] - histograma com mais frames similares,
    d. Se maxValue < 20%*(número total de frames),
        i. Pare o loop.
    e. Se nenhum outro quadro-chave anterior é similar ao candidato
        i. Adicione maxIndex na lista QC
    f. S[maxIndex] <- 0.

---

Após a seleção de quadros-chaves, é comum realizar a etapa de extração de características. No entanto, a inovação do método proposto consiste justamente em adicionar a etapa de seleção de regiões relevantes antes da extração de características.

### 3.3 Seleção de regiões relevantes

Esta é a etapa principal do processo. Nela, é aplicado o modelo YOLO para identificar e delimitar regiões mais relevantes em cada quadro-chave. O YOLO foi escolhido, dentre outras opções, devido a fatores como sua capacidade de generalização, simplicidade conceitual e competitiva acurácia em relação a outros modelos. Além disso, “olhar apenas uma vez” para a imagem completa e detectar diversas classes e regiões em diferentes escalas, torna-a bastante eficiente e rápida, o que contribuiu bastante para a decisão.

O modelo cria automaticamente caixas delimitadoras ao redor dos objetos de interesse presentes no quadro. Apenas as regiões contidas dentro dessas caixas serão consideradas para a extração de características. Espera-se que o modelo YOLO direcione o processo de segmentação de vídeos para áreas mais relevantes da imagem, melhorando o seu desempenho.

Como as classes do modelo YOLO pré-treinado não são as mesmas classes presentes na base de dados do BBC, é necessário mapear as classes relevantes na base e treinar o YOLO. Os passos realizados foram:

1. Mapear as classes de interesse nos vídeos. Foram identificadas 57 classes de animais presentes na base.
2. Baixar as classes mapeadas usando o Image Net, uma base de dados com milhões de imagens e anotações amplamente utilizadas em visão computacional. Inicialmente, optou-se por baixar até 1.000 imagens por classe, contudo devido a limitações computacionais, a restrição foi reduzida pela metade, fixando-se em até 500 imagens por classe.
3. Padronizar o formato das imagens. Os dados obtidos através do Image Net não estavam padronizados, em particular os metadados, o que impacta na fase de treinamento. Dessa maneira, a ferramenta Roboflow foi usada para unificar e padronizar o formato dos metadados conforme as exigências do treinamento que exige um arquivo txt para cada imagem. Neste arquivo, cada linha contém as informações sobre a localização de um bounding box e sua respectiva classe. Se uma imagem possui um gato e um cachorro, por exemplo, o arquivo txt deve conter duas linhas, uma com cada classe do animal, além de conter o ponto central da caixa delimitadora no eixo x, o ponto central da caixa delimitadora no eixo y, a largura e altura da caixa, resultando em um total de cinco colunas.
4. Treinar o modelo com transfer learning ao congelar os pesos iniciais do modelo. Nessa etapa foi dividido o conjunto de dados em teste (80%), validação (10%) e treino (10%)
5. Avaliar os modelos treinados, os quais foram submetidos a treinamentos, considerando a opção de congelar apenas o backbone (10 primeiras camadas) ou todos os pesos. Implementar o modelo escolhido nos keyframes previamente extraídos da base de dados.
6. Extrair as bounding boxes geradas pelo YOLO. Para isso, foi recortada a região correspondente a cada bounding box. Caso o modelo não encontre nenhum bounding box, a imagem completa foi utilizada.

As imagens extraídas foram empregadas na etapa de extração de características para transformar os quadros recortados em representações compreensíveis pelo algoritmo de segmentação.

## 3.4 Extração de características

O resultado da extração de características consiste, geralmente, em vetores numéricos utilizados como entrada para o processo de segmentação de cenas. Ao converter a imagem em vetores numéricos por meio de algoritmos extratores de características, é possível representar algum aspecto do conteúdo visual de modo que seja processável por algoritmos empregados em tarefas como segmentação de cenas.

Foram extraídas as características utilizando o método *bag-of-visual-words*, discutido anteriormente na seção 2.3.3. Nele as características locais da imagem são extraídas usando o extrator de características SIFT. Em seguida, é construído um dicionário de palavras visuais e, posteriormente, um histograma de frequências de palavras visuais. Este histograma é usado como entrada para o algoritmo de segmentação.

A escolha do tamanho do dicionário é uma tarefa empírica, sendo que neste trabalho foram testados os tamanhos 100, 200 e 300. Essa escolha de tamanhos se deve a trabalhos anteriores em tarefas de segmentação, na base BBC, terem obtido sucesso com valores próximos a esses (KISHI, 2020).

## 3.5 Algoritmo de segmentação

Os algoritmos de segmentação recebem como entrada os vetores com as informações extraídas da etapa anterior, onde irão buscar detectar os momentos em que ocorreram transições de cenas, ou seja, tomadas do vídeo que delimitam o final de uma cena e o início da próxima.

A técnica adotada para identificar a mudança de cenas foi a aplicação do algoritmo STG (*Scene Transition Grap*h), já conhecido na literatura (YEUNG, YEO, LIU, 1998). Seu objetivo é agrupar tomadas que pertencem à mesma cena levando em consideração o aspecto temporal dos vídeos. O STG é um algoritmo seminal da área, que por sua flexibilidade, pode ser utilizado como baseline. O objetivo de sua utilização neste trabalho é possibilitar uma comparação entre segmentações de cenas realizadas com características provenientes da

imagem toda versus características (possivelmente mais semânticas) advindas apenas das regiões da imagem consideradas mais relevantes. Mantendo-se o mesmo algoritmo de segmentação para os dois casos.

O algoritmo STG possui dois parâmetros: $\omega$ e $\delta$. O parâmetro $\omega$ é uma restrição temporal para a distância entre duas tomadas e $\delta$ está relacionado ao critério de parada do agrupamento. Vale notar que alterações em ambos os parâmetros podem ter impacto nos resultados obtidos.

A fim de identificar uma combinação dos parâmetros que otimizam a segmentação para base de dados escolhida, foi elaborada uma grade de parâmetros. O $\delta$ foi testado para os valores 0.35 e 0.6, enquanto o $\omega$ foi avaliado com valores 3, 7 e 10 resultando em seis combinações diferentes de parâmetros. A seção 4.2 detalha os resultados dessa análise.

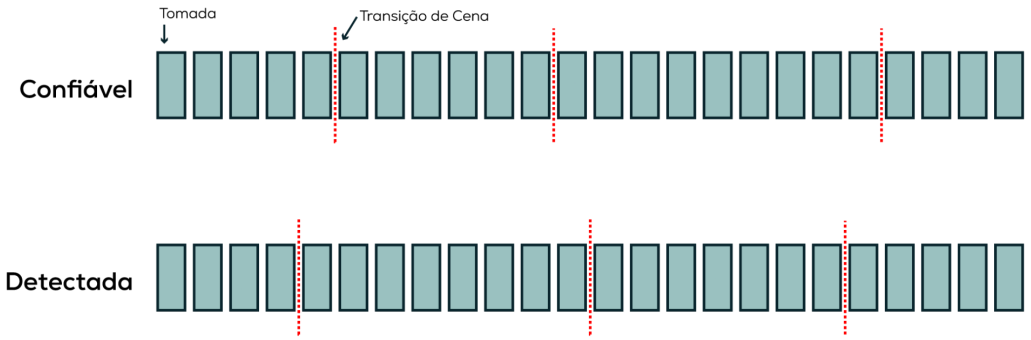
### 3.6 Análise de desempenho

Como a inovação do método proposto é a inclusão da etapa de seleção de regiões relevantes, a análise de desempenho busca avaliar se, ao adicionar essa etapa, o modelo contribui para melhorar a eficácia da segmentação de vídeos em cenas.

Para isso, métricas de desempenho apropriadas devem ser utilizadas. Em tarefas de classificação, as medidas de abrangência e precisão são muito populares. Contudo, não são ideais para a tarefa em questão, dado que não considera segmentações parcialmente corretas. Nesse caso as métricas são muito rigorosas, e não avaliam de forma desejada o resultado da segmentação.

Na Figura 11, um vídeo foi segmentado por meio de um algoritmo de segmentação em cenas. Verifica-se que as transições de cenas, embora não estejam corretas em relação a segmentação confiável, apresentam uma segmentação próxima da desejada. Entretanto, tanto para a precisão quanto para a abrangência, os valores são de 0%.

Figura 11 - Uma segmentação em cenas confiável e outra detectada por um algoritmo de segmentação




Fonte: Adaptada de Kishi (2020)

Neste contexto, as métricas de cobertura (C) introduzidas por Vendrig e Worring (2002) e *New overflow* (O) proposta por Han e Wu (2011) foram utilizadas neste trabalho. Neste trabalho traduz-se o termo *overflow* como transbordamento.

A cobertura mede o quanto as tomadas que pertencem à mesma cena foram corretamente preditas. Quanto maior esse valor, mais próxima da realidade é a segmentação do algoritmo. Por outro lado, o transbordamento mensura quantas tomadas foram erroneamente classificadas como pertencentes à mesma cena. Um valor alto de transbordamento indica uma má segmentação.

Considere $S = \{s_1, ..., s_{|S|}\}$ como as cenas reais e $\hat{S} = \{\hat{s}_1, ..., \hat{s}_{|\hat{s}|}\}$ como as cenas preditas. A cobertura e o transbordamento para a cena $t$ são definidas como:

$$C_t = \frac{max_{i=1,..,|\hat{S}|} |s_t \cap \hat{s}_i|}{|s_t|}$$

$$O_t = 1 - \frac{|s_t|}{\sum_{j, s_t \cap \hat{s}_j \neq \phi} |\hat{s}_j|}$$

Em que $|s_i|$ é o número de tomadas pertencentes à cena $i$. e $C_t$ $e$ $O_t$ são a cobertura e transbordamento para a cena real $t$, respectivamente.

Os resultados obtidos a partir das métricas permitem determinar se a inclusão da etapa de seleção de regiões relevantes obteve uma contribuição significativa para melhorar a detecção de mudanças de cena em vídeos.

### 3.7 Ambiente de Treinamento

O ambiente de treinamento utilizado neste trabalho foi uma máquina com as configurações informadas na Tabela 1. O Vscode e Google Colab foram usados para rodar os scripts em Python 3.11. As principais bibliotecas usadas foram: cv2, pandas, numpy, torch e networkx. Além disso, para treinar o modelo, foi clonado o repositório Yolov5 do github.

Tabela 1 - Especificações do ambiente de treinamento

| **Especificações** | |
|---|---|
| CPU | 16 vCPU Intel 4.60 GHz |
| RAM | 16 GB |
| GPU | NVIDIA 3050 6GB |
| SO | WINDOWS 11 PRO 22H2 |

Fonte: A Autora (2023)

# 4. RESULTADOS

Neste capítulo serão discorridos os resultados obtidos na pesquisa. Este capítulo está dividido em duas seções. Na primeira seção são apresentados os detalhes do treinamento do modelo Yolo, onde se pode compreender como foi desenvolvido e escolhido o método proposto Em seguida, é avaliada a eficácia do método proposto se comparado ao pipeline original a fim de analisar se a proposta apresenta uma melhora na eficácia de detecção de transição de cenas.

## 4.1 Treinamento do modelo YOLO

O modelo YOLO já é treinado para os dados COCO (cocodataset, 2015), que possui 330 mil imagens e 80 classes. Essas classes, porém, não contêm todos os animais presentes na base de dados da BBC. Dessa forma, foi necessário desenvolver um novo modelo YOLO treinado para as 57 classes mapeadas.

Existem cinco arquiteturas diferentes no YOLOv5, que variam principalmente pelo número de parâmetros, são elas: *nano, small, medium, large e extra large*. Quanto maior o modelo, melhor o desempenho do mesmo, entretanto, vale notar que modelos com mais parâmetros exigem mais memória. Optou-se pelo modelo *medium,* a fim de se obter melhores resultados, considerando as limitações do ambiente de treinamento.

Dois modelos foram analisados, ambos com *transfer learning*, uma vez que o número de imagens por classe é limitado. A diferença entre os modelos foi no número de camadas congeladas. O número de camadas pode impactar significativamente os resultados das métricas. Para entender quantas camadas congeladas resultam em um melhor modelo, criou-se dois modelos diferentes.

No primeiro modelo, as camadas responsáveis pela extração das características da imagem foram congeladas. Essas primeiras camadas são chamadas de backbone e consistem das 10 primeiras camadas do modelo. Ao congelar algumas camadas, informações relevantes do modelo anteriormente treinado com mais dados são transferidos para o novo modelo. O número de camadas ideal é sujeito a fatores como o tamanho dos dados, arquitetura do modelo, limite computacional usado para treinar e, principalmente, complexidade da tarefa. Se um modelo for similar ao modelo pré-treinado, pode-se congelar mais camadas e reduzir o custo computacional. Dessa forma, supondo que os modelos são semelhantes com algumas

classes diferentes, optou-se em congelar todas as camadas, exceto a última, no segundo modelo, totalizando 24 camadas congeladas.

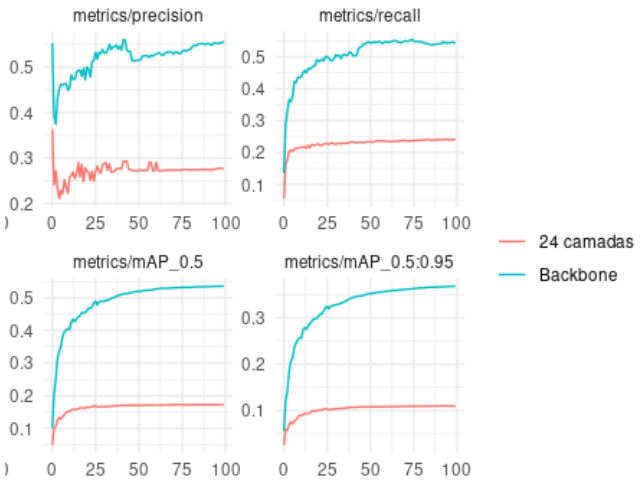
Um teste inicial foi conduzido com 100 épocas, a fim de avaliar o desempenho geral durante o treinamento. A Figura 12 mostra os resultados obtidos pelos modelos. Espera-se verificar qual modelo proposto possui melhor desempenho. Nesta Figura, cada gráfico representa uma medida diferente:

- precisão: é a razão entre verdadeiros positivos e todos os positivos preditos,
- *recall*: é a razão entre verdadeiros positivos e todos os positivos reais,
- AP: é a área embaixo da curva precisão-recall, que resume as duas métricas em uma,
- IoU: quantifica intersecção entre a caixa delimitadora predita e a real,
- mAP0.5: é a média de AP para diferentes classes, em que uma classificação é positiva se o IoU for maior que 0.5 e negativo caso contrário,
- mAP0.5-0.95: é a média de AP para diferentes classes, em que uma classificação é positiva considerando IoU variando entre 0.5 e 0.95.

As funções de perda são usadas para minimizar o erro durante o treinamento. As medidas restantes são usadas para avaliar a qualidade das previsões.

Observa-se que o modelo com apenas o *backbone* congelado possui todas as métricas significativamente mais altas. Além disso, as métricas no conjunto de teste também foram maiores. Portanto, optou-se pelo modelo com backbone congelado.

Figura 12 - Resultados do treinamento comparando os modelos




Fonte: A Autora (2023)

Para determinar o número de épocas, foi treinado um novo modelo com 300 épocas, congelando apenas o *backbone*. Como o objetivo da utilização do YOLO é a extração das caixas delimitadoras, foi dado um foco maior para as funções de perda que envolvem esta tarefa. O YOLO usa 3 funções de perdas:

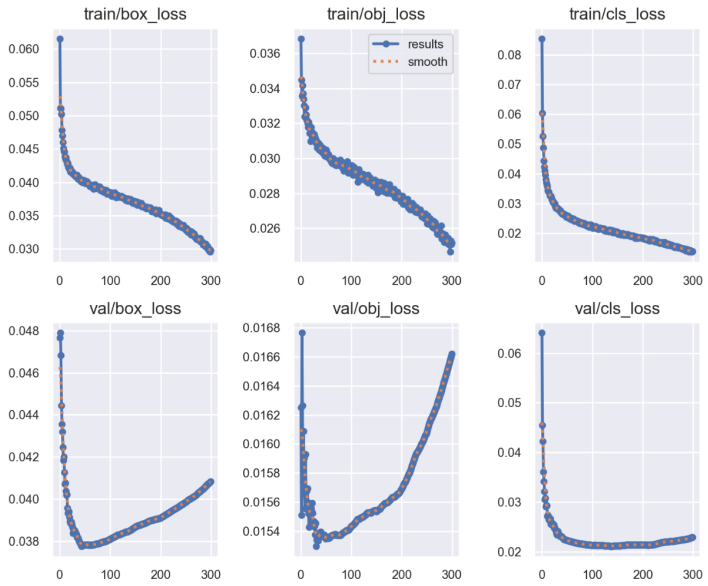
box_loss: é a função de perda das caixas delimitadoras

obj_loss: é a função de perda que mede a presença ou ausência do objeto dado a caixa delimitadora real

cls_loss: é a função de perda da classificação (Cross entropy)

Verificou-se que a partir da época 50 o modelo começa a sobreajustar (*overfit*). O modelo final adotado foi o YOLO de arquitetura média com 50 épocas e backbone congelado.

Figura 13 - Resultados do treinamento de 300 épocas




Fonte: A Autora (2023)

A fim de verificar se o modelo escolhido consegue prever a localização dos objetos, foi realizada uma comparação entre o modelo treinado com as classes do BBC e o modelo nativo do YOLO que utiliza a base COCO. A Tabela 2 mostra a comparação entre os modelos usando as métricas mAP.

Tabela 2: Comparação entre modelo nativo do YOLO e modelo treinado

| | mAP50-95 | mAP50 |
|---|---|---|
| YOLOv5m nativo | 45.4 | 64.1 |
| YOLOv5m treinado | 35.5 | 51.4 |

Fonte: A Autora (2023)

As informações referentes às métricas de base COCO, foram retiradas de uma biblioteca online, YOLOv5, diretamente do *github* (YOLOv5, 2023). Salienta-se que, apesar do modelo nativo possuir melhores métricas, o modelo desenvolvido se aproxima desses valores e é suficientemente eficaz para a tarefa em questão.

Algumas técnicas para melhorar as métricas do modelo proposto incluem limpar e aprimorar as anotações dos dados, aumentar a base de dados, empregar técnicas de aumento dos dados (*data augmentation*), aumentar o número de parâmetros da arquitetura para um YOLOv5l ou YOLOv5x. Esses são tópicos para trabalhos futuros.

### 4.2 Segmentação em cenas

Para otimizar o desempenho da segmentação, foram desenvolvidos diversos segmentadores com combinações de tamanho do dicionário, parâmetro do STG ω e δ diferentes. Para facilitar, o pipeline que usa YOLO será chamado de Y-STG e o pipeline original, que extrai as características de toda a imagem, será chamado de O-STG.

Os resultados obtidos na segmentação foram analisados usando métricas de cobertura (C), *new overflow (NO)* e F1. A métrica F1, que combina a cobertura e *new overflow* em apenas uma métrica, será a principal utilizada para analisar o desempenho dos algoritmos de segmentação.

$$F1 = 2.\frac{C^*(1-NO)}{C+(1-NO)}$$

O segmentador que alcançou maior F1 para Y-STG, dentre os testados, possui os parâmetros de tamanho do dicionário 200, $\delta = 0.35$ e $\omega = 7$.

Em seguida, o mesmo conjunto de parâmetros foi aplicado ao *pipeline* O-STG. A Tabela 3 apresenta a comparação entre Y-STG e O-STG para cada vídeo da base BBC.

Tabela 3: Resultados obtidos comparando o pipeline Y-STG e O-STG, utilizando a melhor combinação de parâmetros de Y-STG

| | F1 | |
|---|---|---|
| Vídeo | Y-STG | O-STG |
| From Pole to Pole | 0.63 | 0.60 |
| Mountains | 0.63 | 0.63 |
| Ice Worlds | 0.05 | 0.05 |
| Great Plains | 0.55 | 0.61 |
| Jungles | 0.62 | 0.64 |
| Seasonal Forests | 0.59 | 0.05 |
| Fresh Water | 0.58 | 0.60 |
| Ocean Deep | 0.64 | 0.67 |
| Shallow Seas | 0.63 | 0.67 |
| Caves | 0.07 | 0.05 |
| Deserts | 0.61 | 0.64 |
| Média | 0.51 | 0.47 |

Fonte: A Autora (2023)

Os valores com F1 muito baixo foram observados nos vídeos ***Ice Worlds*** e ***Caves*** para ambos os processos, e ***Seasonal Forests*** para O-STG. Deve-se estressar que isso ocorre devido a uma limitação do algoritmo escolhido para realizar a segmentação - STG é um algoritmo baseline.

Observa-se na Tabela 3, que o *pipeline* proposto no trabalho possui, em média, um F1 cerca de 3% maior que o *pipeline* original para essas configurações de parâmetros. Contudo ao olhar individualmente cada vídeo, o O-STG possui, em geral, valores de F1 próximos ao Y-STG. O F1 decaiu devido ao vídeo ***Seasonal Forests*** que não foi adequadamente segmentado em O-STG mas foi em Y-STG.

Para uma comparação mais justa, optou-se por comparar o melhor desempenho de O-STG com o melhor desempenho de Y-STG. Para isso, identificou-se que a combinação mais eficaz para o O-STG envolve um tamanho de dicionário de 300, um $\delta$ de 0.35 e um $\omega$ de 7.

Tabela 3: Comparação entre resultados obtidos comparando o pipeline Y-STG e O-STG, utilizando a melhor combinação de parâmetros para Y-STG e O-STG separadamente

| | F1 | |
|---|---|---|
| Vídeo | Y-STG | O-STG |
| From Pole to Pole | 0.63 | 0.62 |
| Mountains | 0.63 | 0.63 |
| Ice Worlds | 0.05 | 0.05 |
| Great Plains | 0.55 | 0.64 |
| Jungles | 0.62 | 0.64 |
| Seasonal Forests | 0.59 | 0.61 |
| Fresh Water | 0.58 | 0.56 |
| Ocean Deep | 0.64 | 0.63 |
| Shallow Seas | 0.63 | 0.69 |
| Caves | 0.07 | 0.06 |
| Deserts | 0.61 | 0.62 |
| Média | 0.51 | 0.52 |

Fonte: A Autora (2023)

Note que com essas configurações, o O-STG possui um valor 1% maior que o Y-STG. Apesar de maior, a diferença é muito pequena.

O *pipeline* original precisa de um tamanho de dicionário um terço maior para conseguir melhorar o desempenho da segmentação. Isso implica em maior esforço computacional já que os vetores de características e os histogramas de representação resultantes possuem sempre dimensionalidade diretamente proporcional ao tamanho do dicionário.

Esses resultados trazem alguns indícios:

1. Ao extrair apenas as informações relevantes das imagens, o Y-STG simplifica a estrutura das características visuais, contudo não necessariamente melhora a eficácia da tarefa de segmentação em vídeo. Isso pode estar relacionado ao tipo de característica utilizada e o tipo de imagens presentes na base. O extrator SIFT produz melhores resultados quando as imagens são ricas em detalhes.
2. A pequena diferença, de 1%, a favor do modelo O-STG, sugere que pode ser benéfico utilizar o Y-STG já que o mesmo necessita de dicionários menores -

com um terço do tamanho. Isso levaria a um menor esforço computacional já que a dimensionalidade dos histogramas de representação é menor.

# 5. CONCLUSÃO

Como proposta desse projeto, foi desenvolvido um método de segmentação de vídeos em cenas com objetivo de testar se essa abordagem contribui para o aprimoramento da eficácia da tarefa.

A segmentação normalmente envolve 3 etapas principais em ordem temporal: a seleção de frames, a extração de características e a segmentação das tomadas em cenas. A inovação introduzida neste trabalho acontece na inclusão de uma etapa adicional, que implica em, antes de extrair as características, identificar regiões relevantes das imagens por meio do YOLO e utilizar estas regiões na etapa seguinte. A inserção desta etapa demonstrou resultados promissores.

Foram comparadas as métricas de *coverage* e *overflow* entre o *pipeline* original e o *pipeline* modificado para cada um dos 11 episódios presentes na base. Observou-se que a inclusão do YOLO, antes da extração das características, não apresentou diferenças significativas, dado que o *pipeline* original possui um valor apenas 1% maior que o *pipeline* modificado. Contudo, a dimensionalidade da representação dos dados produzida pelo modelo proposto possui um tamanho um terço menor. Isso significa que, com menos volume de dados, o modelo proposto se aproxima do modelo original.

Com os resultados obtidos na pesquisa, pode-se concluir que o objetivo foi atendido ao verificar que a aplicação de uma etapa adicional no processo de segmentação de vídeo não contribuiu de forma significativa para o processo. A pesquisa também contribuiu para entender que, ao simplificar a estrutura das características visuais, o SIFT pode não ser ideal para realização da extração de características, dado que produz melhores resultados em imagens ricas em detalhes.

Algumas hipóteses e limitações da técnica proposta foram levantadas e podem ser exploradas em trabalhos futuros em uma tentativa de ampliar o impacto do método proposto:

1. Explorar outras características em lugar de SIFT.
2. Verificar se aprimorar o treinamento do YOLO pode melhorar o desempenho do pipeline.
3. Explorar métodos de detecção de objetos alternativos ao YOLO, como o Fast-RCNN ou modelos Transformers para imagens.
4. Analisar o quão generalizado é o resultado obtido neste trabalho, utilizando outras bases de vídeos.

## REFERÊNCIAS

AYND, T.; KOLAMAN, M. S.; SAHIN, E., (2021). Audio and Video Feature Extraction for Automatic Segmentation of Lecture Videos. International Journal of Educational Technology in Higher Education, 18(1), 1-21.

AWAD, A. I.; HASSABALLAH, M. **Image Feature Detectors and Descriptors:** Foundations and Applications, 2016.

AL-TURAIGI H. M., & Hussain, A. (2019). Feature extraction: A review. International Journal of Computer Applications, 182(22), 9-14.

BENGIO, Yoshua; COURVILLE, A.; GOODFELLOW, Ian. Deep Learning. 2016. Disponível em: http://fa.bme.sut.ac.ir/Downloads/AcademicStaff/3/Courses/44/Book_Deep%20Learning_Ian%20Goodfellow.pdf. Acesso em: 19 mar. 2023.

BLANKEN, H. M.; BLOK, H. E.; FENG, L.; VRIES, A. P. de (Ed.). Multimedia Retrieval. Springer, 2007. (Data-Centric Systems and Applications). ISBN 978-3-540-72894-8. Disponível em:<https://doi.org/10.1007/978-3-540-72895-5>. Acesso em: 26 nov. 2023.

BOLLE, R. M.; YEO, B. L.; YEUNG, M. M. Video query: Research directions. IBM Journal of Research and Development, IBM, v. 42, n. 2, p. 233–252, mar 1998. ISSN 0018-8646. Disponível em:< https://ieeexplore.ieee.org/document/5389317/>. Acesso em: 26 nov. 2023.

BURGHOUTS, G. J.; GEUSEBROEK, J. M. Performance evaluation of local colour invariants. **Computer Vision and Image Understanding, Elsevier Inc**., v. 113, n. 1, p. 48–62, 2009. ISSN 10773142. Disponível em: <https://dx.doi.org/10.1016/j.cviu.2008.07.003>. Acesso em: 26 nov. 2023.

CECI, L. YouTube - Statistics & Facts. **Statista,** 7 feb. 2023. Disponível em: https://www.statista.com/topics/2019/youtube/#topicOverview. Acesso em: 19 mar. 2023.

CHENG, C. M., & Tsai, W. H. (2013). Performance evaluation of feature extraction methods for visual tracking. Journal of Information Hiding and Multimedia Signal Processing, 4(1), 61-70.

LIN, T. Y., et al.. (2015). **Microsoft COCO: Common Objects in Context.** ArXiv preprint arXiv:1405.0312.

Data Science Academy. Deep Learning Book, 2022. Disponível em: <https://www.deeplearningbook.com.br/o-que-sao-redes-neurais-artificiais-profundas/>. Acesso em: 19 mar. 2023.

DEGUCHI, D.; SAWAI, K.; TAKAHASHI, T.; IDE, I.; MURASE, H. Scene segmentation of wedding party videos by scenario-based matching with example videos. In: **Proceedings of the 19th ACM International Conference on Multimedia**. New York, New York, USA: ACM Press, 2011. (MM 11), p. 1545. ISBN 9781450306164. Disponível em: <https://dl.acm.org/citation.cfm?id=2072298.2072061>. Acesso em: 26 nov. 2023.

DIVVALA, S. et al. **You Only Look Once: Unified, Real-Time Object Detection**. 2015. Disponível em: https://arxiv.org/pdf/1506.02640v5.pdf. Acesso em: 19 mar. 2023.

FONSECA, M. S. **Combinando imagem e som para detecção de transições em vídeos digitais. Dissertação** (Mestrado) - Universidade Federal Fluminense, Niterói - RJ, Brasil, 2006. Disponível em: <http://www.dominiopublico.gov.br/pesquisa/DetalheObraForm.do?select_action=&co_obra=159539>. Acesso em: 26 nov. 2023.

HAN, B.; WU, W. Video scene segmentation using a novel boundary evaluation criterion and dynamic programming. In: **2011 IEEE International Conference on Multimedia and Expo.** Barcelona, Spain: IEEE, 2011. p. 1–6. ISSN 1945-7871.

HAQ, I. U.; MUHAMMAD, K.; HUSSAIN, T.; KWON, S.; SODANIL, M.; BAIK, S. W.; LEE, M. Y. Movie scene segmentation using object detection and set theory. Sage Journals, 2019. Disponível em: https://journals.sagepub.com/doi/pdf/10.1177/1550147719845277. Acesso em: 19 mar. 2023.

JI, H.; HOOSHYAR, D.; KIM, K.; LIM, H. A semantic-based video scene segmentation using a deep neural network. Sage Journals, December 19, 2018. Disponível em: https://journals.sagepub.com/doi/abs/10.1177/0165551518819964?journalCode=jisb. Acesso em: 19 mar. 2023.

KENDER, J. R.; YEO, B. L. Video scene segmentation via continuous video coherence. In: **Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition.** Washington, DC, USA: IEEE Computer Society, 1998. (CVPR '98), p. 367–ISBN 0-8186-8497-6. Disponível em: <http://dl.acm.org/citation.cfm?id=794191.794801>. Acesso em: 26 nov. 2023.

KISHI, R. M. **Fusão de informação multimodal por detecção de correlação para tarefas de análise de vídeo**. 2020. 165 p. Tese (Doutorado) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos - SP, 2020.

KOPRINSKA, I.; CARRATO, S. Temporal video segmentation: A survey. Signal Processing: Image Communication, v. 16, n. 5, p. 477-500, 2001. ISSN 0923-5965. Disponível em: https://doi.org/10.1016/S0923-5965(00)00011-4. Acesso em: 26 nov. 2023.

LOWE, D. G. Distinctive Image Features from Scale-Invariant Keypoints. **International Journal of Computer Vision**, v. 60, n. 2, p. 91–110, nov 2004. ISSN 0920-5691. Disponível em: <https://dl.acm.org/citation.cfm?id=993451.996342>. Acesso em: 26 nov. 2023.

LI, H., Wang, W., Xiong, Y., & Lin, D. (2020). Visual feature extraction with convolutional neural networks: a survey. IEEE Transactions on Neural Networks and Learning Systems, 31(9), 3453-3484.

RAO, A.; XU, L.; XIONG, Y.; XU, G.; HUANG, Q.; ZHOU, B.; LIN, D. A 2019. Local-to-Global Approach to Multi-modal Movie Scene Segmentation. Conference on Computer Vision and Pattern Recognition (CVPR), 2020. Disponível em: https://openaccess.thecvf.com/content_CVPR_2020/papers/Rao_A_Local-to-Global-Approac

h-to-Multi-Modal-Movie-Scene-Segmentation-CVPR-2020-paper.pdf. Acesso em: 19 mar. 2023.

LIU, B.; YEUNG, M.; YEO, B.-L.; Segmentation of video by clustering and graph analysis. **Comput. Vis. Image Underst.**, Elsevier Science Inc., New York, NY, USA, v. 71, n. 1, p. 94–109, jul. 1998. ISSN 1077-3142. Disponível em: <http://dx.doi.org/10.1006/cviu.1997.0628>. Acesso em: 26 nov. 2023.

MARQUES, O.; FURHT, B. **Content-Based Image and Video Retrieval.** Springer US, 2002. (Multimedia Systems and Applications). ISBN 9781402070044. Disponível em: <https://books.google.com.br/books?id=ds6xRLdlmJgC>. Acesso em: 26 nov. 2023.

SAHIDULLAH, M.; SAHA, G. Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition. Speech Communication, v. 54, n. 4, p. 543–565, 2012. ISSN 0167-6393. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167639311001622>. Acesso em: 26 nov. 2023.

TERRA.COM.BR. Brasil se destaca no consumo de vídeo em relação à média global. **Terra**, 20 set. 2021. Disponível em: https://www.terra.com.br/noticias/brasil-se-destaca-no-consumo-de-video-em-relacao-a-media-global,dc8f2ddfee59266f24f8a7c1bc38f4d8ln29gaf9.html. Acesso em: 19 mar. 2023.

TROJAHN, T. H. **Um método de segmentação de vídeo em cenas baseado em aprendizagem profunda.** 2019. 129 p. Tese (Doutorado) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

TYAGI, Deepanshu. Introduction to SIFT (Scale Invariant Feature Transform). Medium, 2019. Disponível em: https://medium.com/@deepanshut041/introduction-to-sift-scale-invariant-feature-transform-65d7f3a72d40. Acesso em: 26 nov. 2023.

UNIVERSITY OF BEDFORDSHIRE UNIVERSITY. SQUARE. 2017. **Robust video scene detection using multimodal fusion of optimally grouped features**. Luton, LU1 3JU.United Kingdom. Luton, UK: IEEE, 2017. p. 1–6. ISSN 2473-3628. Disponível em: beds.ac.uk. Acesso em: 19 mar. 2023.

VENDRIG, J.; WORRING, M. Systematic evaluation of logical story unit segmentation. **IEEE Transactions on Multimedia**, Piscataway, NJ, USA, v. 4, n. 4, p. 492–499, dez. 2002. ISSN 1520-9210. Disponível em: <https://ieeexplore.ieee.org/document/1176947/>. Acesso em: 26 nov. 2023.

YOLOv5, github, 2023. **Github.com**. Disponível em: <https://github.com/ultralytics/yolov5?ref=blog.roboflow.com>. Acesso em: 26 nov. 2023.