UNIVERSIDADE DE CAXIAS DO SUL
ÁREA DO CONHECIMENTO DE CIÊNCIAS EXATAS E ENGENHARIAS

MARCOS KLÓSS

AVALIAÇÃO DE MÉTODOS DE ANONIMIZAÇÃO FACIAL EM TEMPO REAL: UM ESTUDO COMPARATIVO ENTRE OFUSCAÇÃO E SÍNTESE

CAXIAS DO SUL

2026

MARCOS KLÓSS

AVALIAÇÃO DE MÉTODOS DE ANONIMIZAÇÃO FACIAL EM TEMPO REAL: UM ESTUDO COMPARATIVO ENTRE OFUSCAÇÃO E SÍNTESE

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de bacharel em Ciência da Computação na Área do Conhecimento de Ciências Exatas e Engenharias da Universidade de Caxias do Sul.

Orientador: Prof. Dr. André Luis Martinotto

CAXIAS DO SUL

2026

MARCOS KLÓSS

AVALIAÇÃO DE MÉTODOS DE ANONIMIZAÇÃO FACIAL EM TEMPO REAL: UM ESTUDO COMPARATIVO ENTRE OFUSCAÇÃO E SÍNTESE

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de bacharel em Ciência da Computação na Área do Conhecimento de Ciências Exatas e Engenharias da Universidade de Caxias do Sul.

Aprovado em 02/07/2026

BANCA EXAMINADORA

Prof. Dr. André Luis Martinotto
Universidade de Caxias do Sul - UCS

Prof. Dr. Guilherme Holsbach Costa
Universidade de Caxias do Sul - UCS

Prof. Dra. Maria de Fatima Webber do Prado Lima
Universidade de Caxias do Sul - UCS

“No broto da flor —
ainda em segredo, dorme
a cor da primavera.”
Matsuo Bashō

RESUMO

O uso indevido ou não autorizado de imagens tornou-se um problema relevante em um cenário no qual a produção de vídeos e transmissões ao vivo se tornaram amplamente acessíveis. Nesse contexto, técnicas automatizadas de anonimização facial apresentam-se como uma solução capaz de operar em tempo real, reduzindo riscos associados ao armazenamento e à disseminação de vídeos não anonimizados, além de representarem um ganho significativo de agilidade em relação à edição de vídeo manual. Este trabalho apresenta uma avaliação comparativa de métodos de anonimização facial com foco na operação em tempo real, analisando e implementando abordagens baseadas em ofuscação e em síntese de faces para a preservação da privacidade. Para a etapa de detecção facial, utilizou-se a arquitetura YOLOv8n pré-treinada integrada ao *framework* OpenCV para processamento de vídeo. O método de ofuscação foi implementado por meio de um filtro gaussiano, enquanto a abordagem por síntese combinou os modelos StyleGAN, para a geração de faces artificiais, e SimSwap, para a transferência de identidade. A avaliação dos métodos ocorreu utilizando-se um subconjunto de clipes de vídeo extraídos da base de dados CelebV-HQ. Os resultados demonstraram que o modelo YOLOv8n obteve bom desempenho, com um tempo médio de detecção de 7,19 milissegundos por quadro. No que se refere à anonimização, a análise revelou que apenas a técnica de ofuscação cumpriu os requisitos de tempo real, atingindo uma taxa média de 14,48 QPS. Em contrapartida, o método por síntese alcançou uma taxa de apenas 1,15 QPS, apresentando também artefatos visuais com aspecto artificial, evidenciando a dualidade entre fotorrealismo e performance.

Palavras-chave: Anonimização facial. Detecção facial. Síntese de faces. Processamento de vídeo em tempo real. Ofuscação.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE ABREVIATURAS E SIGLAS

AdaIN Adaptação Normativa de Instância

ASM *Active Shape Model*

C2f *Cross-Stage Feature Fusion*

CIAGAN *Conditional Identity Anonymization Generative Adversarial Networks*

CNN *Convolutional Neural Networks*

DBNN *Decision Based Neural Network*

GAN *Generative Adversarial Network*

GB *Gigabyte*

GHz *Gigahertz*

GPU *Graphics Processing Unit*

HOG *Histogram of Oriented Gradients*

HSI *Hue Saturation Intensity*

IoU *Intersection over Union*

LDA *Linear Discriminant Analysis*

LFW *Labelled Faces in the Wild*

PCA *Principal Component Analysis*

QPS Quadros por Segundo

RAM *Random Access Memory*

RGB *Red Green Blue*

RNA Rede Neural Artificial

SPPF *Spatial Pyramid Pooling Fast*

SSD *Single Shot Detector*

SVM *Support Vector Machines*

YCbCr *Luma, Blue Chrominance, Red Chrominance*

YOLO *You Only Look Once*

SUMÁRIO

1 INTRODUÇÃO

No Brasil cerca de 86,5% das pessoas com 10 ou mais anos de idade possuem um celular de uso pessoal com acesso a internet (IBGE – Instituto Brasileiro de Geografia e Estatística, 2023). Com a utilização de aparelhos cada vez mais modernos, capazes de gravar vídeo em alta qualidade e até mesmo fazer transmissões ao vivo, o direito de imagem se torna relevante no contexto da privacidade de dados. Ao realizar gravações de vídeo em público torna-se inviável solicitar o consentimento de todos os envolvidos, logo, é necessário preservar a identidade das pessoas, seja por meio de ferramentas de edição, ou através do processamento do vídeo em tempo real (Autoridade Nacional de Proteção de Dados, 2023).

O processamento de vídeo em tempo real é uma tarefa desafiadora devido principalmente às restrições de tempo, pois cada quadro precisa ser processado rapidamente para não comprometer a taxa de transmissão (KECHICHE; TOUIL; OUNI, 2016). Para resolver esse problema, diversas técnicas de detecção e anonimização facial foram desenvolvidas, tendo em vista o equilíbrio entre a precisão e o tempo de execução (SRIVASTAVA *et al.*, 2017).

Dentre as técnicas de reconhecimento facial destacam-se as abordagens baseadas por características e as baseadas por imagens (KUMAR *et al.*, 2019). Em uma abordagem baseada em características, a detecção facial é realizada por meio da identificação de elementos discriminantes, como por exemplo, os olhos, nariz e boca. Esses elementos são extraídos e utilizados para a construção de um modelo estatístico que, a partir da relação entre eles, consegue identificar a presença ou não da face. Essa abordagem apresenta vantagens como a velocidade de detecção e a simplicidade de implementação. No entanto, variações na iluminação e na qualidade da imagem podem degradar a acurácia da predição (KUMAR *et al.*, 2019).

A abordagem baseada em imagens utiliza técnicas de aprendizado de máquina para a geração de modelos capazes de classificar imagens como com ou sem faces. Essas técnicas identificam padrões visuais nos dados de treinamento e, a partir disso, geram funções discriminantes capazes de classificar se uma imagem contém ou não uma face. Essa abordagem apresenta como vantagens uma melhor capacidade de lidar com artefatos de imagem, variações de iluminação e diferentes angulações. No entanto, apresenta um custo computacional mais elevado e uma complexidade de implementação maior (KUMAR *et al.*, 2019).

Após a detecção facial, podem ser aplicadas técnicas de anonimização irreversíveis, sendo os métodos tradicionais geralmente classificados em dois grupos principais: ofuscação e síntese. As técnicas de ofuscação atuam degradando a qualidade da imagem para impedir a visualização da face. Entre as abordagens de ofuscação mais comuns estão o mascaramento (CHINOMI *et al.*, 2008), desfoque (SARWAR; CAVALLARO; RINNER, 2018) e distorção (KORSHUNOV; EBRAHIMI, 2013). Por outro lado, as técnicas de síntese baseiam-se na geração de rostos

sintéticos, garantindo a privacidade sem comprometer a estética da imagem (ANDRADE, 2023).

1.1 OBJETIVOS

O objetivo geral deste trabalho é desenvolver e avaliar uma solução capaz de anonimizar faces em vídeos, buscando garantir a privacidade dos indivíduos sob os requisitos de tempo real. Para atingir esse propósito, foram definidos os seguintes objetivos específicos:

1. Definir a técnica de detecção facial a ser utilizada;
2. Definir as técnicas de anonimização facial a serem implementadas;
3. Desenvolver um algoritmo para detecção e anonimização facial em vídeos;
4. Realizar os testes experimentais e analisar comparativamente os resultados obtidos.

1.2 ESTRUTURA DO TRABALHO

Este trabalho está estruturado em seis capítulos. Neste primeiro capítulo, foi realizada a apresentação do tema, descrevendo a motivação e os objetivos da pesquisa. O Capítulo 2 apresenta uma descrição do problema da anonimização facial no contexto de tempo real, além de expor o levantamento dos trabalhos relacionados. Em seguida, o Capítulo 3 aborda os conceitos fundamentais do modelo de detecção de objetos *You Only Look Once* (YOLO) e da rede generativa de síntese de faces StyleGAN. A definição da solução e a implementação desenvolvida, englobando as etapas de detecção e as abordagens de anonimização facial, são detalhadas no Capítulo 4. No Capítulo 5, são apresentados e analisados de forma comparativa os resultados experimentais obtidos. Por fim, o Capítulo 6 encerra o estudo com as considerações finais e as propostas de trabalhos futuros.

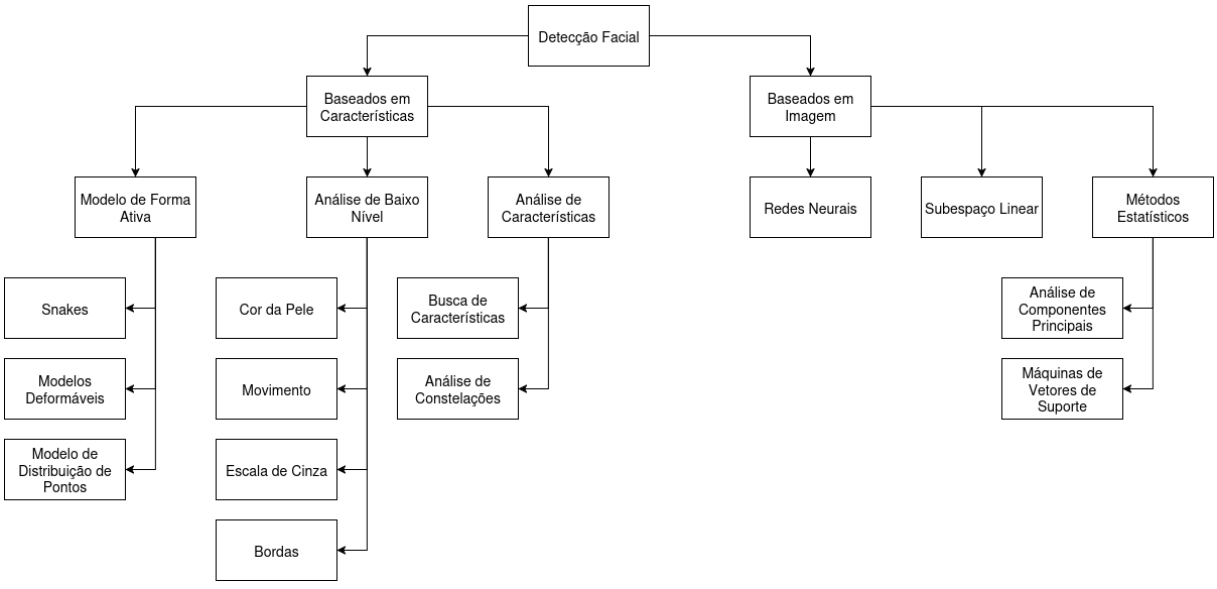
2 ANONIMIZAÇÃO FACIAL EM TEMPO REAL

O desempenho de um sistema de anonimização facial em tempo real está diretamente ligado à etapa de detecção facial. De fato, antes de aplicar qualquer técnica de anonimização, é necessário que a face seja corretamente detectada e delimitada na imagem. Assim, a capacidade de detectar as faces com precisão é fundamental para garantir a eficiência de um modelo de anonimização facial (ANDRADE, 2023).

2.1 DETECÇÃO FACIAL

De acordo com Kumar *et al.* (2019) as técnicas de detecção facial podem ser classificadas em duas categorias: abordagens baseadas em características e abordagens baseadas em imagem (Figura 1). As abordagens baseadas em características utilizam métodos tradicionais de visão computacional, aplicando filtros e técnicas específicas para detectar os elementos discriminantes, como por exemplo, olhos, nariz e boca. Por outro lado, na abordagem por imagem são utilizados métodos de aprendizado de máquina, como as redes neurais convolucionais (CNNs do inglês *Convolutional Neural Networks*) (KUMAR *et al.*, 2019).

Figura 1 – Técnicas de detecção facial.



Fonte: Adaptado de Kumar *et al.* (2019)

2.1.1 Abordagens Baseadas em Características

Essa abordagem é subdividida em três grupos: modelos de forma ativa (ASM - *Active Shape Model*), análise de baixo nível e análise de características. No ASM, a ênfase encontra-se nas características físicas, como o contorno da face, olhos, nariz e boca. Na análise de baixo nível a detecção é realizada através de informações dos *pixels*, como cor, bordas e movimento em vídeos. Por fim, na análise de características são consideradas a estrutura e a geometria facial para a localização das faces nas imagens (HASAN *et al.*, 2021).

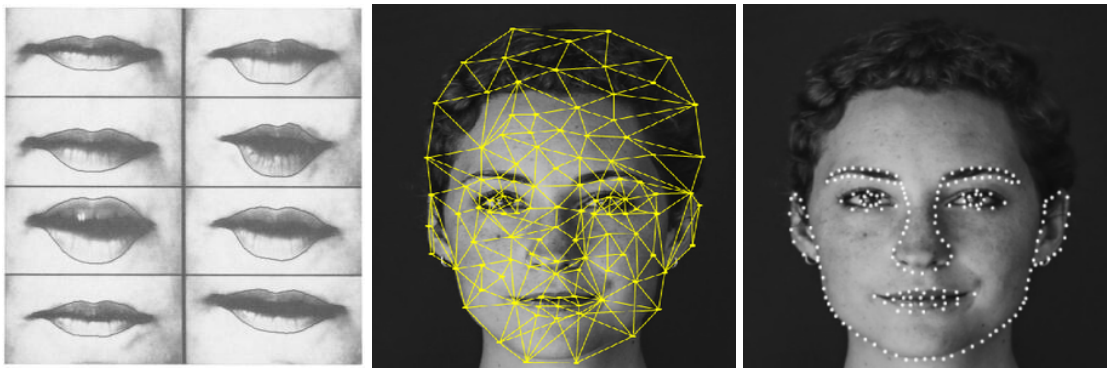
Na classe dos modelos ASM encontram-se métodos que necessitam de intervenções manuais, seja na inicialização ou durante o processo de treinamento. Entre esses métodos destacam-se:

- *Snakes*: esse método foi proposto por Kass, Witkin e Terzopoulos (1988) e tem como objetivo detectar objetos por meio de uma curva que se ajusta aos seus contornos (Figura 2a). Esse método baseia-se na minimização de uma função que combina informações de intensidade da imagem, como bordas e variações locais associadas às feições faciais. Dessa forma, o contorno ajusta-se progressivamente à estrutura da imagem, permitindo a delimitação da região da face. O *Snakes* apresenta como principal vantagem a facilidade de implementação, mas requer uma inicialização próxima à característica de interesse, podendo se prender a contornos falsos ou indesejados.

- *Modelos Deformáveis*: esse modelo, proposto por Yuille, Hallinan e Cohen (1992), aprimora o *Snakes* ao incorporar características globais da face, como olhos, nariz e boca. Diferentemente do *Snakes*, o método utiliza um conjunto de polígonos predefinidos que são ajustados para delimitar a face (Figura 2b). Nesse contexto, cada característica facial é representada por um polígono específico, cuja posição é ajustada com base na geometria da face.

- *Modelo de Distribuição de Pontos*: essa técnica foi proposta por Cootes e Taylor (1992) e, posteriormente, foi aplicada à detecção facial por Lanitis, Cootes e Taylor (1994). Esse método gera um modelo através de imagens anotadas com pontos de referência (Figura 2c). Em seguida, o modelo é posicionado nas proximidades da face, sendo utilizado como referência para a busca da mesma. Esse é um método mais robusto, porém, a criação do conjunto de treinamento requer a anotação manual dos pontos nas imagens.

Figura 2 – Modelos *Snakes*, Modelos Deformáveis e Modelo de Distribuição de Pontos.



(a) *Snakes* (b) Modelos Deformáveis (c) Modelo de Distribuição de Pontos

Fonte: (a) adaptado de Kass, Witkin e Terzopoulos (1988); (b) e (c) adaptados de Hasan *et al.* (2021)

Na Análise de Baixo Nível encontram-se os métodos que extraem descritores fundamentais da imagem, como bordas, cores e intensidade de brilho. Nessa classe encontram-se os métodos que utilizam:

- *Cor da pele*: esse modelo foi proposto por Crowley e Coutaz (1996) e utiliza um histograma de cores normalizado e um limiar predefinido para determinar se um *pixel* pertence a um tom de pele (Figura 3a). Essa técnica pode ser utilizada em diferentes espaços de cores, como RGB (*Red Green Blue*), HSI (*Hue Saturation Intensity*) ou YCbCr (*Luma, Blue Chrominance, Red Chrominance*). A principal vantagem dessa técnica é a sua rapidez, porém essa apresenta uma alta sensibilidade a mudanças de iluminação, visto que essas mudanças afetam diretamente a percepção da cor.

- *Movimento*: esse método realiza uma análise quadro a quadro para a detecção das faces em vídeos. O processo baseia-se na detecção do movimento do corpo através da utilização de filtros espaço-temporais, que possibilitam segmentar o objeto de interesse em relação ao plano de fundo (Figura 3b). A partir disso, são utilizadas heurísticas que procuram identificar a presença da cabeça no topo do corpo. Essa técnica é mais adequada quando o fundo é estático e o objeto de interesse está em movimento. Além disso, outras informações como os contornos e a posição dos olhos podem ser utilizadas durante a análise (HASAN *et al.*, 2021).

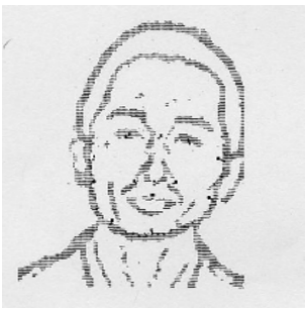
- *Escala de cinza*: as imagens em tons de cinza representam a intensidade de brilho de cada *pixel*, e podem ser utilizadas para localizar regiões faciais, uma vez que as regiões como sobrancelhas, pupilas e lábios tendem a ser mais escuras (Figura 3c). Assim, algumas implementações aplicam realce de contraste, seguido da busca por mínimos locais, para extrair essas características (HJELMÅS; LOW, 2001). Outras técnicas procuram regiões mais claras, como a ponta do nariz. Também é possível utilizar imagens em baixa resolução

para a identificação de áreas mais uniformes que podem conter a face. Posteriormente, essas áreas podem ser refinadas em resoluções mais altas (YANG; HUANG, 1994).

- *Bordas*: proposto por Sakai (1972), esse método baseia-se na detecção de bordas para extrair os contornos faciais (Figura 3d). Para a detecção das bordas tradicionalmente são utilizados filtros, como por exemplo, *Sobel*, *Laplaciano* e *Marr-Hildreth* (HASAN *et al.*, 2021). As informações obtidas a partir da aplicação desses filtros são utilizadas como entrada para classificadores que determinam a presença ou não de uma face.

Figura 3 – Modelos Cor da pele, Movimento, Escala de cinza e Bordas.

(a) *Cor da pele* (b) Movimento (c) Escala de cinza



(d) Bordas

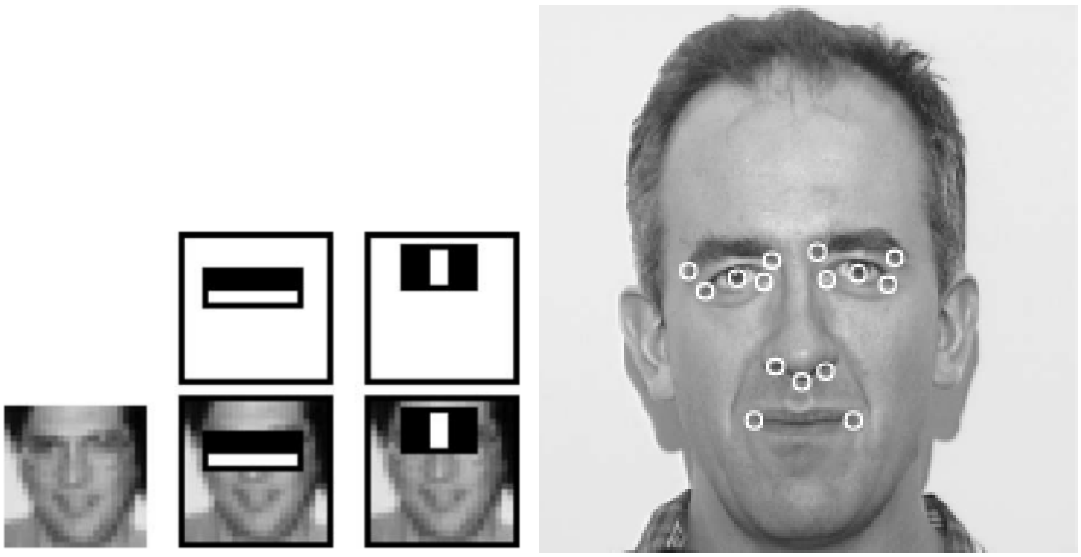
Fonte: (a) adaptado de Kumar *et al.* (2019); (b) adaptado de Beek *et al.* (1992); (c) adaptado de Yang e Huang (1994); (d) adaptado de Sakai (1972)

Na Análise de Características, encontram-se métodos que se baseiam nas características faciais e nas relações geométricas entre essas características para realizar a detecção e a delimitação da face em imagens. Esses métodos podem ser agrupados em:

- *Busca de características*: tem como objetivo identificar as características faciais mais proeminentes e, posteriormente, localizar as demais estruturas menos evidentes com base em relações e proporções geométricas. Neste caso, os olhos costumam ser a principal característica devido à sua disposição simétrica e facilidade de detecção (HJELMÅS; LOW, 2001). Um dos métodos mais conhecidos dessa classe foi desenvolvido por Viola e Jones (2004), o qual utiliza filtros *Haar* para a extração das características, o algoritmo *AdaBoost* para seleção das características mais relevantes e classificadores em cascata para a detecção da face (Figura 4a).

- *Análise de constelações*: esses métodos baseiam-se nas relações existentes entre as características faciais, como por exemplo, as distâncias relativas e o posicionamento das mesmas (HJELMÅS; LOW, 2001). Entre esses métodos, destaca-se o modelo proposto por Burl *et al.* (1995), onde detectores locais são aplicados para identificar possíveis elementos faciais, como olhos, nariz e boca. As combinações desses pontos são avaliadas por um modelo estatístico, que considera as relações geométricas entre eles para a classificação entre face ou não-face.

Figura 4 – Métodos de Busca de características e Análise de constelações.



(a) Implementação de Viola e Jones (2004) do método Busca de características

(b) Análise de constelações

Fonte: (a) adaptado de Viola e Jones (2004); (b) adaptado de Burl *et al.* (1995)

2.1.2 Abordagens Baseadas em Imagem

Essa abordagem incorpora os métodos mais recentes de detecção facial e tradicionalmente são organizados em: redes neurais, métodos estatísticos e métodos baseados em subespaço linear (HASAN *et al.*, 2021).

As redes neurais são algoritmos inspirados no funcionamento dos neurônios biológicos, sendo projetadas para reconhecer padrões por meio de um processo de treinamento. Nesse processo, os pesos das conexões entre os neurônios são ajustados iterativamente de modo a reduzir o erro entre a saída prevista e a saída desejada. Esse processo de treinamento é realizado até que o modelo seja capaz de generalizar e realizar predições em novos dados. Neste contexto, as redes neurais podem ser classificadas em três categorias principais: redes neurais artificias (RNAs), redes neurais baseadas em decisão e redes neurais difusas (HASAN *et al.*, 2021).

- *Redes neurais artificiais*: são compostas por neurônios artificiais organizados em camadas. A implementação considerada mais simples dessa arquitetura de é a *perceptron* multicamadas, na qual os dados percorrem a rede em apenas uma direção. O treinamento é realizado de forma supervisionada, sendo aplicado um algoritmo de retropropagação de erro para o ajuste dos pesos (HASAN *et al.*, 2021). Outra arquitetura relevante no contexto de visão computacional e detecção facial são as CNNs (*Convolutional Neural Networks*), as quais utilizam camadas de convolução para extrair características visuais. Esse processo permite capturar padrões espaciais complexos da imagem, permitindo alcançar um bom desempenho em tarefas de classificação de imagens e detecção de objetos (LI *et al.*, 2022).

- *Redes neurais baseadas em decisão*: as redes do tipo DBNN (*Decision Based Neural Network*), foram utilizadas pela primeira vez em problemas de classificação de imagens por Kung e Taur (1995). Trata-se de uma rede neural supervisionada orientada por decisões de classificação, onde no processo de treinamento as fronteiras de decisão são ajustadas com base no erro ou acerto do modelo. O aprendizado acontece reforçando os pesos vinculados à predições corretas e atenuando os vinculados à incorretas. Além disso, sua arquitetura combina funções discriminantes e uma estrutura de decisão hierárquica para lidar com problemas de maior complexidade (KUNG; TAUR, 1995).

- *Redes neurais difusas*: essas combinam características da lógica difusa[1] e das redes neurais artificiais, aproveitando a capacidade da lógica difusa de lidar com incertezas e imprecisões. Nessa arquitetura, as entradas e saídas processadas pelos neurônios estão vinculadas à graus de pertinência, representados por valores no intervalo de $[0, 1]$. Desta forma, o uso da lógica difusa permite modelar a incerteza e a imprecisão, visto que uma entrada pode pertencer a mais de uma classe, com diferentes níveis de intensidade (BHUTANI; FARSAIE, 1991).

Entre os métodos estatísticos, existem algoritmos generalistas que podem ser aplicados à detecção facial, destacando-se:

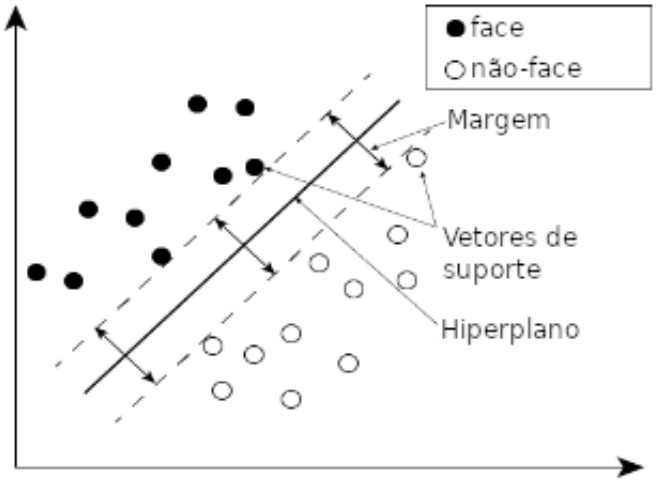
- *Análise de Componentes Principais*: esse algoritmo foi proposto por Hotelling (1933) e aplicado à detecção facial por Turk e Pentland (1991). A PCA (*Principal Component Analysis*) busca reduzir a dimensionalidade dos dados através de uma transformação ortogonal, convertendo as variáveis correlacionadas em um conjunto de variáveis não correlacionadas, denominadas componentes principais. No contexto da detecção facial, a PCA

[1] A lógica difusa, introduzida por Zadeh (1965), constitui uma extensão da lógica clássica, permitindo a representação de conceitos vagos ou imprecisos, especialmente no campo do reconhecimento de padrões. Nessa abordagem, cada objeto de estudo é associado a um grau de pertinência em relação a uma determinada classe, expresso por um valor no intervalo $[0, 1]$. Por exemplo, valores próximos de 1 indicam maior pertencimento do objeto à classe em questão. Esse grau de pertinência é definido por meio de uma função de pertinência, cujo objetivo é quantificar o nível de associação entre o objeto e a classe, possibilitando o estabelecimento de uma hierarquia entre os elementos.

pode ser utilizada para extrair as principais características faciais de uma imagem. Em seguida, essas características podem ser utilizadas como entrada para um classificador responsável por distinguir entre face e não-face.

- *Máquinas de Vetores de Suporte*: o método SVM (*Support Vector Machines*) foi proposto por Boser, Guyon e Vapnik (1992). Neste método, durante a etapa de treinamento, são extraídas características da imagem que são utilizadas para ajustar um modelo cujo objetivo é definir um hiperplano capaz de separar as classes face e não-face. Posteriormente, esse hiperplano é utilizado para a classificação de novas imagens, determinando a classe da imagem com base no lado em que ela se posiciona em relação ao hiperplano (Figura 5).

Figura 5 – Diagrama da aplicação do SVM.



Fonte: Adaptado de Hasan *et al.* (2021)

Os métodos baseados em subespaço linear são caracterizados pela capacidade de representar estruturas complexas em um espaço de baixa dimensionalidade. Entre esses métodos destacam-se:

- *Eigenfaces*: esse método foi proposto por Turk e Pentland (1991) e parte do princípio de que, mesmo diante de variações de pose, iluminação e rotação, as faces ainda apresentam padrões estruturais característicos, como a presença dos olhos, nariz e boca. Essas características são denominadas *eigenfaces* e são extraídas de um conjunto de imagens de treinamento utilizando uma técnica de PCA. As *eigenfaces* coletadas durante o treinamento são chamadas de espaço de faces, e se assemelham a média das faces do conjunto de treinamento (Figura 6). Assim, na classificação de uma nova imagem, um conjunto de *eigenfaces* é extraído e esse é comparado ao espaço de faces para realizar a identificação entre face e não-face.

Figura 6 – Espaço de faces resultante do treinamento no método *Eigenfaces*.

(a) Imagens de treinamento (b) Espaço de faces

Fonte: Adaptado de Turk e Pentland (1991)

- *Eigenspaces* Probabilísticos: esse método foi proposto por Moghaddam e Pentland (1997) e consiste em uma extensão do modelo de *Eigenfaces*. Diferentemente da abordagem tradicional, que apenas compara a nova imagem com o espaço de faces, esse método incorpora uma abordagem estatística baseada em estimativas de densidade de probabilidade, tornando o método mais robusto a ruídos, variações e à presença de oclusões parciais na face (MOGHADDAM; PENTLAND, 1997).

- *Fisherfaces*: esse foi método proposto por Belhumeur, Hespanha e Kriegman (1997) e utiliza o método de projeção LDA (*Linear Discriminant Analysis*) de Fisher[2] para obter uma representação que maximize a separação entre as classes. A técnica LDA reduz a dimensionalidade ao projetar os dados em um espaço de características no qual a dispersão entre classes é maximizada e a intra-classe é minimizada, resultando em uma melhor separação das classes.

2.2 ANONIMIZAÇÃO FACIAL

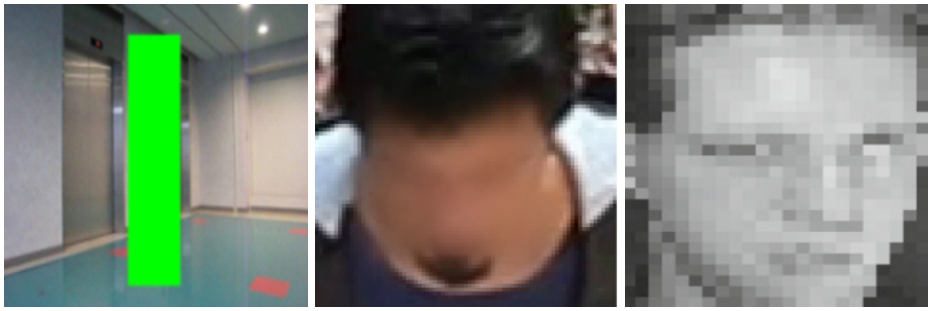
Segundo Meden *et al.* (2021), os métodos de anonimização facial podem ser divididos em dois grupos: ofuscação e síntese. As técnicas de ofuscação consistem em métodos simples que degradam a qualidade da imagem na área onde a face está localizada (ANDRADE, 2023). Entre essas técnicas destacam-se:

- *Máscara*: consiste em ocultar a região desejada com formas geométricas, que podem incluir retângulos, elipses ou círculos de cor sólida, entre outras possibilidades (Figura 7a).

[2] Proposto por Fisher (1936) e amplamente utilizado em problemas de reconhecimento de padrões

- *Desfoque*: afeta o nível de detalhe da região facial por meio da aplicação de filtros de suavização, como por exemplo, os filtros média, mediana ou gaussiano (Figura 7b).
- *Pixelização*: reduz a resolução da área do rosto agrupando *pixels* semelhantes em blocos quadrados ou retangulares (Figura 7c) (FAN, 2018).
- *Deformação*: consiste em uma transformação geométrica que distorce a imagem. Nesta técnica, a posição de cada *pixel* é alterada e, posteriormente, é aplicada uma interpolação para ajustar as intensidades (Figura 7d).

Figura 7 – Técnicas de ofuscação.



(a) Máscara (b) Desfoque (c) Pixelização

(d) Deformação

Fonte: (a) adaptado de Chinomi *et al.* (2008); (b) adaptado de Sarwar, Cavallaro e Rinner (2018); (c) adaptado de Fan (2018); (d) adaptado de Korshunov e Ebrahimi (2013)

As técnicas de síntese representam abordagens mais sofisticadas, nas quais as faces são substituídas por faces geradas artificialmente (Figura 8). Esse processo permite substituir as características faciais reais por versões sintéticas, mantendo a aparência geral da imagem e garantindo a anonimidade dos indivíduos (ANDRADE, 2023). Nesse contexto destacam-se as redes generativas adversariais (GANs do inglês *Generative Adversarial Network*s) e métodos da família *k-Same* (ANDRADE, 2023).

A implementação de uma GAN é baseada em duas redes neurais artificiais: uma geradora, responsável por produzir as imagens sintéticas que se assemelham aos dados reais; e uma discriminadora, treinada para distinguir entre os dados reais e sintéticos. O treinamento é realizado até que a rede geradora consiga produzir imagens que a discriminadora não consiga identificar como sintéticas. Algumas implementações de GANs utilizam informações pre-

sentes na imagem ou inseridas via parâmetros para a sintetização de faces. Por exemplo, no modelo *Deep Privacy*, informações relativas ao plano de fundo e pose foram incorporadas à rede geradora para aumentar o realismo das faces sintetizadas (Figura 8a) (HUKKELÅS; MESTER; LINDSETH, 2019). Já o modelo CIAGAN (*Conditional Identity Anonymization Generative Adversarial Networks*) possibilita que características como silhueta, boca e nariz possam ser fornecidas como entrada à rede geradora, possibilitando a geração de faces com características predefinidas (Figura 8b) (MAXIMOV; ELEZI; LEAL-TAIXE, 2020).

Figura 8 – Modelos de síntese do tipo GAN.

(a) Modelo *Deep Privacy*

(b) Modelo CIAGAN

Fonte: (a) adaptado de Hukkelås, Mester e Lindseth (2019); (b) adaptado de Maximov, Elezi e Leal-Taixe (2020)

Os métodos da família *k-Same* combinam características de diferentes rostos para a geração de uma face sintética que não corresponda a nenhuma das faces utilizadas, mas que mantenha traços realistas. Abordagens como *k-Same-Pixel* e *k-Same-Eigen* geram faces sintéticas ao combinar rostos semelhantes através de média dos *pixels* ou projeção via PCA, respectivamente. Já os métodos *k-Same-Select* e *k-Same-M* combinam forma e textura facial para gerar faces sintéticas mais realistas e com menos artefatos visuais (ANDRADE, 2023).

2.3 TRABALHOS RELACIONADOS

A evolução dos sistemas de anonimização facial em vídeo é marcada pelo avanço das técnicas de detecção e anonimização facial. Para a detecção facial, os estudos demonstram uma evolução de modelos estatísticos em favor de modelos baseados em aprendizado de máquina. Já para a anonimização, nota-se a transição de técnicas de aplicação de filtros matemáticos para o uso de redes neurais, convergindo para modelos de síntese facial que buscam preservar as identidades e manter a qualidade visual do vídeo.

No trabalho de Newton, Sweeney e Malin (2005) foi proposto um sistema de anonimização facial voltado à proteção da identidade em vídeos de câmeras de segurança. Para a detecção facial, é utilizado o método *Eigenfaces* e a anonimização é realizada por meio do método *k-Same*. Os testes foram realizados utilizando a base de dados *FERET* (PHILLIPS *et al.*, 2000).

O trabalho de Gross *et al.* (2006) apresenta um sistema de anonimização facial baseado na geração de rostos sintéticos. A detecção facial é realizada por meio do método Modelos de Aparência Ativa, uma técnica do tipo ASM derivada do Modelo de Distribuição de Pontos (COOTES; EDWARDS; TAYLOR, 2001). Para a anonimização, é utilizado o método *k-Same-M*. De acordo com os autores, em testes realizados na base de dados CMU Multi-PIE (GROSS *et al.*, 2006), o método desenvolvido apresentou melhores resultados que os métodos de ofuscação.

No trabalho desenvolvido por Gross e Sweeney (2007) a detecção facial é realizada utilizando o método Modelos de Aparência Ativa. Para a anonimização facial foram utilizados os métodos ϵ*-map* e *(*ϵ*, k)-map*, que são derivados do k-*Same*. Os testes foram realizados utilizando a base de dados CMU Multi-PIE (GROSS *et al.*, 2006). Segundo os autores, os métodos propostos demonstraram uma qualidade visual superior em comparação a outras abordagens, como pixelização ou desfoque.

No trabalho de Mrityunjay e Narayanan (2011) foi desenvolvida uma solução embarcada em uma câmera para a anonimização facial em tempo real. O plano de fundo é removido por meio de um modelo de misturas gaussianas (ZIVKOVIC, 2004). O método HOG (*Histogram of Oriented Gradients*) (DALAL; TRIGGS, 2005) foi utilizado para a extração de características e um classificador SVM foi empregado para a detecção facial. Além disso, um rastreamento da face é realizado utilizando o método *Camshift* (KWOLEK, 2005), evitando a necessidade de uma nova detecção a cada quadro. A anonimização é feita através de uma técnica de desfoque. Os testes em vídeos demonstraram um desempenho de até 10 quadros por segundo.

No trabalho desenvolvido por Won *et al.* (2019) foi apresentada uma otimização de uma CNN com arquitetura YOLOv3 (REDMON; FARHADI, 2018) para a detecção facial em vídeos de segurança. O modelo otimizado possui menos camadas que a YOLOv3 tradicional, o que resultou em uma maior velocidade. A anonimização facial é realizada por meio da técnica de desfoque. Em testes utilizando a base WIDER FACE (YANG *et al.*, 2016), o modelo alcançou uma precisão média de 87,48% e 100,5 quadros por segundo.

O estudo desenvolvido por Angus *et al.* (2022) apresenta uma solução para anonimização facial em tempo real em vídeos de segurança pública. A detecção facial é realizada através de uma CNN com arquitetura YOLOv4 (BOCHKOVSKIY; WANG; LIAO, 2020). A anonimização é feita por meio da técnica de desfoque. Para a atender a demanda de tempo real, o sistema foi desenvolvido para ser executado em GPUs e utilizando o *framework* NVIDIA TensorRT[3]

[3] Um *framework* de aprendizado de máquina publicado pela Nvidia para executar inferências de aprendizado de máquina em suas GPUs.

(NVIDIA, 2022). Em testes realizados na plataforma *COSMOS* (RAYCHAUDHURI *et al.*, 2020), o sistema atingiu *recall* de 98% e taxas superiores a 60 quadros por segundo em uma placa gráfica Nvidia A100.

No trabalho desenvolvido por Kim *et al.* (2022) foi proposto um sistema de anonimização facial em tempo real para vídeos de vigilância. A detecção facial foi realizada utilizando uma CNN do tipo YOLOv4 (BOCHKOVSKIY; WANG; LIAO, 2020) e a anonimização foi feita por desfoque. Para melhorar o desempenho do sistema foi utilizado a arquitetura YOLOv4-*tiny* acelerada por GPU. Em testes realizados com a base de dados WIDER FACE (YANG *et al.*, 2016), o modelo atingiu uma precisão média de 62,21% e desempenho superior a 60 quadros por segundo.

No trabalho desenvolvido por Jaichuen *et al.* (2023) é proposto um sistema para anonimização facial em vídeos de monitoramento em tempo real. Neste, foram realizados testes utilizando dois modelos pré-treinados de CNNs: *RetinaFace* (DENG *et al.*, 2020) e *YOLOv5Face* (QI *et al.*, 2023). A anonimização foi realizada utilizando a técnica de pixelização. Para testes foi utilizada a base de dados *NTU CCTV-Fights Dataset* (PEREZ; KOT; ROCHA, 2019). O modelo *RetinaFace* alcançou uma acurácia de 72,5%, *recall* de 71,2% e tempo médio de 2 segundos por quadro. Já o modelo *YOLOv5Face* obteve uma acurácia de 38,1%, *recall* de 31,8% e tempo de 0,218 segundos por quadro.

No trabalho de Tran *et al.* (2024) foi desenvolvido um sistema de anonimização facial com base na técnica de síntese. A detecção facial é realizada por uma CNN com arquitetura *SE-ResNeXt50*, que também extrai informações de idade e gênero. Para a anonimização foi utilizado o modelo *StyleGAN2* (KARRAS *et al.*, 2020). A substituição da face é feita através de uma técnica de *SimSwap* (CHEN *et al.*, 2020), que preserva pose e expressões faciais. O sistema apresentou um desempenho de aproximadamente 17 quadros por segundo em testes realizados com as bases CelebA-HQ (KARRAS *et al.*, 2018a) e *Labelled Faces in the Wild* (LFW) (HUANG *et al.*, 2007).

No estudo realizado por Lei *et al.* (2024) foi proposto um sistema de anonimização facial voltado para ambientes aeroportuários. A detecção facial é realizada através de uma CNN do tipo SSD (do inglês *Single Shot Detector*) (LIU *et al.*, 2015). A anonimização é feita por meio de síntese utilizando o modelo *StyleGAN* (KARRAS; LAINE; AILA, 2021). Segundo os autores, os testes realizados nas bases CelebA-HQ (KARRAS *et al.*, 2018a) e LFW (HUANG *et al.*, 2007) apresentaram bons resultados visuais e eficácia na preservação da privacidade.

No trabalho desenvolvido por More *et al.* (2024) foi proposto um sistema de anonimização facial baseado em síntese através de uma abordagem GAN. Neste trabalho, a detecção facial é realizada através de uma CNN do tipo YOLOv5 (Ultralytics, 2024) e a anonimização é realizada utilizando a StyleGAN3 (KARRAS *et al.*, 2021). Em testes com a base FFHQ (KARRAS; LAINE; AILA, 2021), foi obtida precisão média de 86%.

2.3.1 Definição das Técnicas de Detecção e Anonimização Facial

O Quadro 1 apresenta uma sistematização das diferentes técnicas empregadas nos trabalhos relacionados. Observa-se que, ao longo do tempo, os métodos de detecção facial evoluíram de abordagens baseadas em características para abordagens baseadas em imagem, destacando-se o uso crescente de variações de CNNs com arquitetura YOLO (*You Only Look Once*). Desta forma, neste trabalho será utilizado um modelo YOLO para detecção facial devido à sua ampla adoção em pesquisas recentes e pela disponibilidade de modelos pré-treinados específicos para faces. O Quadro 1 apresenta diferentes variantes utilizadas em detecção facial, sendo a YOLOv5 a mais recente dentre as listadas. No entanto, neste trabalho será utilizada a YOLOv8, por se tratar de uma evolução direta da YOLOv5, incorporando melhorias de desempenho e precisão (WANG; LIAO, 2024).

Em relação às técnicas de anonimização facial, observa-se o uso tanto de métodos baseados em ofuscação quanto em síntese. Nos últimos anos, entretanto, as abordagens de síntese têm se tornado predominantes, especialmente aquelas baseadas no modelo StyleGAN. Neste trabalho, dois métodos de anonimização facial serão implementados: ofuscação e síntese. O método de ofuscação será implementado usando um filtro gaussiano, utilizado nos trabalhos de Angus *et al.* (2022) e Mrityunjay e Narayanan (2011). Já o método de síntese será implementado através do modelo StyleGAN, modelo amplamente utilizada em estudos recentes.

Quadro 1 – Trabalhos relacionados, técnicas e bases de dados utilizadas

Trabalho	**Técnica de Detecção**	**Técnica de Anonimização**	**Base de Dados**
Newton, Sweeney e Malin (2005)	Eigenfaces	k-Same	FERET
Gross *et al.* (2006)	Modelos de Aparência Ativa	k-Same-M	CMU Multi-PIE
Gross e Sweeney (2007)	Modelos de Aparência Ativa	ϵ-map, (ϵ,k)-map	CMU Multi-PIE
Mrityunjay e Narayanan (2011)	HOG e SVM	Desfoque	Base de dados própria
Won *et al.* (2019)	YOLOv3-39	Desfoque	*WIDER FACE*
Angus *et al.* (2022)	YOLOv4	Desfoque	COSMOS
Kim *et al.* (2022)	YOLOv4-tiny	Desfoque	*WIDER FACE*
Jaichuen *et al.* (2023)	RetinaFace, YOLOv5Face	Pixelização	NTU CCTV-Fights
Tran *et al.* (2024)	SE-ResNeXt50	StyleGAN2	CelebA-HQ LFW
Lei *et al.* (2024)	SSD	StyleGAN	CelebA-HQ LFW
More *et al.* (2024)	YOLOv5	StyleGAN3	FFHQ

3 MODELOS DE DETECÇÃO E ANONIMIZAÇÃO FACIAL

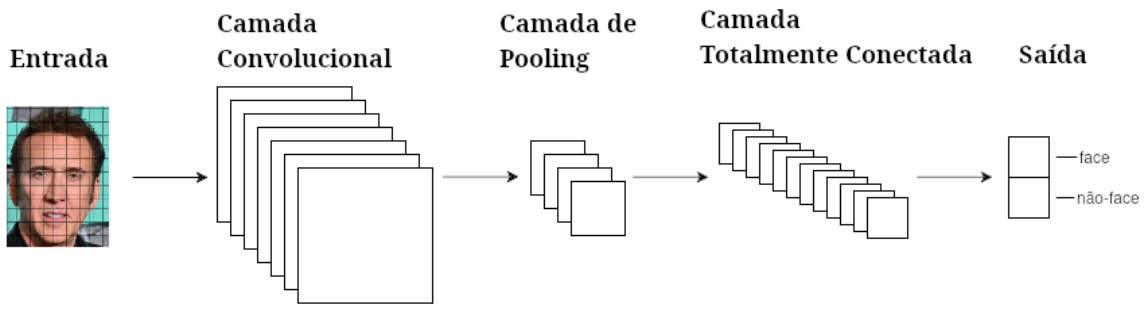
O emprego de modelos baseados em aprendizado de máquina para detecção e anonimização facial tem se tornado cada vez mais recorrente. As redes neurais convolucionais, em particular, consolidaram-se como elementos centrais em diversas tarefas de visão computacional (LI *et al.*, 2022). Neste capítulo, serão apresentadas as CNNs empregadas neste trabalho para a tarefa de detecção facial, com ênfase na arquitetura YOLOv8. Na sequência, serão introduzidas as redes generativas adversariais, com destaque para a StyleGAN, adotada neste estudo para a anonimização por síntese de faces.

3.1 REDES NEURAIS CONVOLUCIONAIS

As CNNs destacam-se como uma das arquiteturas mais relevantes no campo de aprendizado profundo, sendo amplamente utilizadas em diferentes áreas (LI *et al.*, 2022). No contexto da visão computacional, seu emprego em tarefas de detecção facial tem se tornado predominante (JIANG; LEARNED-MILLER, 2017). A característica principal de uma CNN é a capacidade de extrair atributos representativos por meio de operações de convolução.

A Figura 9 apresenta a arquitetura básica de uma CNN, onde a camada de convolução é responsável por identificar padrões locais da imagem, resultando em um mapa de características que preserva somente as informações mais relevantes. Em seguida, a camada de *pooling* reduz a dimensionalidade do mapa de características, descartando informações menos relevantes. Por fim, a camada totalmente conectada utiliza os atributos extraídos ao longo da rede para realizar a classificação. Em geral, uma CNN pode ser composta por diversas camadas convolucionais e de *pooling* organizadas de forma hierárquica, permitindo uma extração progressiva de atributos (LI *et al.*, 2022).

Figura 9 – Arquitetura de uma CNN.

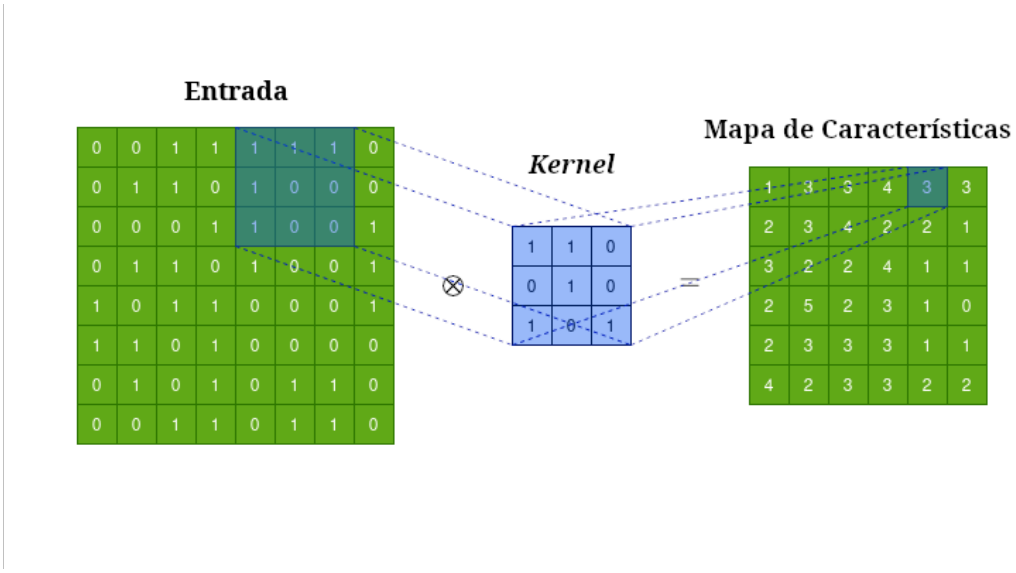


Fonte: Adaptado de O'Shea e Nash (2015)

3.1.1 Camada Convolucional

A camada convolucional é o principal componente de uma CNN, sendo responsável pela extração de características da imagem de entrada. O processo de convolução consiste na aplicação de um filtro (*kernel*) que percorre a imagem, produzindo como resultado uma matriz denominada mapa de características. O *kernel* é uma matriz de valores numéricos que é deslocada sobre a imagem. Conforme pode ser observado na Figura 10, é realizada a multiplicação elemento a elemento entre os coeficientes do *kernel* e os valores dos *pixels* correspondentes. O somatório desses produtos é atribuído à posição correspondente no mapa de características.

Figura 10 – Processo de convolução em uma CNN.

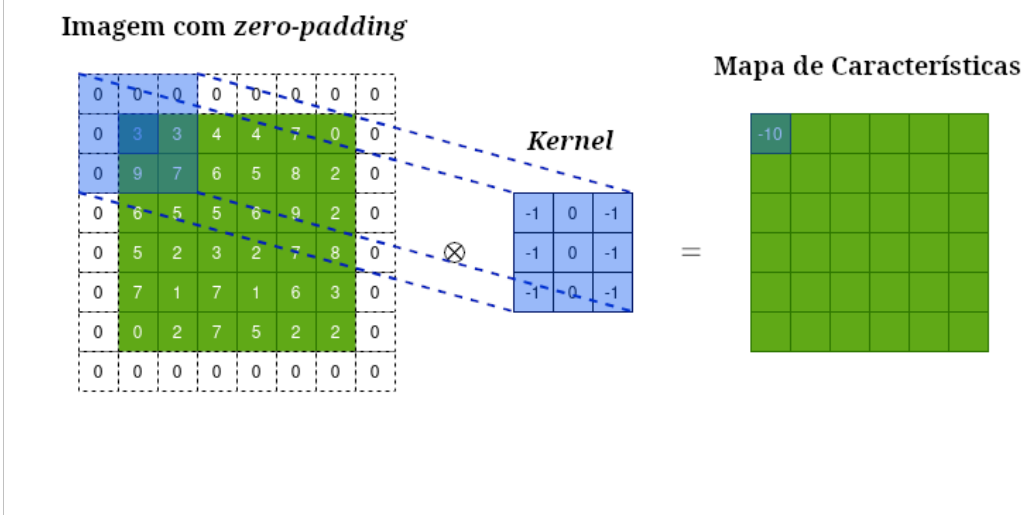


Fonte: O Autor (2025).

A quantidade de *pixels* que o *kernel* avança a cada deslocamento é denominada *stride*. A utilização de um *stride* menor faz com que as regiões de aplicação do filtro se sobreponham significativamente, resultando em um mapa de características maior e mais detalhado. No entanto, esse processo aumenta o custo computacional. Por outro lado, a adoção de um *stride* maior reduz a sobreposição entre as regiões analisadas, gerando mapas de características menores, menos detalhados e com menor custo computacional (O'SHEA; NASH, 2015).

A Figura 10 ilustra o processo de geração do mapa de características, que apresenta dimensões reduzidas em relação à imagem de entrada. Isso ocorre porque cada operação de convolução do *kernel* gera um único valor de saída. Os dados perdidos nesse processo correspondem as bordas da imagem. Para evitar essa perda de informação e preservar a dimensionalidade do mapa de características, aplica-se um preenchimento ao redor da imagem, denominado *padding*. Um método comumente utilizado é o *zero-padding*, que consiste em fazer o preenchimento com zeros (O'SHEA; NASH, 2015). A Figura 11 ilustra o processo de convolução em uma imagem com *zero-padding*, evidenciando que a dimensionalidade do mapa de características resultante é igual à da imagem de entrada.

Figura 11 – Processo de convolução em uma imagem com *zero-padding*.



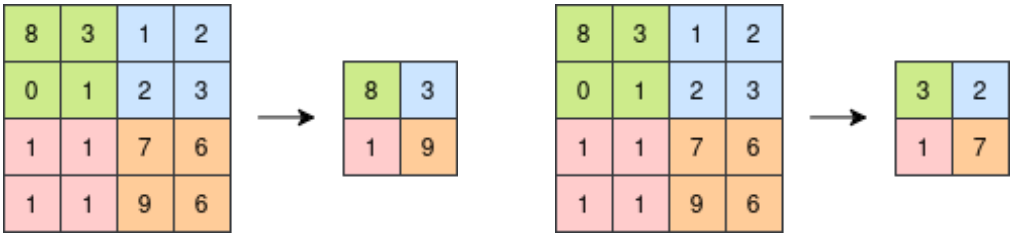
Fonte: O Autor (2025).

Adicionalmente, diferentes *kernels* podem ser aplicados de forma simultânea, possibilitando a extração de múltiplas características da imagem, como por exemplo, bordas, texturas e outros padrões (LI *et al.*, 2022). Em geral, os *kernels* utilizados possuem formato quadrado e dimensões reduzidas. Contudo, estudos recentes têm explorado variações no tamanho e no formato dos *kernels*. O trabalho de Aboukhair *et al.* (2025) investiga e discute as potenciais vantagens do uso de filtros com dimensões superiores a 3×3, comumente adotadas em arquiteturas tradicionais. Já o trabalho de Sun, Ozay e Okatani (2016) analisa o uso de *kernels* com formato hexagonal e as suas implicações no desempenho dos modelos.

3.1.2 Camada de *Pooling*

Após a geração do mapa de características, pode-se aplicar uma camada de subamostragem (*pooling*). Esse processo tem como objetivo reduzir a dimensionalidade do mapa de características, descartando informações menos relevantes e contribuindo para a capacidade de generalização do modelo. Entre os tipos mais comuns de *pooling* estão o *max-pooling* e o *average pooling*. Ambos dividem o mapa de características em regiões, sendo que cada região é representada por um único valor. No *max-pooling*, esse valor corresponde ao maior elemento da região (Figura 12a), enquanto no *average pooling* corresponde à média dos valores (Figura 12b) (LI *et al.*, 2022).

Figura 12 – Métodos *max-pooling* e *average pooling*.



(a) Método *max-pooling* (b) Método *average pooling*

Fonte: O Autor (2025).

3.1.3 Camada Totalmente Conectada

A camada totalmente conectada representa a etapa final de uma CNN, sendo responsável pela fase de classificação. Nessa camada, todos os neurônios estão conectados a todas as ativações provenientes da camada anterior. A entrada dessa etapa é um vetor unidimensional obtido a partir do achatamento dos mapas de características extraídos nas camadas anteriores. Esse processo converte os mapas, originalmente representados por matrizes multidimensionais, em um único vetor unidimensional que servirá como entrada para as camadas totalmente conectadas (Figura 13).

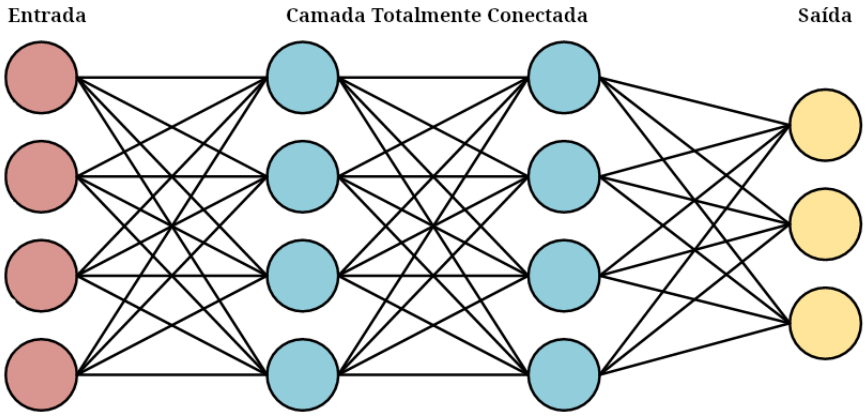
Figura 13 – Processo de achatamento de uma matriz.

Fonte: O Autor (2025).

A Figura 14 ilustra a estrutura típica de uma camada totalmente conectada, destacando a ligação entre neurônios. Durante o processo de classificação, cada neurônio calcula a soma dos produtos entre as entradas x e os pesos w, adicionando um termo de viés (*bias*) ao resultado. A Equação 3.1 demonstra essa operação, onde x_i corresponde a cada valor de entrada, w_i o respectivo peso e b o termo de viés. O resultado dessa operação é transformado por uma função de ativação não linear, que estabelece a saída final do neurônio (ALZUBAIDI *et al.*, 2021; PURWONO *et al.*, 2023).

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

Figura 14 – Camada totalmente conectada.



Fonte: Adaptado de Li *et al.* (2022)

3.1.4 Treinamento da CNN

O objetivo da etapa de treinamento de uma CNN é ajustar iterativamente os valores dos pesos (w), *bias* e *kernels*, de modo que as predições realizadas se aproximem dos valores esperados. O processo de treinamento ocorre ao longo de diversas iterações, denominadas épocas, nas quais os parâmetros da rede são atualizados de forma a otimizar o seu desempenho em relação aos dados de treinamento. Esse é um processo supervisionado, no qual as amostras devem estar rotuladas com suas respectivas classes, permitindo uma comparação entre as predições e os valores reais. O treinamento prossegue por um número máximo de épocas predefinido ou até que o erro calculado atinja um valor inferior a um determinado limiar (SZELISKI, 2022; GOODFELLOW; BENGIO; COURVILLE, 2016).

Para quantificar o erro entre a predição do modelo e a saída esperada é empregada uma função de perda. Em problemas de classificação multiclasse a função de entropia cruzada é geralmente aplicada (SZELISKI, 2022). A Equação 3.2 representa a função de entropia cruzada, em que $y_k^{(i)}$ corresponde o rótulo real da classe k para a amostra i, $\hat{y}_k^{(i)}$ representa a predição do modelo para essa classe, e K representa o número de classes.

$$L = -\sum_{k=1}^{K} y_k^{(i)} \log(\hat{y}_k^{(i)}) \tag{3.2}$$

O valor obtido pela função de perda é utilizado a atualização dos pesos da rede. Durante o treinamento de uma CNN, diversas amostras de entrada são processadas para o cálculo da função de perda antes da atualização dos pesos. O conjunto dessas amostras constitui um lote (*mini-batch*), e para a atualização dos parâmetros da rede a média das perdas do lote é utilizada (SZELISKI, 2022). Essa média é dada pela Equação 3.3, em que n corresponde ao número de amostras do lote.

$$L = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} y_k^{(i)} \log(\hat{y}_k^{(i)}) \tag{3.3}$$

Após o cálculo da função de perda, algoritmo de *backpropagation* é aplicado para calcular os gradientes dessa função em relação aos pesos da rede ($\frac{\partial L}{\partial w_i}$). Esse processo realiza a atualização dos pesos w, propagando os erros das camadas finais em direção às camadas iniciais da rede. A Equação 3.4 representa a atualização do peso w_i, em que α corresponde a taxa de aprendizagem. Esse parâmetro controla a magnitude das atualizações e é um dos principais hiperparâmetros que devem ser ajustados para o treinamento da CNN (SZELISKI, 2022).

$$w_i \leftarrow w_i - \alpha \frac{\partial L}{\partial w_i} \tag{3.4}$$

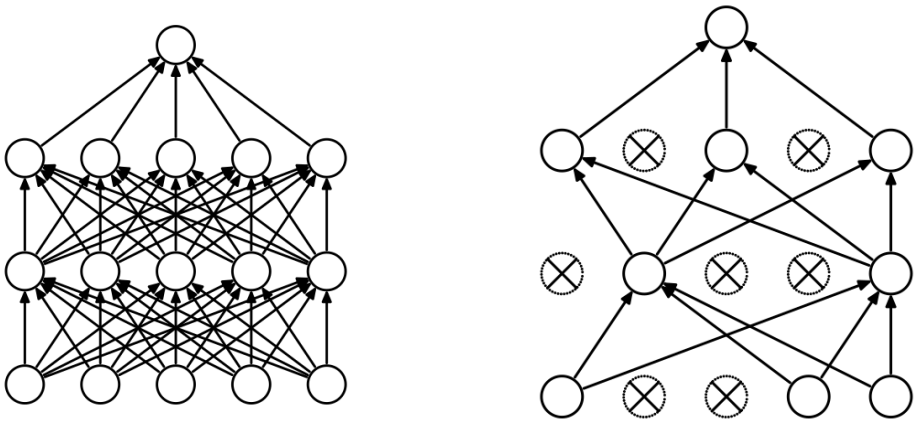
3.1.5 Técnicas de Regularização

As técnicas de regularização possuem como objetivo garantir que os modelos apresentem bom desempenho em dados inéditos, ou seja, dados distintos aos utilizados para o treinamento. Essa característica é denominada capacidade de generalização. Um modelo que apresenta bom desempenho apenas nos dados de treinamento, e desempenho insatisfatório em novos dados, se encontra em estado de *overfitting*. Já quando o modelo não apresenta bons resultados nos dados de treinamento, e também um baixo desempenho em dados novos, se encontra em um estado de *underfitting* (GOODFELLOW; BENGIO; COURVILLE, 2016).

As técnicas de regularização modificam o processo de aprendizado do modelo com o objetivo de reduzir o erro de generalização, procurando manter o desempenho do modelo nos dados de teste estável e evitando o problema de *overfitting*. Entre as principais estratégias de regularização destacam-se o *dropout*, a regularização de parâmetros L2, o *early stopping* e o aumento de dados (GOODFELLOW; BENGIO; COURVILLE, 2016).

A técnica de *dropout*, proposta por Srivastava *et al.* (2014), consiste em desativar aleatoriamente um ou mais neurônios a cada iteração de treinamento. Dessa forma, a cada época, determinados neurônios são temporariamente desligados e, em épocas posteriores, são novamente reativados (Figura 15). Essa abordagem impede que o modelo se torne excessivamente dependente de conexões específicas, promovendo maior robustez e redundância nas conexões neurais.

Figura 15 – Aplicação do *dropout* em uma rede neural.



(a) Rede neural padrão (b) Após aplicação do *dropout*

Fonte: Adaptado de Srivastava *et al.* (2014)

A regularização de parâmetros L2, introduzida por Krogh e Hertz (1991), consiste em adicionar um termo de penalização (λ) à função de perda (Equação 3.2), evitando que os pesos assumam valores excessivamente elevados. Essa técnica contribui para reduzir o *overfitting* e aprimorar a capacidade de generalização do modelo. A Equação 3.5 demonstra a função de perda L com a inclusão do parâmetro de regularização λ. O termo w_i representa o valor de cada peso.

$$L' = L + \frac{1}{2}\lambda \sum_i w_i^2 \tag{3.5}$$

O *early stopping* é técnica proposta por Prechelt (1998), que consiste em interromper o processo de treinamento antes que o modelo apresente sinais de *overfitting*. Para determinar o momento adequado da interrupção, um conjunto de validação é utilizado, sendo avaliado pelo modelo após cada iteração do processo de treinamento. O objetivo é comparar o desempenho obtido nos dados de treinamento com o desempenho nos dados de validação. Quando o erro nos dados de treinamento continua diminuindo, mas o erro nos dados de validação começa a aumentar, indica-se a ocorrência de *overfitting*, nesse caso o treinamento é interrompido. Além disso, uma outra abordagem ao *early stopping* consiste em definir um número máximo de épocas de treinamento, quando esse limite é alcançado o processo de treinamento é encerrado.

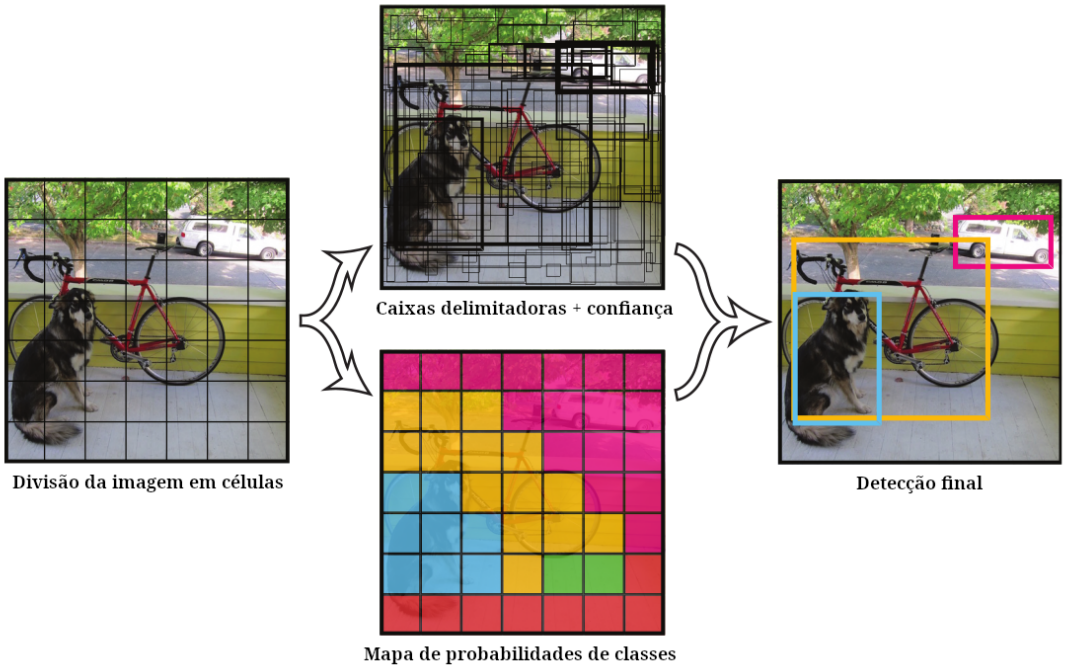
A técnica de aumento de dados (*data augmentation*) consiste em ampliar artificialmente a diversidade do conjunto de dados de treinamento por meio de transformações aplicadas aos dados originais. No caso de imagens, as transformações mais comuns incluem operações geométricas, como rotação, translação, espelhamento e escalonamento, e transformações fotométricas, como variação de brilho, contraste e saturação, conversão para escala de cinza e ajustes de cor (GOODFELLOW; BENGIO; COURVILLE, 2016).

3.1.6 *You Only Look Once* (YOLO)

O modelo YOLO é uma técnica de detecção de objetos em imagens que teve sua primeira versão desenvolvida por Redmon *et al.* (2016). Essa arquitetura apresentou uma nova forma de tratar o problema da detecção de objetos ao propor uma abordagem unificada, na qual a localização e a classificação dos objetos são realizadas em uma única etapa, diferentemente dos métodos anteriores que dependiam de múltiplas etapas de processamento (REDMON *et al.*, 2016). No contexto de detecção facial, o modelo pode ser configurado especificamente para identificar faces quando treinado em bases de dados especializadas, como realizado nos trabalhos de Won *et al.* (2019), Angus *et al.* (2022), Kim *et al.* (2022) e More *et al.* (2024).

O processamento das imagens pela YOLOv8 inicia-se com o redimensionamento imagem de entrada para uma dimensão fixa de 512×512 *pixels*. Em seguida, a imagem é dividida em uma grade de $S \times S$ células, em que cada célula é responsável por prever a presença e a localização dos objetos cuja região central se encontra dentro de seus limites. Cada célula prevê B caixas delimitadoras e níveis de confiança associados a essas caixas, os quais indicam o grau de certeza do modelo de que a região contém um objeto da classe C, bem como a precisão da caixa em delimitar esse objeto (VARGHESE; M., 2024). A Figura 16 demonstra a divisão da imagem de entrada em uma grade de $S \times S$ células, destacando as caixas delimitadoras e as probabilidades de classe estimadas para cada célula. A integração das caixas delimitadoras com as probabilidades de classe permite a detecção final dos objetos.

Figura 16 – Etapas de processamento de uma imagem pela YOLO.

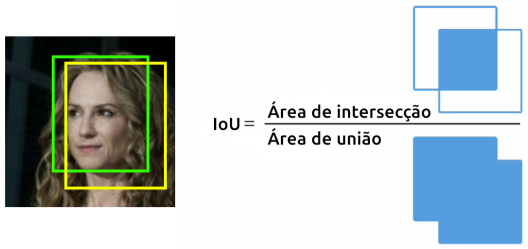


Fonte: Adaptado de Redmon *et al.* (2016)

A Equação 3.6 apresenta o cálculo do nível de confiança Pr, de um objeto O pertencer à classe C. A métrica $IoU(predicao, verdade)$ corresponde à intersecção sobre a união (IoU - *Intersection over Union*) entre a caixa prevista e a real (SOHAN; RAM; REDDY, 2024). Essa métrica é utilizada para quantificar a proximidade entre a caixa delimitadora real, presente nas imagens de treinamento, e a caixa prevista pelo modelo. A Figura 17 ilustra o processo de cálculo da *IoU*, que é obtida pela razão entre a área de intersecção e a área de união das duas regiões.

$$Pr(O, C) \times IoU(predicao, verdade) \tag{3.6}$$

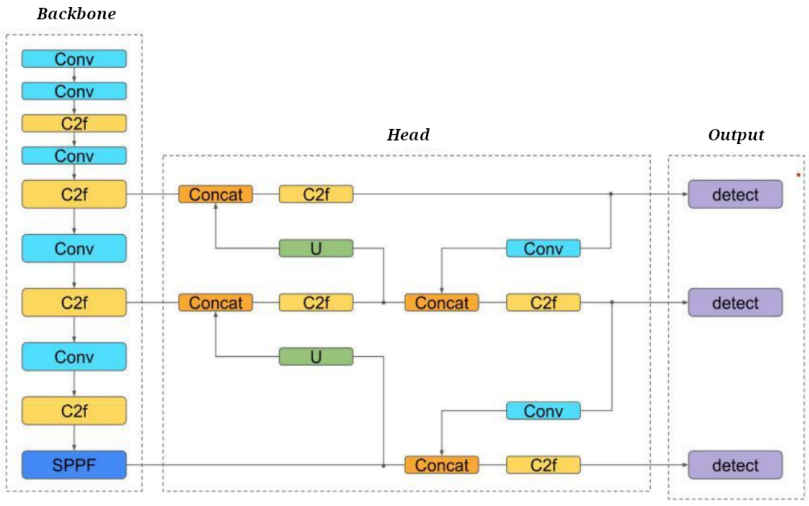
Figura 17 – Cálculo da IoU. A caixa verde representa a caixa real e a caixa amarela representa a caixa prevista pela CNN.



Fonte: O Autor (2025).

A arquitetura do modelo YOLOv8 é apresentada na Figura 18, a qual é composta por três componentes: *backbone*, *head* e *output*. O *backbone* é responsável por extrair as características das imagens em diferentes escalas. O *head* realiza a combinação e refinamento das características. Por fim, o *output* é responsável por gerar as caixas delimitadoras e os níveis de confiança para cada classe (HIDAYATULLAH *et al.*, 2025; SOHAN; RAM; REDDY, 2024).

Figura 18 – Arquitetura da YOLOv8.

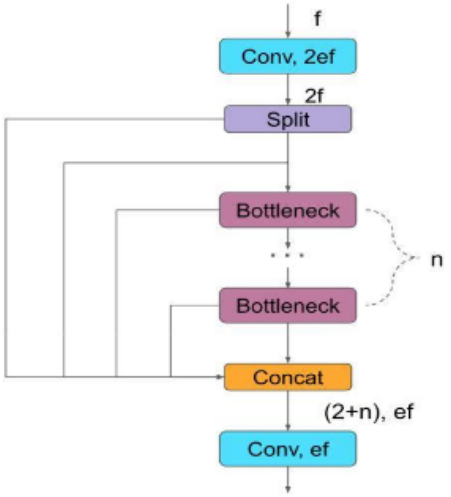


Fonte: Adaptado de Sohan, Ram e Reddy (2024)

O *backbone* é baseado em uma rede com arquitetura CSPDarknet53 (BOCHKOVSKIY; WANG; LIAO, 2020). Esse é composto principalmente por módulos convolucionais (*Conv*), módulos C2f (*Cross-Stage Feature Fusion*) e SPPF (*Spatial Pyramid Pooling Fast*) (HIDAYATULLAH *et al.*, 2025). O módulo *Conv* consiste em camadas convolucionais com *kernels* de tamanho 3×3 e *stride* igual a 2. O uso de *stride* de tamanho 2 faz com que os mapas de características resultantes apresentem metade da dimensão dos dados de entrada, o que contribui para reduzir o custo computacional e para a capacidade de generalização do modelo (HIDAYATULLAH *et al.*, 2025).

O módulo C2f (Figura 19) atua na extração de características através do uso de convoluções e blocos do tipo *Bottleneck*. Nesse módulo, o fluxo de dados é dividido, sendo uma das partes encaminhada para o processamento pelos blocos *Bottleneck* e o restante encaminhado para a saída. Ao final, ambas as partes são concatenadas e submetidas a uma convolução final, consolidando as características extraídas (HIDAYATULLAH *et al.*, 2025; SOHAN; RAM; REDDY, 2024).

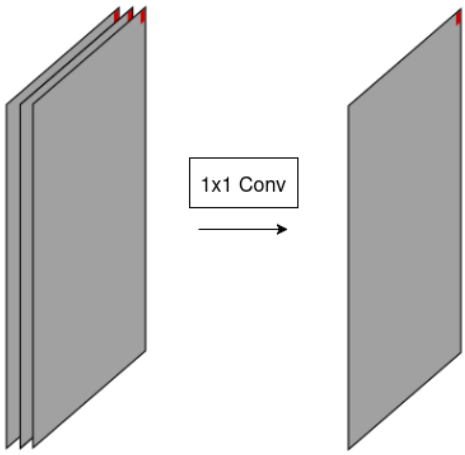
Figura 19 – Arquitetura do módulo C2f.



Fonte: Adaptado de Sohan, Ram e Reddy (2024)

Os blocos *Bottleneck* extraem os mapas de características através de convoluções convencionais, em seguida, aplicam operações de convoluções 1×1. Uma convolução 1×1 opera sobre cada *pixel* individualmente, realizando combinações lineares entre os canais e permitindo ajustes na profundidade do mapa de características (Figura 20). Essa operação preserva as dimensões de largura e altura, mas permite alterar o número de canais (profundidade), o que não apenas pode reduzir o custo computacional, como também possibilita gerar uma representação mais consistente para as camadas subsequentes (HIDAYATULLAH *et al.*, 2025).

Figura 20 – Aplicação de uma convolução 1×1.



Fonte: Adaptado de Younesi *et al.* (2024)

O módulo SPPF é uma variação otimizada da técnica SPP, introduzida por He *et al.* (2014). Nesse módulo, múltiplas operações de *max-pooling* com filtros de tamanho 5×5 são aplicadas de maneira sequencial sobre um mesmo mapa de características. Ao final do processo, todos os resultados são concatenados, produzindo uma combinação de características em diferentes escalas (HIDAYATULLAH *et al.*, 2025; SOHAN; RAM; REDDY, 2024).

O fluxo de dados segue do *backbone* para o *head*, o qual utiliza módulos de concatenação (*Concat*) para combinar as características previamente extraídas com versões de maior resolução espacial das mesmas, geradas por módulos de *upsample* (U). O aumento de resolução é realizado através do algoritmo do vizinho mais próximo (*nearest neighbor*), com o objetivo de aprimorar a detecção de objetos em diferentes escalas. Nesse estágio, também são aplicados módulos C2f e camadas convolucionais (*Conv*), que refinam e integram as informações extraídas (HIDAYATULLAH *et al.*, 2025).

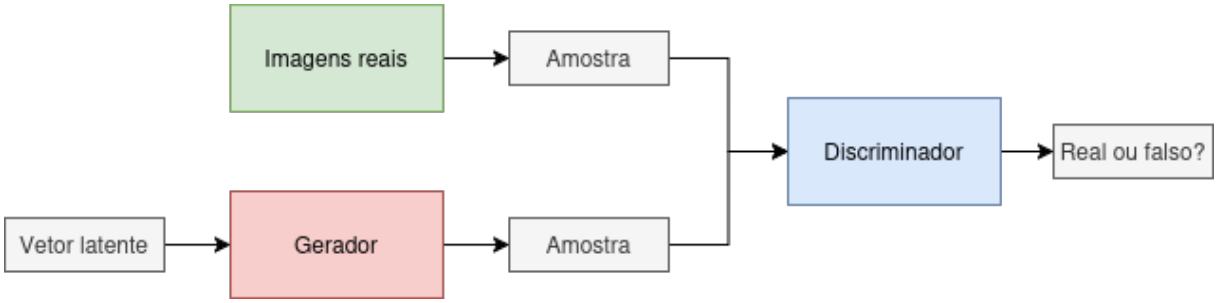
A etapa final da detecção ocorre no bloco de saída (*output*), composto por três módulos *detect*, responsáveis por gerar as caixas delimitadoras e os níveis de confiança para cada classe. Cada módulo *detect* é otimizado para identificar objetos dentro de uma faixa particular de escalas, o que garante a efetividade da detecção multiescala (HIDAYATULLAH *et al.*, 2025).

3.2 REDES GENERATIVAS ADVERSARIAIS

As redes generativas adversariais (GANs - *Generative Adversarial Network*s) foram introduzidas por Goodfellow *et al.* (2014). A ideia consiste em um modelo composto por duas redes neurais: a geradora e a discriminadora, que são treinadas simultaneamente em um processo de competição.

O processo de treinamento ocorre de forma iterativa (Figura 21). Inicialmente, um vetor de números aleatórios (vetor latente) é gerado e utilizado como ponto de partida. As camadas da rede geradora transformam esse vetor gradualmente, criando formas, texturas e detalhes. Ao final, a rede geradora sintetiza uma imagem que segue padrões adquiridos durante o treinamento. Em seguida, a rede discriminadora avalia a imagem sintetizada, indicando se ela se assemelha às amostras utilizadas no treinamento. Após, os pesos são atualizados de modo a minimizar as perdas de cada rede: a rede geradora ajusta seus parâmetros para produzir imagens cada vez mais semelhantes às amostras reais, enquanto a rede discriminadora aprimora sua capacidade de distinguir entre dados reais e sintetizados. Esse processo é encerrado após um número pré-definido de épocas ou quando a taxa de acerto da rede discriminadora se estabiliza em torno de 50% (GOODFELLOW *et al.*, 2014).

Figura 21 – Processo de treinamento de uma rede GAN.

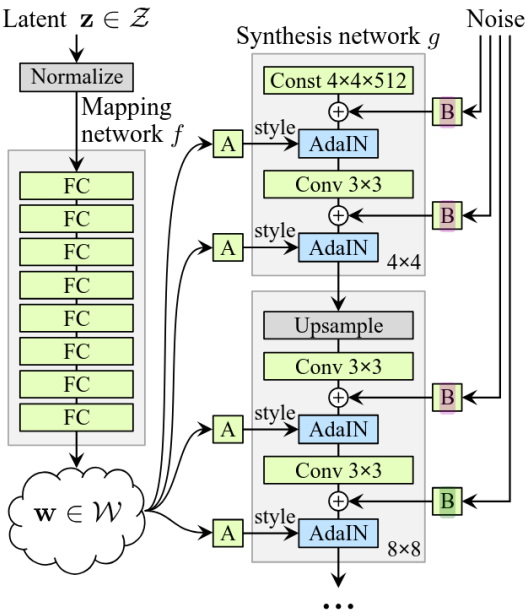


Fonte: Adaptado de Creswell *et al.* (2018)

3.2.1 Modelo *StyleGAN*

O StyleGAN é um modelo do tipo GAN, proposto por Karras, Laine e Aila (2021), com foco na síntese de faces humanas. O modelo é baseado na *Progressive GAN* de Karras *et al.* (2018b), incorporando melhorias na arquitetura da rede geradora (KARRAS; LAINE; AILA, 2021). A Figura 22 ilustra a arquitetura da rede geradora, composta por dois componentes principais: a rede de mapeamento (*mapping network*) e a rede de síntese (*synthesis network*). A rede de mapeamento é a principal diferença do modelo, sendo responsável por transformar o vetor latente em um espaço latente ajustado (espaço intermediário), reduzindo correlações entre os elementos do vetor e produzindo uma representação mais adequada para a síntese das imagens. Já a rede de síntese utiliza essas informações para gerar a imagem sintética (KARRAS; LAINE; AILA, 2021).

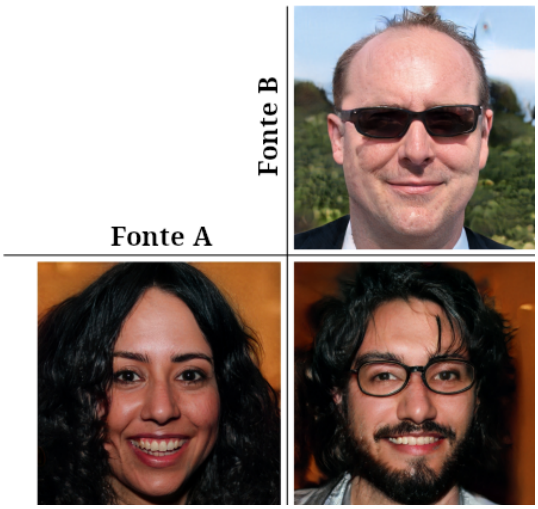
Figura 22 – Arquitetura da *StyleGAN*.



Fonte: Adaptado de Karras, Laine e Aila (2021)

A rede de mapeamento recebe como entrada um vetor latente (z), de dimensão 512, composto por valores numéricos aleatórios. Esse vetor é então transformado em um vetor intermediário (w) de mesmas dimensões. A transformação é realizada por uma rede neural composta por oito camadas totalmente conectadas (FC na Figura 22), cujo objetivo é projetar o vetor z em um espaço menos correlacionado. O vetor w define os estilos derivados do vetor latente z, sendo esses os valores que irão definir as principais características da face gerada. No trabalho de Karras, Laine e Aila (2021), diferentes vetores w foram combinados, permitindo mesclar características faciais entre as duas imagens geradas originalmente (Figura 23).

Figura 23 – Face gerada a partir da combinação de dois vetores w.



Fonte: Adaptado de Karras, Laine e Aila (2021)

A rede de síntese é composta por uma sequência de módulos (g), organizados de forma hierárquica (Figura 22). Cada módulo g inclui: vetores constantes (*Const* $4 \times 4 \times 512$), exclusivo para o primeiro módulo; blocos de Adaptação Normativa de Instância (AdaIN); camadas convolucionais (*Conv* 3×3); e uma combinação de vetores de ruído (*noise*) e fatores de escala (B). O fluxo de dados de um módulo para o outro é feitor por um bloco de *Upsample*.

A configuração de cada módulo g é similar, porém o primeiro módulo usa como dados iniciais o bloco *Const* $4 \times 4 \times 512$. Já os módulos restantes da série utilizam os dados do módulo anterior modificados pelo bloco *Upsample*. O bloco *Const* $4 \times 4 \times 512$ contém mapas de características gerados pela rede durante o processo de treinamento. Esse tem como objetivo inicializar o processo de geração da imagem. O módulo *Upsample* é responsável por duplicar a resolução espacial por meio de interpolação bilinear (KARRAS; LAINE; AILA, 2021).

O processamento dos dados iniciais do módulo começa por uma camada convolucional de *kernels* de tamanho 3×3 e *stride* igual a 1, essa operação é representada pelo bloco *Conv* 3×3. Dessa maneira, a resolução espacial é preservada enquanto a rede extrai e refina as características locais presentes nos dados.

Em seguida, os mapas gerados são incrementados com valores de ruído e direcionados ao bloco *AdaIN*. A operação AdaIN foi introduzida por Huang e Belongie (2017) e tem como objetivo combinar o conteúdo de uma imagem com os estilos (cores, efeitos visuais) de outra imagem. Mais especificamente, no modelo StyleGAN, a operação *AdaIN* é utilizada para combinar os estilos extraídos do vetor w à imagem que está sendo gerada pela rede.

Os estilos são representados por um vetor y, organizado em pares (y_s, y_b), em que y_s controla a intensidade das ativações (escala) e y_b ajusta sua média (deslocamento). O vetor y é gerado pelo *bloco A*, o qual aplica em w uma transformação, que é definida pela Equação 3.7. Nessa equação, A é uma matriz estimada durante o treinamento, enquanto b é o termo de viés igualmente estimado pela rede.

$$y = Aw + b \tag{3.7}$$

A operação *AdaIN* é apresentada na Equação 3.8, onde x_i representa cada mapa de características, y o vetor de estilos, $\mu(x_i)$ corresponde à média de x_i e $\sigma(x_i)$ o desvio padrão.

$$\text{AdaIN}(x_i, y) = y_{s,i} \frac{x_i - \mu(x_i)}{\sigma(x_i)} + y_{b,i} \tag{3.8}$$

A quantidade de módulos g, presentes na rede de síntese, segue o princípio do crescimento progressivo da resolução da imagem sintetizada: a cada módulo, a dimensão espacial da imagem é duplicada, iniciando em 4×4. Dessa forma, o processo de geração ocorre de forma gradual, com a resolução sendo aumentada progressivamente até atingir o tamanho final desejado (KARRAS; LAINE; AILA, 2021).

A geração da imagem final segue o procedimento do *Progressive GAN* (KARRAS *et al.*, 2018b), onde os mapas de características gerados pela rede de síntese passam por uma convolução 1×1 que transforma esses valores em *pixels* no espaço de cores RGB (*Red Green Blue*) (KARRAS; LAINE; AILA, 2021; KARRAS *et al.*, 2018b).

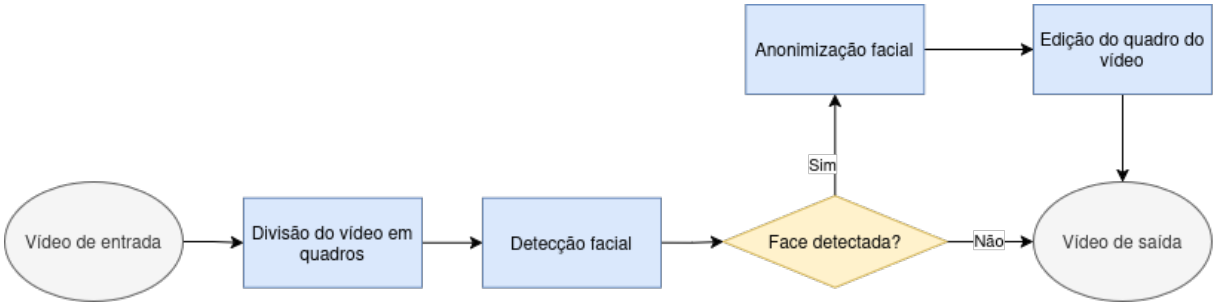
4 IMPLEMENTAÇÃO DESENVOLVIDA

O processo de detecção das faces, etapa necessária para a anonimização, foi realizado utilizando o modelo YOLOv8, disponibilizado pela Ultralytics[1]. Foram adotados os modelos pré-treinados fornecidos pela Lindevs[2], os quais correspondem as variações do modelo YOLOv8 treinadas para detecção facial utilizando a base de dados WIDER FACE (YANG *et al.*, 2016). Para a anonimização facial, o sistema implementa duas abordagens: ofuscação e síntese. O processo de desenvolvimento foi realizado utilizando a plataforma de computação em nuvem RunPod[3], a qual disponibiliza um ambiente para desenvolvimento com a linguagem Python[4], com suporte a GPUs da fabricante Nvidia. Para o processamento dos fluxos de vídeo foi utilizado o *framework* OpenCV[5].

4.1 ARQUITETURA DO SISTEMA

A Figura 24 apresenta o fluxo de processamento de vídeo adotado. O processo inicia com a segmentação do vídeo em quadros por meio do *framework* OpenCV, possibilitando que cada quadro seja processado de forma sequencial e independente. Em seguida, o algoritmo de detecção facial identifica e extrai as regiões de interesse correspondentes às faces presentes no quadro. Essas regiões são utilizadas como entrada para o processo de anonimização facial. Essa arquitetura promove um desacoplamento entre as etapas de processamento de vídeo, detecção de faces e o método de anonimização empregado. Essa característica modular possibilita, por exemplo, a substituição da técnica de anonimização sem a necessidade de alterações nos demais componentes do sistema.

Figura 24 – Fluxo de processamento do sistema de anonimização facial.



Fonte: O Autor (2026).

[1] https://docs.ultralytics.com/pt/models/yolov8/
[2] https://github.com/lindevs/yolov8-face
[3] https://www.runpod.io/
[4] https://www.python.org/
[5] https://opencv.org/

4.2 DETECÇÃO FACIAL

O Anexo B apresenta a implementação da etapa de detecção facial. Essa utiliza um modelo da arquitetura YOLOv8, o qual é disponibilizado pela Ultralytics em diferentes versões que variam conforme a quantidade de parâmetros. A Tabela 1 apresenta uma comparação entre as variantes disponíveis. Considerando os requisitos de processamento em tempo real do sistema, adotou-se o modelo YOLOv8n (*nano*), por ser uma versão mais leve e rápida, o que reduz a latência de detecção e favorece o processamento em tempo real.

Tabela 1 – Modelos YOLOv8, número de parâmetros e velocidade.

Modelo	**Parâmetros (M)**	**Velocidade GPU (ms)**
YOLOv8n	3,2	0,99
YOLOv8s	11,2	1,20
YOLOv8m	25,9	1,83
YOLOv8l	43,7	2,39
YOLOv8x	68,2	3,53

Fonte: Jocher, Chaurasia e Qiu (2023).

Além disso, os resultados de desempenho reportados pela Lindevs na avaliação dos modelos pré-treinados, utilizando a base de dados WIDER FACE, reforçam essa decisão. Conforme apresentado na Tabela 2, a diferença de acurácia entre a versão *nano* e o modelo com maior número de parâmetros é de aproximadamente 6%. No contexto desta implementação, esse ganho de precisão não compensa o aumento do tempo de detecção observado nos modelos de maior porte.

Tabela 2 – Modelos YOLOv8 pré-treinados e precisão.

Modelo	**Precisão (%)**
YOLOv8n-Face	79,38
YOLOv8s-Face	82,90
YOLOv8m-Face	84,55
YOLOv8l-Face	85,43
YOLOv8x-Face	85,80

Fonte: Lindevs (2023).

4.3 ANONIMIZAÇÃO FACIAL POR OFUSCAÇÃO

Para a anonimização facial por ofuscação, adotou-se o método de desfoque, utilizado nos trabalhos de Angus *et al.* (2022) e Mrityunjay e Narayanan (2011). O procedimento consiste na aplicação de um filtro gaussiano, onde a máscara de convolução (*kernel*) atua diretamente sobre a área de interesse, de modo que os *pixels* da região sejam suavizados, ocasionando a perda da definição das características faciais. O Anexo C apresenta a implementação do desfoque sobre as faces detectadas.

O filtro gaussiano é definido pela Equação 4.1, na qual x e y representam as distâncias horizontal e vertical do *pixel* em relação ao centro do *kernel*, respectivamente, enquanto σ corresponde ao desvio padrão da distribuição de pesos espaciais, o qual determina a intensidade do desfoque.

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4.1}$$

Neste trabalho, adotou-se uma técnica de dimensionamento dinâmico para o *kernel*, na qual o tamanho é calculado de forma adaptativa em função da largura da face detectada. Essa estratégia de anonimização baseia-se no estudo desenvolvido por Shvai, Carmona e Nakib (2023), o qual aponta que, ao ajustar o tamanho do *kernel* à escala do objeto, garante-se uma anonimização eficaz independente de sua dimensão.

4.4 ANONIMIZAÇÃO FACIAL POR SÍNTESE

Para a anonimização facial por síntese, adotou-se uma abordagem baseada na integração dos modelos StyleGAN e SimSwap, conforme proposto por Tran *et al.* (2024). O fluxo de processamento inicia-se com a geração de uma imagem sintética por meio da rede StyleGAN, a qual fornece a identidade sintética utilizada na anonimização. Na sequência, utiliza-se o modelo SimSwap para realizar a transferência de identidade entre a face sintética e a face detectada.

O SimSwap, proposto por Chen *et al.* (2020), baseia-se em redes neurais convolucionais para a realização de troca de identidade entre faces (*face swap*). Esse modelo realiza a extração das características da face de origem e as transfere para a face alvo, preservando atributos como expressões faciais, condições de iluminação e pose, de modo a manter a coerência visual da imagem resultante. A implementação da geração de faces sintéticas e da troca de faces encontra-se disponível nos Anexos D e E, respectivamente.

O fluxo de processamento do vídeo opera de forma sequencial, realizando a detecção facial em um quadro e, posteriormente, aplicando a técnica de anonimização com a face sintética previamente gerada. Para que o resultado visual seja coerente, é necessário manter a mesma identidade sintética associada ao mesmo indivíduo ao longo de toda a sequência de quadros. Em cenários com múltiplos indivíduos, essa consistência exigiria um mapeamento entre as faces detectadas e as identidades geradas (TRAN *et al.*, 2024). No entanto, na arquitetura proposta neste trabalho, os quadros são processados de forma independente, sem a utilização de memória temporal ou mecanismos de rastreamento de identidade. Como consequência, a implementação realizada limita-se à anonimização de uma única identidade por vídeo, aplicando a mesma face sintética em todos os quadros.

5 TESTES E RESULTADOS OBTIDOS

Este capítulo apresenta os testes realizados e os resultados obtidos na avaliação dos métodos de anonimização facial por ofuscação e síntese. A partir desses testes, são analisadas tanto a capacidade de anonimização quanto o desempenho computacional das abordagens. Inicialmente, é descrito o conjunto de dados utilizado nos testes, bem como o critério de seleção das amostras. Em seguida, é realizada uma avaliação visual dos resultados de anonimização, seguida de uma análise quantitativa da capacidade de anonimização dos métodos avaliados. Por fim, é avaliado o desempenho computacional das implementações desenvolvidas, considerando os requisitos de tempo real.

5.1 CONJUNTO DE DADOS PARA TESTE

Para realização dos testes foi utilizado o conjunto de dados CelebV-HQ desenvolvido por Zhu *et al.* (2022), e também empregado nos trabalhos de Tran *et al.* (2024) e Lei *et al.* (2024). O CelebV-HQ é composto por vídeos com duração entre 3 e 20 segundos, contendo faces de diversos indivíduos em diferentes poses e expressões faciais.

A base de dados CelebV-HQ caracteriza-se pela representação de cenários reais, contendo 35.666 clipes de vídeo de 15.653 identidades distintas, anotados com 83 atributos faciais distribuídos em quatro categorias principais: aparência, que engloba traços físicos, acessórios (como óculos e máscaras) e estilos de cabelo; gênero; ação, que abrange atividades como falar, rir, bocejar ou realizar movimentos de cabeça; e emoção, compreendendo estados neutros, alegria, raiva, surpresa e tristeza. Além disso, o conjunto caracteriza-se por apresentar uma ampla variação natural de escala e enquadramento das faces.

Neste trabalho, optou-se por utilizar um subconjunto de dados que mantivesse a mesma representatividade de atributos da base de dados original, porém em um volume reduzido de amostras. Devido a alta densidade de atributos por vídeo, foi possível representar os 83 atributos faciais da base de dados original em 60 amostras, as quais foram escolhidas de forma manual e utilizadas para os testes dos modelos implementados. Nessas amostras, 30 são do sexo masculino e 30 são do sexo feminino, cobrindo as 40 categorias de aparência, 35 de ação e 8 de emoção do CelebV-HQ. Além disso, possuem variação de tipos de cabelo e traços faciais, além de acessórios como óculos, batom, brincos, maquiagem, chapéu e gravata. O Anexo A apresenta os identificadores dos vídeos selecionados.

5.2 AVALIAÇÃO VISUAL DA ANONIMIZAÇÃO

Para a avaliação visual, foi adotada uma estratégia de amostragem temporal, extraindo-se um quadro por segundo dos vídeos de teste, de modo a contemplar diferentes expressões, poses e condições de iluminação. A Figura 25 apresenta uma comparação entre o quadro original e os resultados obtidos pelos métodos de anonimização por ofuscação e síntese.

Figura 25 – Resultado das faces anonimizadas.

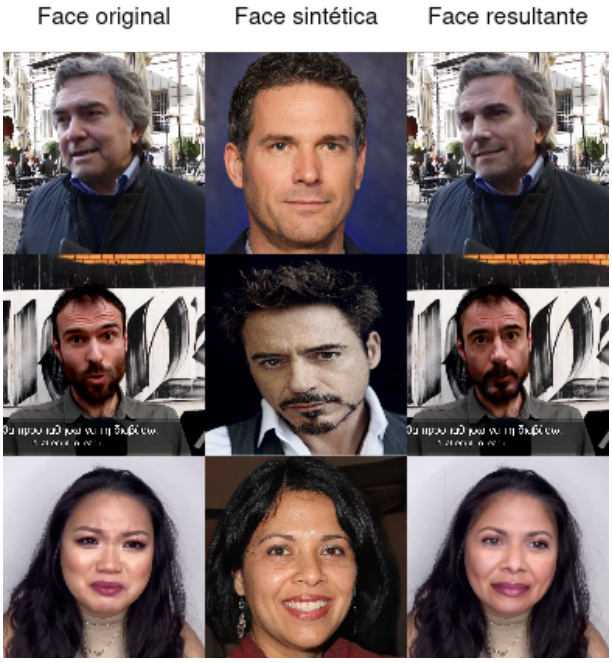
(a) Método de ofuscação

(b) Método de síntese

Fonte: O Autor (2026).

O método de ofuscação apresentou resultados visuais satisfatórios devido à sua natureza simples e determinística. Em contrapartida, a abordagem por síntese demonstrou dificuldade em reproduzir movimentos faciais finos e variações sutis ao longo dos quadros do vídeo, além de apresentar um aspecto artificial às faces anonimizadas. Observou-se, ainda, que a qualidade visual do método por síntese foi superior quando a face original e a face sintética aplicada compartilhavam semelhanças, os cenários nos quais as faces divergiam significativamente resultaram em faces menos harmoniosas. A Figura 26 evidencia esses cenários. Na Figura 26a, observa-se a semelhança de atributos faciais entre as faces original e sintética, resultando em uma face mais realista. Em contrapartida, a Figura 26b evidencia os impactos da disparidade de características, como a diferença de cabelo e o uso de óculos, o que acarreta em artefatos visuais e aspecto artificial.

Figura 26 – Combinação da face original e face sintética.



(a) Semelhança entre face original e sintética

(b) Disparidade entre face original e sintética

Fonte: O Autor (2026).

Além disso, a abordagem por síntese facial mostrou-se mais suscetível à geração de artefatos visuais e falhas em cenários específicos, principalmente na presença de acessórios como óculos. Também, como a identidade sintética é gerada aleatoriamente para cada vídeo, casos em que suas características faciais diferiam significativamente da face original resultaram em degradação da qualidade visual. Em contrapartida, o método de ofuscação apresentou maior consistência visual, uma vez que se limita à aplicação de um filtro matemático sobre a face detectada. A Figura 27 apresenta um compilado dos principais artefatos visuais observados.

Figura 27 – Artefatos na anonimização por síntese.

Fonte: O Autor (2026).

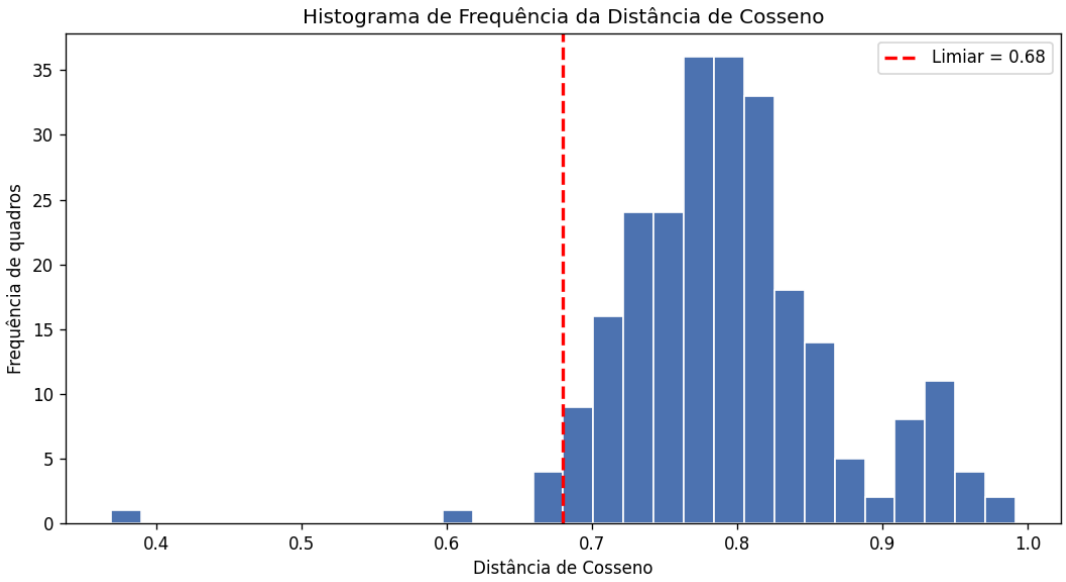
5.3 AVALIAÇÃO QUANTITATIVA DA ANONIMIZAÇÃO

A eficácia de um sistema de anonimização facial pode ser avaliada por meio de modelos de reconhecimento facial, verificando se uma face anonimizada ainda pode ser associada à identidade original (ANDRADE, 2023). Neste trabalho, utilizou-se o modelo ArcFace (DENG *et al.*, 2022), que realiza a comparação entre duas faces ao transformá-las em representações vetoriais de alta dimensão (*embeddings*) e calcular a similaridade geométrica entre esses vetores por meio da Distância de Cosseno (SERENGIL; OZPINAR, 2020).

Considerou-se que o processo de anonimização falhou quando a Distância de Cosseno entre duas faces foi inferior ao limiar estabelecido de $0,68$. Assim, valores inferiores a esse limite configuram falhas de anonimização, enquanto valores iguais ou superiores indicam sucesso. Esse limiar é citado por Serengil e Ozpinar (2020) como um valor de referência para aplicações generalistas de reconhecimento facial. As amostras anonimizadas por ofuscação não foram consideradas nas métricas de similaridade, uma vez que a degradação da região facial impossibilita a detecção das faces pelo ArcFace.

Nos testes de anonimização por síntese facial, foram excluídos os clipes com artefatos visuais ou deformações severas. Dessa forma, o conjunto de dados final concentrou-se nos cenários em que a anonimização foi realizada com sucesso, totalizando 249 quadros. Os resultados obtidos indicaram uma Distância de Cosseno média de $0,7946$, com desvio padrão de $0,0734$. Adicionalmente, a Taxa de Reidentificação foi de $2,42\%$, indicando que apenas uma pequena parcela das amostras apresentou Distância de Cosseno inferior ao limiar adotado e, portanto, pôde ser associada à identidade original. A Figura 28 apresenta o histograma de frequência dos resultados, no qual se observa que a maioria das amostras está acima do limiar adotado, indicando o sucesso da anonimização.

Figura 28 – Distribuição das distâncias de cosseno.



Fonte: O Autor (2026).

5.4 ANÁLISE DO TEMPO DE ANONIMIZAÇÃO

Os testes foram realizados na plataforma de computação em nuvem RunPod, utilizando uma máquina virtual equipada com 6 núcleos virtuais de um processador AMD EPYC 7443 (frequência base de 2,8 GHz e frequência máxima de 4,0 GHz), 32 GB de memória RAM e uma GPU Nvidia RTX 2000 Ada Generation com 16 GB de memória de vídeo. Ambos os métodos foram executados duas vezes sobre os 60 vídeos do conjunto de testes, totalizando aproximadamente 466 segundos de vídeo e mais de 25.000 quadros. Os tempos de detecção facial e de anonimização foram registrados para cada quadro, sendo calculadas a média (μ), o desvio padrão (σ) e a taxa média de QPS (Quadros por Segundo). A Tabela 3 apresenta os resultados obtidos.

Tabela 3 – Desempenho computacional dos métodos de anonimização.

Método de anonimização	**Etapa de Processamento**	**Tempo** ($\mu \pm \sigma$) (ms)	**QPS Médio**
Ofuscação	Detecção facial	7,19 ± 1,03	14,48
	Aplicação do desfoque	61,77 ± 29,58	
	Tempo Total por Quadro	**69,03 ± 29,82**	
Síntese	Detecção facial	8,08 ± 1,28	1,15
	Aplicação da face sintética	859,63 ± 202,17	
	Tempo Total por Quadro	**867,80 ± 202,34**	

Fonte: O Autor (2026).

Os resultados mostram que a etapa de detecção facial apresentou um baixo custo computacional. Em contrapartida, a etapa de anonimização evidenciou uma diferença significativa entre as abordagens. Enquanto o método de ofuscação consumiu, em média, $61,77$ ms por quadro, a síntese facial demandou aproximadamente 14 vezes mais tempo de processamento. Esse comportamento pode ser atribuído à complexidade do modelo *SimSwap*, que, para cada quadro, realiza a extração das características de identidade das faces de origem e de destino, seguida da síntese e reconstrução da face resultante."

Em termos de processamento em tempo real, os resultados indicam que apenas o método de ofuscação atende aos requisitos da aplicação. A abordagem alcançou uma taxa média de 14,48 QPS, superando o limite mínimo de 8 QPS recomendado por Keval e Sasse (2008). Em contrapartida, a síntese facial registrou apenas 1,15 QPS, inviabilizando sua utilização em cenários de processamento em tempo real e restringindo seu emprego a aplicações de pós-processamento.

6 CONSIDERAÇÕES FINAIS

Este trabalho teve como objetivo principal a avaliação comparativa de métodos de anonimização facial voltados para a operação em tempo real, analisando e implementando duas abordagens distintas: a ofuscação e a síntese facial. As soluções foram desenvolvidas na linguagem Python, utilizando o *framework* OpenCV para o processamento de imagens e vídeos, e a plataforma em nuvem RunPod como ambiente de desenvolvimento e execução de testes.

Para a avaliação dos métodos desenvolvidos, foi utilizada a base de dados CelebV-HQ. Essa escolha foi motivada pela necessidade de avaliar os algoritmos em cenários reais, que apresentam faces sob diferentes condições de iluminação, poses e enquadramentos. Devido à elevada dimensionalidade da base original, foi selecionado um subconjunto de 57 clipes de vídeo para a realização dos testes, preservando a representatividade dos atributos faciais presentes no conjunto completo.

Para a etapa de detecção facial, optou-se pelo uso do modelo de detecção YOLO, disponibilizado pela Ultralytics. Especificamente, utilizou-se a arquitetura YOLOv8n em sua versão pré-treinada pela Lindevs para detecção facial, a escolha dessa variante foi motivada pela necessidade de reduzir a latência do sistema e viabilizar o processamento em tempo real. Os resultados obtidos demonstraram o elevado desempenho do método, com tempo médio de detecção de 7,19 ms por quadro, atendendo satisfatoriamente aos requisitos de execução em tempo real.

Quanto à anonimização facial por ofuscação, foi adotada uma técnica de desfoque baseada na aplicação de um filtro gaussiano sobre as faces detectadas. Essa abordagem apresentou alta consistência visual, mantendo-se estável mesmo diante de variações de iluminação e pose. Além disso, o baixo custo computacional do algoritmo contribuiu para a eficiência em desempenho, registrando um tempo médio de processamento de 61,77 milissegundos por quadro e uma taxa média de 14,48 QPS, cumprindo com sucesso os requisitos de tempo real.

O método de anonimização facial por síntese utilizado consiste na combinação do modelo *StyleGAN*, responsável pela geração de faces sintéticas, com o modelo *SimSwap*, empregado na transferência de identidade para a face detectada. A abordagem apresentou a presença de artefatos visuais e aparência artificial, além de dificuldades na reprodução de variações sutis de expressão facial ao longo do vídeo. Adicionalmente, o desempenho computacional não se mostrou adequado para execução em tempo real, atingindo um tempo médio de processamento de 859,63 milissegundos por quadro e uma taxa média de 1,15 QPS.

6.1 TRABALHOS FUTUROS

Como sugestão para trabalhos futuros, propõe-se:

- Implementar um módulo de rastreamento temporal de identidades para viabilizar o suporte a múltiplos indivíduos por vídeo no método de anonimização por síntese.
- Estudar a aplicação de modelos alternativos ao *SimSwap* para a tarefa de troca de identidades, com o objetivo de otimizar o desempenho e melhorar a qualidade visual.
- Expandir a anonimização por síntese para incluir atributos faciais secundários, tais como cabelo e estilos de penteado, conforme apresentado no trabalho de Tran *et al.* (2024).

REFERÊNCIAS

ABOUKHAIR, M. *et al.* CNN filter sizes, effects, limitations, and challenges: An exploratory study. **Network**, Informa UK Limited, p. 1–29, jul. 2025.

ALZUBAIDI, L. *et al.* Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. **J. Big Data**, Springer Science and Business Media LLC, v. 8, n. 1, p. 53, mar. 2021.

ANDRADE, R. G. de. **Privacy-Preserving Face Detection: A Comprehensive Analysis of Face Anonymization Techniques**. Dissertação (Mestrado) — Universidade do Porto, 2023.

ANGUS, A. *et al.* Real-time video anonymization in smart city intersections. In: **2022 IEEE 19th International Conference on Mobile Ad Hoc and Smart Systems (MASS)**. [S.l.: s.n.], 2022. p. 514–522.

Autoridade Nacional de Proteção de Dados. **Estudo Técnico sobre Anonimização de Dados na LGPD: Uma Visão de Processo Baseado em Risco e Técnicas Computacionais**. Brasília, DF, 2023. Disponível em: <https://www.gov.br/anpd/pt-br/centrais-de-conteudo/documentos-tecnicos-orientativos/estudo_tecnico_sobre_anonimizacao_de_dados_na_lgpd_uma_visao_de_processo_baseado_em_risco_e_tecnicas_computacionais.pdf>.

BEEK, P. J. L. van *et al.* Semantic segmentation of videophone image sequences. In: MARAGOS, P. (Ed.). **Visual Communications and Image Processing '92**. [S.l.]: SPIE, 1992.

BELHUMEUR, P. N.; HESPANHA, J. P.; KRIEGMAN, D. J. Eigenfaces vs. fisherfaces: recognition using class specific linear projection. **IEEE Trans. Pattern Anal. Mach. Intell.**, Institute of Electrical and Electronics Engineers (IEEE), v. 19, n. 7, p. 711–720, jul. 1997.

BHUTANI, K.; FARSAIE, A. Fuzzy approach to a neural network. In: **[Proceedings] 1991 IEEE International Joint Conference on Neural Networks**. [S.l.: s.n.], 1991. p. 1675–1680 vol.2.

BOCHKOVSKIY, A.; WANG, C.-Y.; LIAO, H.-Y. M. **YOLOv4: Optimal Speed and Accuracy of Object Detection**. 2020. Disponível em: <https://arxiv.org/abs/2004.10934>.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: **Proceedings of the fifth annual workshop on Computational learning theory**. New York, NY, USA: ACM, 1992.

BURL *et al.* Face localization via shape statistics. In: ____. [S.l.]: University of Zurich, 1995. Funding by Center for Neuromorphic Systems Engineering, Caltech.

CHEN, R. *et al.* Simswap: An efficient framework for high fidelity face swapping. In: **Proceedings of the 28th ACM International Conference on Multimedia**. New York, NY, USA: Association for Computing Machinery, 2020. (MM '20), p. 2003–2011. ISBN 9781450379885. Disponível em: <https://doi.org/10.1145/3394171.3413630>.

CHINOMI, K. *et al.* Prisurv: Privacy protected video surveillance system using adaptive visual abstraction. In: SATOH, S.; NACK, F.; ETOH, M. (Ed.). **Advances in Multimedia Modeling**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. p. 144–154. ISBN 978-3-540-77409-9.

COOTES, T.; EDWARDS, G.; TAYLOR, C. Active appearance models. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 23, n. 6, p. 681–685, 2001.

COOTES, T. F.; TAYLOR, C. J. Active shape models — 'smart snakes'. In: **BMVC92**. London: Springer London, 1992. p. 266–275.

CRESWELL, A. *et al.* Generative adversarial networks: An overview. **IEEE Signal Processing Magazine**, v. 35, n. 1, p. 53–65, 2018.

CROWLEY, J. L.; COUTAZ, J. Vision for man machine interaction. In: ____. **Engineering for Human-Computer Interaction: Proceedings of the IFIP TC2/WG2.7 working conference on engineering for human-computer interaction, Yellowstone Park, USA, August 1995**. Boston, MA: Springer US, 1996. p. 28–45. ISBN 978-0-387-34907-7. Disponível em: <https://doi.org/10.1007/978-0-387-34907-7_3>.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: **2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)**. [S.l.: s.n.], 2005. v. 1, p. 886–893 vol. 1.

DENG, J. *et al.* Retinaface: Single-shot multi-level face localisation in the wild. In: **2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2020. p. 5202–5211.

____. Arcface: Additive angular margin loss for deep face recognition. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, Institute of Electrical and Electronics Engineers (IEEE), v. 44, n. 10, p. 5962–5979, out. 2022. ISSN 1939-3539. Disponível em: <http://dx.doi.org/10.1109/TPAMI.2021.3087709>.

FAN, L. Image pixelization with differential privacy. In: **Data and Applications Security and Privacy XXXII**. Cham: Springer International Publishing, 2018, (Lecture notes in computer science). p. 148–162.

FISHER, R. A. The use of multiple measurements in taxonomic problems. **Ann. Eugen.**, Wiley, v. 7, n. 2, p. 179–188, set. 1936.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>.

GOODFELLOW, I. J. *et al.* **Generative Adversarial Networks**. 2014. Disponível em: <https://arxiv.org/abs/1406.2661>.

GROSS, R. *et al.* **The CMU Multi-PIE Face Database**. [S.l.], 2006. Forthcoming.

GROSS, R.; SWEENEY, L. Towards real-world face de-identification. In: **2007 First IEEE International Conference on Biometrics: Theory, Applications, and Systems**. [S.l.: s.n.], 2007. p. 1–8.

GROSS, R. *et al.* Model-based face de-identification. In: **2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)**. [S.l.: s.n.], 2006. p. 161–161.

HASAN, M. K. *et al.* Human face detection techniques: A comprehensive review and future research directions. **Electronics**, v. 10, n. 19, 2021. ISSN 2079-9292. Disponível em: <https://www.mdpi.com/2079-9292/10/19/2354>.

HE, K. *et al.* Spatial pyramid pooling in deep convolutional networks for visual recognition. In: ____. **Computer Vision – ECCV 2014**. Springer International Publishing, 2014. p. 346–361. ISBN 9783319105789. Disponível em: <http://dx.doi.org/10.1007/978-3-319-10578-9_23>.

HIDAYATULLAH, P. *et al.* **YOLOv8 to YOLO11: A Comprehensive Architecture In-depth Comparative Review**. 2025. Disponível em: <https://arxiv.org/abs/2501.13400>.

HJELMÅS, E.; LOW, B. K. Face detection: A survey. **Comput. Vis. Image Underst.**, Elsevier BV, v. 83, n. 3, p. 236–274, set. 2001.

HOTELLING, H. Analysis of a complex of statistical variables into principal components. **J. Educ. Psychol.**, American Psychological Association (APA), v. 24, n. 6, p. 417–441, set. 1933.

HUANG, G. B. *et al.* **Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments**. [S.l.], 2007.

HUANG, X.; BELONGIE, S. **Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization**. 2017. Disponível em: <https://arxiv.org/abs/1703.06868>.

HUKKELÅS, H.; MESTER, R.; LINDSETH, F. DeepPrivacy: A generative adversarial network for face anonymization. In: **Lecture Notes in Computer Science**. Cham: Springer International Publishing, 2019, (Lecture notes in computer science). p. 565–578.

IBGE – Instituto Brasileiro de Geografia e Estatística. **Acesso à internet e à televisão e posse de telefone móvel celular para uso pessoal 2022 / IBGE, Coordenação de Pesquisas por Amostra de Domicílios**. Rio de Janeiro, RJ: IBGE, 2023. Informativo Digital.

JAICHUEN, T. *et al.* Blur track: Real-time face detection with immediate blurring and efficient tracking. In: **2023 20th International Joint Conference on Computer Science and Software Engineering (JCSSE)**. [S.l.: s.n.], 2023. p. 167–172.

JIANG, H.; LEARNED-MILLER, E. Face detection with the faster r-cnn. In: **2017 12th IEEE International Conference on Automatic Face Gesture Recognition (FG 2017)**. [S.l.: s.n.], 2017. p. 650–657.

JOCHER, G.; CHAURASIA, A.; QIU, J. **Ultralytics YOLOv8**. 2023. Disponível em: <https://github.com/ultralytics/ultralytics>.

KARRAS, T. *et al.* **Progressive Growing of GANs for Improved Quality, Stability, and Variation**. 2018. Disponível em: <https://arxiv.org/abs/1710.10196>.

____. **Progressive Growing of GANs for Improved Quality, Stability, and Variation**. 2018. Disponível em: <https://arxiv.org/abs/1710.10196>.

____. **Alias-Free Generative Adversarial Networks**. 2021. Disponível em: <https://arxiv.org/abs/2106.12423>.

KARRAS, T.; LAINE, S.; AILA, T. A style-based generator architecture for generative adversarial networks. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 43, n. 12, p. 4217–4228, 2021.

KARRAS, T. *et al.* Analyzing and improving the image quality of stylegan. In: **2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2020. p. 8107–8116.

KASS, M.; WITKIN, A.; TERZOPOULOS, D. Snakes: Active contour models. **Int. J. Comput. Vis.**, Springer Science and Business Media LLC, v. 1, n. 4, p. 321–331, jan. 1988.

KECHICHE, L.; TOUIL, L.; OUNI, B. Real-time image and video processing: Method and architecture. In: **2016 2nd International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)**. [S.l.: s.n.], 2016. p. 194–199.

KEVAL, H.; SASSE, M. A. to catch a thief – you need at least 8 frames per second. In: **Proceedings of the 16th ACM international conference on Multimedia**. New York, NY, USA: ACM, 2008. p. 941–944.

KIM, R. *et al.* Accelerating face de-identification system for real-time video surveillance services. In: **2022 13th International Conference on Information and Communication Technology Convergence (ICTC)**. [S.l.: s.n.], 2022. p. 1477–1479.

KORSHUNOV, P.; EBRAHIMI, T. Using warping for privacy protection in video surveillance. In: **2013 18th International Conference on Digital Signal Processing (DSP)**. [S.l.: s.n.], 2013. p. 1–6.

KROGH, A.; HERTZ, J. A. A simple weight decay can improve generalization. In: **Proceedings of the 5th International Conference on Neural Information Processing Systems**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1991. (NIPS'91), p. 950–957. ISBN 1558602224.

KUMAR *et al.* Face detection techniques: a review. **Artif. Intell. Rev.**, Springer Science and Business Media LLC, v. 52, n. 2, p. 927–948, ago. 2019.

KUNG, S.; TAUR, J. Decision-based neural networks with signal/image classification applications. **IEEE Transactions on Neural Networks**, v. 6, n. 1, p. 170–181, 1995.

KWOLEK, B. CamShift-based tracking in joint color-spatial spaces. In: **Computer Analysis of Images and Patterns**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, (Lecture notes in computer science). p. 693–700.

LANITIS, A.; COOTES, T. F.; TAYLOR, C. J. Automatic tracking, coding and reconstruction of human faces, using flexible appearance models. **Electron. Lett.**, Institution of Engineering and Technology (IET), v. 30, n. 19, p. 1587–1588, set. 1994.

LEI, Z. *et al.* A lightweight real-time face de-identification privacy-preserving algorithm based on airport equipment. In: **2024 IEEE 7th International Conference on Information Systems and Computer Aided Education (ICISCAE)**. [S.l.: s.n.], 2024. p. 408–412.

LI, Z. *et al.* A survey of convolutional neural networks: Analysis, applications, and prospects. **IEEE Transactions on Neural Networks and Learning Systems**, v. 33, n. 12, p. 6999–7019, 2022.

Lindevs. **YOLOv8-face: Face detection using YOLOv8**. [S.l.]: GitHub, 2023. <https://github.com/lindevs/yolov8-face>. Acessado em: 12 maio 2026.

LIU, W. *et al.* SSD: Single shot MultiBox detector. **arXiv [cs.CV]**, arXiv, 2015.

MAXIMOV, M.; ELEZI, I.; LEAL-TAIXE, L. CIAGAN: Conditional identity anonymization generative adversarial networks. In: **2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.]: IEEE, 2020.

MEDEN, B. *et al.* Privacy–enhancing face biometrics: A comprehensive survey. **IEEE Transactions on Information Forensics and Security**, v. 16, p. 4147–4183, 2021.

MOGHADDAM, B.; PENTLAND, A. Probabilistic visual learning for object representation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 19, n. 7, p. 696–710, 1997.

MORE, R. *et al.* Privacy-preserving video analytics through gan-based face de-identification. In: **2024 Second International Conference on Networks, Multimedia and Information Technology (NMITCON)**. [S.l.: s.n.], 2024. p. 1–6.

MRITYUNJAY; NARAYANAN, P. The de-identification camera. In: **2011 Third National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics**. [S.l.: s.n.], 2011. p. 192–195.

NEWTON, E.; SWEENEY, L.; MALIN, B. Preserving privacy by de-identifying face images. **IEEE Transactions on Knowledge and Data Engineering**, v. 17, n. 2, p. 232–243, 2005.

NVIDIA. **NVIDIA TensorRT, an SDK for high-performance deep learning inference**. 2022. <https://developer.nvidia.com/tensorrt>. Acessado em 2025.

O'SHEA, K.; NASH, R. **An Introduction to Convolutional Neural Networks**. 2015. Disponível em: <https://arxiv.org/abs/1511.08458>.

PEREZ, M.; KOT, A. C.; ROCHA, A. Detection of real-world fights in surveillance videos. In: **ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. [S.l.: s.n.], 2019. p. 2662–2666.

PHILLIPS, P. *et al.* The feret evaluation methodology for face-recognition algorithms. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 22, n. 10, p. 1090–1104, 2000.

PRECHELT, L. Early stopping-but when? In: **Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop**. Berlin, Heidelberg: Springer-Verlag, 1998. p. 55–69. ISBN 3540653112.

PURWONO, P. *et al.* Understanding of convolutional neural network (cnn): A review. **International Journal of Robotics and Control Systems**, v. 2, n. 4, p. 739–748, 2023. ISSN 2775-2658. Disponível em: <https://pubs2.ascee.org/index.php/IJRCS/article/view/888>.

QI, D. *et al.* Yolo5face: Why reinventing a face detector. In: KARLINSKY, L.; MICHAELI, T.; NISHINO, K. (Ed.). **Computer Vision – ECCV 2022 Workshops**. Cham: Springer Nature Switzerland, 2023. p. 228–244. ISBN 978-3-031-25072-9.

RAYCHAUDHURI, D. *et al.* Challenge: Cosmos: A city-scale programmable testbed for experimentation with advanced wireless. In: **Proceedings of the 26th Annual International Conference on Mobile Computing and Networking**. New York, NY, USA: ACM, 2020.

REDMON, J. *et al.* You only look once: Unified, real-time object detection. In: **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.]: IEEE, 2016.

REDMON, J.; FARHADI, A. **YOLOv3: An Incremental Improvement**. 2018. Disponível em: <https://arxiv.org/abs/1804.02767>.

SAKAI, T. Computer analysis and classification of photographs of human faces. In: **Proceedings of the First USA–Japan Computer Conference**. Tokyo, Japan: [s.n.], 1972. p. 55–62.

SARWAR, O.; CAVALLARO, A.; RINNER, B. Temporally smooth privacy-protected airborne videos. In: **2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**. [S.l.]: IEEE, 2018.

SERENGIL, S. I.; OZPINAR, A. Lightface: A hybrid deep face recognition framework. In: **2020 Innovations in Intelligent Systems and Applications Conference (ASYU)**. [S.l.: s.n.], 2020. p. 1–5.

SHVAI, N.; CARMONA, A. L.; NAKIB, A. Adaptive image anonymization in the context of image classification with neural networks. In: **2023 IEEE/CVF International Conference on Computer Vision (ICCV)**. [S.l.: s.n.], 2023. p. 5051–5060.

SOHAN, M.; RAM, T. S.; REDDY, C. V. R. A review on yolov8 and its advancements. In: JACOB, I. J.; PIRAMUTHU, S.; FALKOWSKI-GILSKI, P. (Ed.). **Data Intelligence and Cognitive Informatics**. Singapore: Springer Nature Singapore, 2024. p. 529–545. ISBN 978-981-99-7962-2.

SRIVASTAVA, A. *et al.* A survey of face detection algorithms. In: **2017 International Conference on Inventive Systems and Control (ICISC)**. [S.l.: s.n.], 2017. p. 1–4.

SRIVASTAVA, N. *et al.* Dropout: A simple way to prevent neural networks from overfitting. **Journal of Machine Learning Research**, v. 15, n. 56, p. 1929–1958, 2014. Disponível em: <http://jmlr.org/papers/v15/srivastava14a.html>.

SUN, Z.; OZAY, M.; OKATANI, T. **Design of Kernels in Convolutional Neural Networks for Image Classification**. 2016. Disponível em: <https://arxiv.org/abs/1511.09231>.

SZELISKI, R. **Computer Vision: Algorithms and Applications**. 2nd. ed. Berlin, Heidelberg: Springer-Verlag, 2022.

TRAN, D. T. *et al.* Privacy-preserving face and hair swapping in real-time with a gan-generated face image. **IEEE Access**, v. 12, p. 179265–179280, 2024.

TURK, M.; PENTLAND, A. Face recognition using eigenfaces. In: **Proceedings. 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 1991. p. 586–591.

Ultralytics. **YOLOv5**. 2024. <https://github.com/ultralytics/yolov5>. Acessado em 09/06/2025.

VARGHESE, R.; M., S. Yolov8: A novel object detection algorithm with enhanced performance and robustness. In: **2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)**. [S.l.: s.n.], 2024. p. 1–6.

VIOLA, P.; JONES, M. J. Robust real-time face detection. **International Journal of Computer Vision**, v. 57, n. 2, p. 137–154, May 2004. ISSN 1573-1405. Disponível em: <https://doi.org/10.1023/B:VISI.0000013087.49260.fb>.

WANG, C.-Y.; LIAO, H.-Y. M. **YOLOv1 to YOLOv10: The fastest and most accurate real-time object detection systems**. 2024. Disponível em: <https://arxiv.org/abs/2408.09332>.

WON, J.-H. *et al.* An improved yolov3-based neural network for de-identification technology. In: **2019 34th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC)**. [S.l.: s.n.], 2019. p. 1–2.

YANG, G.; HUANG, T. S. Human face detection in a complex background. **Pattern Recognit.**, Elsevier BV, v. 27, n. 1, p. 53–63, jan. 1994.

YANG, S. *et al.* **WIDER FACE: A Face Detection Benchmark**. 2016. Accessed: April 19, 2025. Disponível em: <http://shuoyang1213.me/WIDERFACE/>.

YOUNESI, A. *et al.* **A Comprehensive Survey of Convolutions in Deep Learning: Applications, Challenges, and Future Trends**. 2024. Disponível em: <https://arxiv.org/abs/2402.15490>.

YUILLE, A. L.; HALLINAN, P. W.; COHEN, D. S. Feature extraction from faces using deformable templates. **Int. J. Comput. Vis.**, Springer Science and Business Media LLC, v. 8, n. 2, p. 99–111, ago. 1992.

ZADEH, L. Fuzzy sets. **Information and Control**, v. 8, n. 3, p. 338–353, 1965. ISSN 0019-9958. Disponível em: <https://www.sciencedirect.com/science/article/pii/S001999586590241X>.

ZHU, H. *et al.* CelebV-HQ: A large-scale video facial attributes dataset. In: **ECCV**. [S.l.: s.n.], 2022.

ZIVKOVIC, Z. Improved adaptive gaussian mixture model for background subtraction. In: **Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.** [S.l.: s.n.], 2004. v. 2, p. 28–31 Vol.2.

APÊNDICE A – DECLARAÇÃO DE USO DE INTELIGÊNCIA ARTIFICIAL

Este trabalho contou com o apoio de ferramentas de inteligência artificial para determinadas etapas de seu desenvolvimento. As ferramentas utilizadas tiveram a função de auxiliar na organização de ideias, na revisão textual e na verificação de clareza e coesão, sem substituir a análise crítica, o conteúdo autoral e a responsabilidade do autor pelas informações apresentadas. Foi utilizado o modelo Gemini 3.5 [1] para revisão ortográfica e realizar ajustes de fluidez do texto.

[1] https://gemini.google.com/

ANEXO A – CONJUNTO DE DADOS DE TESTE

O quadro abaixo lista os identificadores dos vídeos da base de dados CelebV-HQ utilizados nos testes, cada vídeo pode conter um ou mais clipes.

0O35rDdcoL0,	0YGtaYHJQaU,	1R_wzGrOLFs,	3LNfOrrXKvI,	3NUMcmRDlDU,
3nf2j5uWKmE,	3otEW3PQG2Y,	5MJR74vZgX0,	E7fYM97TrZ8,	F-57KCDK0Ps,
F9SedpzIRYg,	JNrH1W7aKc8,	MUvcpjeUFpU,	P8ZCZJpluDA,	TJ_a9ckoHyU,
U2vVv9BzKos,	UoiQ3ZrTo7E,	UssIXhT7bm0,	ZVYqgUcQBbk,	Z_9KUBCidow,
au6h8bNE8ec,	bYlPxQI8uYw,	cuGs0oSqpBE,	hK2Em4D7fhU,	hlKMLkrSrDo,
iViIy5ci5yU,	j76m3AqMhOc,	kVTYhZ0NBOs,	kaX-XfcAZY8,	l6LfwtdM8Fg,
q3kIwrng5aA,	qw62N4v8Cwo,	tgBlsrzJx5g,	tjBBL2ZtHt0,	wBM9Aa_HG8g,
xr1V8HDt8GQ,	zW2T9TkSIcM.			

ANEXO B – DETECÇÃO FACIAL

O Algoritmo 1 apresenta a implementação da etapa de detecção facial. Nesta etapa o modelo YOLO é inicializado com os pesos pré-treinados selecionados. Com isso, executa-se a inferência sobre o quadro de vídeo, resultando em uma lista de predições. Cada predição representa uma face identificada, a qual é composta pelo índice de confiança e pelas coordenadas da caixa delimitadora da face.

Algoritmo 1 – Detecção facial em quadros de vídeo

```
class FaceDetector:
def __init__(self, model_size = "n"):
self.model = YOLO(f"model-params/yolov8{model_size}-face-lindevs.pt")

def detect(self, video_frame):
results = self.model(video_frame, verbose=False)
return self.__extract_coords(results)

def __extract_coords(self, results):
detections = []

for result in results:
  for box in result.boxes:
    coords = box.xyxy[0].tolist()
    confidence = box.conf[0].item()

    detections.append({
      "confidence": confidence,
      "coords": coords
    })

return detections
```

Fonte: O Autor (2026)

ANEXO C – ANONIMIZAÇÃO FACIAL POR DESFOQUE

O Algoritmo 2 detalha a aplicação do desfoque sobre as faces detectadas no quadro de vídeo. Pra cada região identificada, aplica-se o desfoque gaussiano por meio do *framework OpenCV*. O tamanho do *kernel* utilizado é calculado dinamicamente com base na largura da face, garantindo que a eficácia da anonimização seja preservada independentemente da escala da face no quadro.

Algoritmo 2 – Anonimização facial por desfoque Gaussiano

```
def _process_frame(self, frame):
  detections = self.yolo_detector.detect(frame)

  if len(detections) == 0:
    return frame

  if self.mode == 'blur':
    self._blur_faces(frame, detections)

  if self.mode == 'swap':
    self._swap_face(frame, detections)

  return frame

def _blur_faces(self, frame, detections):
  BLUR_INTENSITY = 2

  for detection in detections:
    x1, y1, x2, y2 = map(int, detection["coords"])
    face_roi = frame[y1:y2, x1:x2]

    face_w = x2 - x1
    k = max(1, (face_w // BLUR_INTENSITY) | 1)

    blurred_roi = cv2.GaussianBlur(face_roi, (k, k), 0)
    frame[y1:y2, x1:x2] = blurred_roi

  return frame
```

Fonte: O Autor (2026)

ANEXO D – CRIAÇÃO DE FACES SINTÉTICAS

O Algoritmo 3 apresenta a implementação da etapa de geração de faces sintéticas utilizando o modelo *StyleGAN*. O processo inicia-se com o carregamento de pesos pré-treinados do modelo. A partir disso, a execução do modelo resulta em uma imagem de alta fidelidade, que mantém realismo e serve como a identidade de origem para o processo de troca de identidade.

Algoritmo 3 – Criação de face sintética com StyleGAN

```
class StyleGANGenerator:
  def __init__(self):
    self.device = torch.device("cuda")
    self.G = self._load_model('model-params/stylegan2-ffhq.pkl')

  def generate(self, seed = None):
    if seed is None:
      seed = np.random.ranint(0, 999999)

    z = torch.from_numpy(np.random.RandomState(seed) \
          .randn(1, self.G.z_dim)).to(self.device)

    with torch.no_grad():
        img = self.G(z, None, truncation_psi=0.8, noise_mode='const')
        img = (img.permute(0, 2, 3, 1) * 127.5 + 128) \
                .clamp(0, 255).to(torch.uint8)
        return Image.fromarray(img[0].cpu().numpy(), 'RGB')

  def _load_model(self, pkl_path):
    with open(pkl_path, "rb") as f:
      data = pickle.load(f)
     G = data["G_ema"].to(self.device)
    return G
```

ANEXO E – TROCA DE FACES

O Algoritmo 4 detalha a implementação da troca de faces. Durante o processamento, as características da face detectada no vídeo são integradas à identidade sintética por meio de um processo de transferência de atributos. Por fim, a face resultante é reintegrada ao quadro original, utilizando técnicas de segmentação e normalização para preservar a iluminação, pose e as expressões faciais.

Algoritmo 4 – Troca de faces com SimSwap

```
class SimSwap:
  def __init__(self, stylegan_image_path):
    opt = TestOptions()
    opt.initialize()
    opt.parser.add_argument('-f')
    self.opt = opt.parse()
    self.opt.isTrain = False
    self.opt.use_mask = True

    self.opt.crop_size = 512
    self.opt.which_epoch = 550000
    self.opt.name = '512'
    mode = 'ffhq'

    torch.nn.Module.dump_patches = True
    self.model = create_model(self.opt)
    self.model.eval()

    n_classes = 19
    self.net = BiSeNet(n_classes=n_classes)
    self.net.cuda()
    save_pth = os.path.join('./parsing_model/checkpoint', '79999_iter.pth')
    self.net.load_state_dict(torch.load(save_pth))
    self.net.eval()

    self.spNorm = SpecificNorm()

    self.app = Face_detect_crop(
        name='antelope',
        root='./insightface_func/models')
    self.app.prepare(
        ctx_id=0,
        det_thresh=0.3,
        det_size=(640,640), mode=mode)

```

```
36      self.transformer_Arcface = transforms.Compose([
37          transforms.ToTensor(),
38          transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
39      ])
40
41      self._prepare_latent_id(stylegan_image_path)
42
43    def process(self, roi_frame):
44      with torch.no_grad():
45        target_info = self.app.get(roi_frame, self.opt.crop_size)
46
47        if target_info is None or len(target_info[0]) == 0:
48          return roi_frame
49
50        target_align_crop_list, target_mat_list = target_info
51
52        target_align_crop = target_align_crop_list[0]
53        target_mat = target_mat_list[0]
54
55        target_align_crop_tensor = self._totensor(
56            cv2.cvtColor(target_align_crop, cv2.COLOR_BGR2RGB)
57        )[None,...].cuda()
58
59        swap_result = self.model(
60            None,
61            target_align_crop_tensor,
62            self.latend_id, None, True)[0]
63
64        result = reverse2wholeimage(
65            [target_align_crop_tensor],
66            [swap_result],
67            [target_mat],
68            self.opt.crop_size,
69            roi_frame,
70            None,
71            None,
72            True,
73            pasring_model=self.net,
74            use_mask=self.opt.use_mask,
75            norm=self.spNorm,
76            save_final_img = False
77        )
78        return result
79
80    def _prepare_latent_id(self, img_path):
81      with torch.no_grad():
82        img_a_whole = cv2.imread(img_path)
```

```
        result = self.app.get(img_a_whole, self.opt.crop_size)
        img_a_align_crop, _ = result

        img_a_align_crop_pil = Image.fromarray(
            cv2.cvtColor(img_a_align_crop[0], cv2.COLOR_BGR2RGB))
        img_a = self.transformer_Arcface(img_a_align_crop_pil)
        img_id = img_a.view(
            -1,
            img_a.shape[0], img_a.shape[1], img_a.shape[2]
        ).cuda()

        img_id_downsample = F.interpolate(img_id, size=(112, 112))
        self.latend_id = self.model.netArc(img_id_downsample)
        self.latend_id = F.normalize(self.latend_id, p=2, dim=1)

    def _totensor(self, array):
        tensor = torch.from_numpy(array)
        img = tensor.transpose(0, 1).transpose(0, 2).contiguous()
        return img.float().div(255)
```