



Instituto Federal Catarinense
Curso de Bacharelado em Ciência da Computação
Campus Blumenau

SANDY HOFFMANN

**TÉCNICAS DE HASHING E REDES NEURAIS CONVOLUCIONAIS
APLICADAS À BUSCA REVERSA DE IMAGENS**

Blumenau
2024

SANDY HOFFMANN

**TÉCNICAS DE HASHING E REDES NEURAIS CONVOLUCIONAIS
APLICADAS À BUSCA REVERSA DE IMAGENS**

Trabalho de Conclusão de Curso submetido ao Curso de Bacharelado em Ciência da Computação do Instituto Federal Catarinense — Campus Blumenau para a obtenção do título de Bacharel em Ciência da Computação.

Orientador: Ricardo de la Rocha Ladeira, Me.

Blumenau

2024


SANDY HOFFMANN

**TÉCNICAS DE HASHING E REDES NEURAIS CONVOLUCIONAIS
APLICADAS À BUSCA REVERSA DE IMAGENS**


Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de “Bacharel em Ciência da Computação” e aprovado em sua forma final pelo curso de Bacharelado em Ciência da Computação do Instituto Federal Catarinense — Campus Blumenau.

Blumenau, 16 de Dezembro de 2024.


Banca Examinadora:

Documento assinado digitalmente
 **RICARDO DE LA ROCHA LADEIRA**
Data: 17/12/2024 00:17:00-0300
Verifique em <https://validar.iti.gov.br>

Prof. Ricardo de la Rocha Ladeira, Me.
Orientador – IFC Campus Blumenau

 Documento assinado digitalmente
HYLSON VESCOVI NETTO
Data: 17/12/2024 15:30:45-0300
CPF: ***.166.897-**
Verifique as assinaturas em <https://v.ufsc.br>

Prof. Hylson Vescovi Netto, Dr.
IFC Campus Blumenau

Documento assinado digitalmente
 **FABRICIO ALVES OLIVEIRA**
Data: 17/12/2024 10:10:10-0300
Verifique em <https://validar.iti.gov.br>

Prof. Fabricio Alves Oliveira, Dr.
IFC Campus Blumenau

Este trabalho é dedicado aos meus colegas e professores do
IFC Campus Blumenau.

AGRADECIMENTOS

Aos meus pais, Charles e Cristiane, por terem me incentivado e me apoiado durante os meus estudos.

Ao IFC Campus Blumenau por me dar oportunidade de concluir tanto o ensino médio técnico em informática quanto o bacharelado em ciências da computação. Foram sete anos de muito aprendizado, aos quais só tenho a agradecer.

Aos meus professores, que foram fontes de inspiração, transmitindo conhecimento e motivação ao longo dessa jornada. Em especial, ao professor Ricardo Ladeira, que aceitou ser meu orientador e me ajudou durante todo o processo do TCC.

Aos meus colegas de classe, que me deram muito suporte durante esse tempo de faculdade.

Agradeço também a Deus, que sempre esteve comigo.

As máquinas me surpreendem muito frequentemente. (TURING, 1950)

RESUMO

Este trabalho explora técnicas aplicadas à busca reversa de imagens, focando no uso de *hashing* e redes neurais convolucionais, afim de comparar suas taxas de acurácia e média de desempenho de tempo. Durante os experimentos, foram utilizadas imagens manipuladas com efeitos como desfoque gaussiano, dessaturação (para escala de cinza), alteração de resolução, realce de nitidez, rotação (90°), espelhamento, corte e inserção de elementos gráficos. O *hashing* perceptivo é um tipo de *hash* que aplica menor variação em sua saída diante de pequenas modificações no seu conteúdo de entrada. Foi analisada sua variação utilizando a transformada direta de cosseno (DCT). Também foi explorado o uso da rede neural convolucional (CNN), um modelo de inteligência artificial frequentemente empregado em tarefas relacionadas a imagens, utilizando operações de convolução para extrair características específicas que podem ser aplicadas na busca reversa. Nos testes realizados, a rede neural convolucional demonstrou maior robustez frente a alterações como corte, rotação e espelhamento. Além disso, o trabalho apresenta uma solução baseada em *hashing* sensível à localidade aleatório (LSH), uma técnica de particionamento que emprega hiperplanos aleatórios para segmentar o espaço de busca. A aplicação do LSH juntamente com a rede neural convolucional visou aumentar a assertividade da busca e diminuir o tempo de execução. Os resultados do uso do LSH mostraram uma melhoria na acurácia em relação aos resultados da CNN, juntamente com a diminuição do tempo médio de execução. Conclui-se que, em comparação ao DCT, o uso dos vetores característicos extraídos das camadas da CNN é uma abordagem mais robusta. Além disso, o LSH, ao reduzir o espaço de busca, contribui para aumentar a taxa de assertividade da pesquisa e diminuir o tempo médio de execução.

Palavras-chave: Busca Reversa de Imagens; Hashing Perceptivo; Redes Neurais Convolucionais; Hashing Sensível à Localidade.

ABSTRACT

This work explores techniques applied to reverse image search, focusing on the use of hashing and convolutional neural networks to compare their accuracy rates and average execution time performance. During the experiments, images were manipulated with effects such as Gaussian blur, desaturation (to grayscale), resolution change, sharpness enhancement, rotation (90°), mirroring, cropping, and the insertion of graphic elements. Perceptual hashing is a type of hash that exhibits minimal variation in its output in response to small modifications in its input content. Its variation was analyzed using the discrete cosine transform (DCT). The use of convolutional neural networks (CNN), an artificial intelligence model commonly applied to image-related tasks, was also explored. CNNs use convolution operations to extract specific features that can be applied to reverse search. In the tests conducted, the convolutional neural network demonstrated greater robustness against alterations such as cropping, rotation, and mirroring. Additionally, the work presents a solution based on random locality-sensitive hashing (LSH), a partitioning technique that uses random hyperplanes to segment the search space. The application of LSH combined with the convolutional neural network aimed to increase search accuracy and reduce execution time. The results of using LSH showed an improvement in accuracy compared to CNN results, along with a reduction in average execution time. It is concluded that, compared to DCT, the use of characteristic vectors extracted from the CNN layers is a more robust approach. Moreover, LSH, by reducing the search space, contributes to increasing the accuracy rate of the search and decreasing the average execution time.

Keywords: Reverse Image Search; Perceptual Hashing; Convolutional Neural Network; Locality Sensitive Hashing.

LISTA DE FIGURAS

Figura 1 – Possíveis alterações visuais nas imagens.	13
Figura 2 – Exemplos práticos de RIS.	19
Figura 3 – Exemplos de mecanismos de busca utilizando imagens.	20
Figura 4 – Exemplo de ataque de segunda pré-imagem direcionada.	22
Figura 5 – Exemplo de aplicação da distância de Hamming.	23
Figura 6 – Exemplos de geração de DCT em imagens modificadas.	24
Figura 7 – Representação do particionamento dos dados utilizando LSH aleatório.	25
Figura 8 – Organização das camadas da rede VGG16.	27
Figura 9 – Efeito do filtro de Sobel sobre as imagens.	28
Figura 10 – Forma de avaliação de acurácia.	33
Figura 11 – Porcentagem de acertos obtidos com as técnicas DCT, CNN e LSH.	38

LISTA DE TABELAS

Tabela 1	– Distâncias de Hamming entre as imagens da Figura 6.	25
Tabela 2	– Configurações do Computador.	34
Tabela 3	– Porcentagem de acertos obtidos com as técnicas DCT e CNN (Total de 4500 imagens).	36
Tabela 4	– Média de tempo de execução obtida com as técnicas DCT e CNN. . . .	36
Tabela 5	– Porcentagem de acertos obtidos com as técnicas CNN e LSH (Total de 4500 imagens).	37
Tabela 6	– Média de tempo de execução obtida com as técnicas CNN e LSH. . . .	37
Tabela 7	– Média de tempo de execução obtida com as técnicas DCT, CNN e LSH.	38

LISTA DE ABREVIATURAS E SIGLAS

CBIR	<i>Content-based image retrieval</i> – Recuperação de imagem baseada em conteúdo
CNN	<i>Convolutional Neural Network</i> – Rede Neural Convolucional
DCT	<i>Discrete Cosine Transformation</i> – Transformada Discreta do Cosseno
DFT	<i>Discrete Fourier Transformation</i> – Transformada Discreta de Fourier
JPEG	<i>Joint Photographic Experts Group</i>
LSH	<i>Locality Sensitive Hashing</i> – <i>Hashing</i> sensível à Localidade
ReLU	<i>Rectified Linear Unit</i> – Unidade Linear Retificada
RIS	<i>Reverse Image Search</i> – Busca reversa de imagens
SGD	<i>Stochastic Gradient Descent</i> – Descida Estocástica do Gradiente
VGG16	<i>Convolutional Network for Classification and Detection</i>
XOR	<i>Exclusive or</i> – Ou Exclusivo

SUMÁRIO

1	INTRODUÇÃO	12
1.1	OBJETIVOS	13
1.1.1	Objetivo Geral	13
1.1.2	Objetivos Específicos	14
1.2	JUSTIFICATIVA	14
1.3	METODOLOGIA	15
1.4	REVISÃO BIBLIOGRÁFICA	16
2	FUNDAMENTAÇÃO TEÓRICA	18
2.1	BUSCA REVERSA DE IMAGENS	18
2.2	TÉCNICAS HASHING	20
2.2.1	Hashing Perceptivo	20
2.2.1.1	<i>DCT (Discrete Cosine Transformation – Transformada Discreta do Cosseno)</i>	<i>22</i>
2.2.1.2	<i>Distância de Hamming</i>	<i>22</i>
2.2.1.3	<i>Uso do Hashing Perceptivo no campo da RIS</i>	<i>23</i>
2.2.2	Hashing Sensível à Localidade	24
2.2.2.1	<i>Uso do Hashing Sensível à Localidade no campo da RIS</i>	<i>26</i>
2.3	REDE NEURAL CONVOLUCIONAL	26
2.3.1	Arquitetura	26
2.3.1.1	<i>Camadas de Convolução</i>	<i>27</i>
2.3.1.2	<i>Camadas de Pooling</i>	<i>28</i>
2.3.1.3	<i>Camadas de Ativação</i>	<i>29</i>
2.3.1.3.1	<i>Softmax</i>	<i>29</i>
2.3.1.3.2	<i>ReLU</i>	<i>29</i>
2.3.1.4	<i>Camadas Densas</i>	<i>30</i>
2.3.2	Treinamento	30
2.3.2.1	<i>Preparação dos Dados</i>	<i>30</i>
2.3.2.2	<i>Propagação Direta e Propagação Reversa</i>	<i>31</i>
2.3.3	Uso da CNN no campo da RIS	32
2.3.3.1	<i>Distância Euclidiana</i>	<i>32</i>
3	EXPERIMENTOS	33
3.1	EXPERIMENTO 1	34
3.1.1	Implementação da Técnica RIS utilizando DCT	34
3.1.2	Implementação da Técnica RIS utilizando CNN	34
3.1.3	Análise dos Resultados do Experimento 1	35
3.2	EXPERIMENTO 2	36
3.2.1	Implementação da Técnica RIS utilizando LSH	36

3.2.2	Análise dos Resultados do Experimento 2	37
3.3	COMPARAÇÃO DAS TÉCNICAS DCT, CNN E LSH	38
4	CONCLUSÃO	39
	REFERÊNCIAS	40

1 INTRODUÇÃO

O aumento exponencial do volume de conteúdo disponível na *Internet* trouxe também um desafio: como localizar e recuperar os dados requisitados pelos usuários de maneira eficiente? Há diversas estratégias e mecanismos de busca para extrair conteúdos *online*, destacando-se a pesquisa através de imagens, implementada por plataformas como Bing Imagens¹ e Google Lens². A técnica que permite identificar imagens com conteúdos visuais semelhantes é definida como RIS (*Reverse Image Search* – Busca reversa de imagens), onde o algoritmo facilita a automação da extração das características predominantes nas figuras, a fim de identificar e correlacionar os dados.

A lacuna semântica é um desafio na área da RIS, sendo definida como a complexidade de representar as características das figuras ao algoritmo de maneira parecida com a percepção do ser humano para interpretar a imagem (QAZANFARI; ALYANNEZHADI; KHOSHDAREGI, 2023). Esse conceito consiste na diferença entre como uma máquina percebe os elementos de uma imagem (por meio de características numéricas ou computacionais, por exemplo) e como o ser humano interpreta esses mesmos elementos (considerando contexto, significado).

A capacidade de busca em meio à escalabilidade do volume de dados também se torna um problema crítico, pois conforme há o crescimento da quantidade de informações, o espaço de busca se torna mais denso, afetando a assertividade e a eficiência do sistema (QAZANFARI; ALYANNEZHADI; KHOSHDAREGI, 2023). O algoritmo da RIS também deve se atentar às possíveis modificações digitais que as imagens podem obter estando na *Internet*, como por exemplo, desfoque gaussiano, dessaturação, alteração de resolução, realce de nitidez, rotação, espelhamento, corte, inserção de elementos gráficos, entre outros cenários, conforme ilustrado na Figura 1.

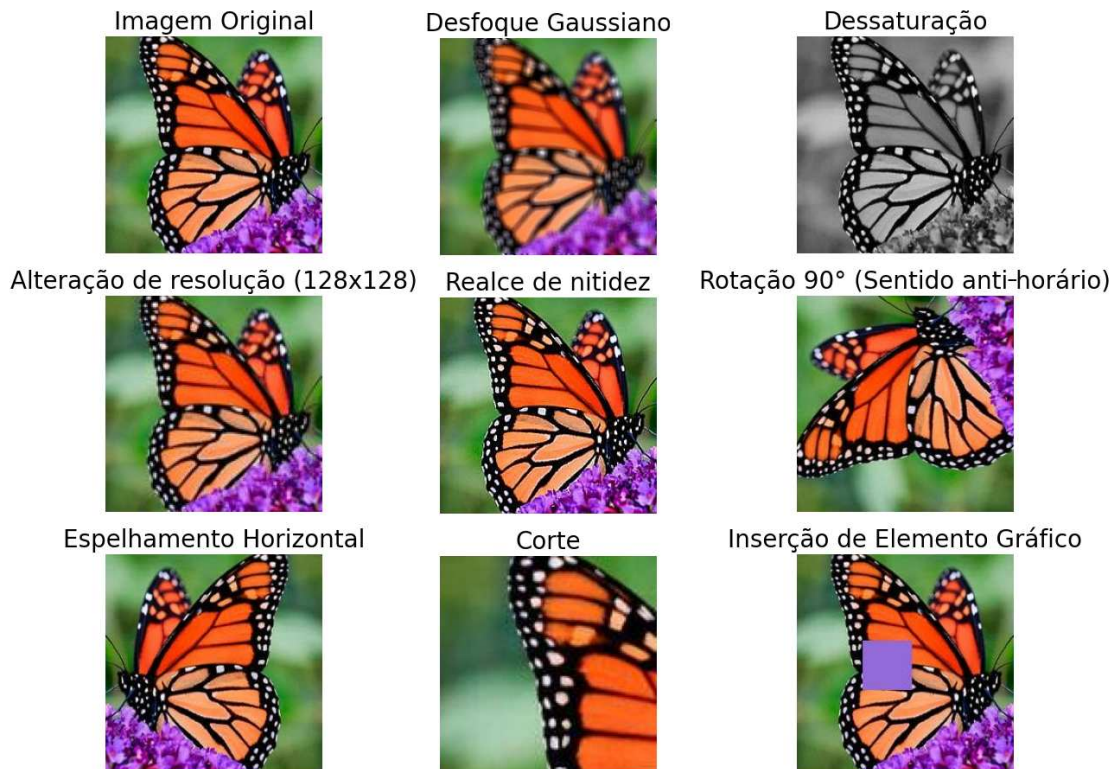
Diferentes algoritmos possuem o objetivo de resolver os desafios enfrentados na busca reversa. Entre eles, nesta monografia, serão abordadas três possibilidades principais: o uso do *hashing* perceptivo, a utilização da CNN (*Convolutional Neural Network* – Rede Neural Convolutacional) e a combinação da CNN com o algoritmo de particionamento LSH (*Locality Sensitive Hashing* – *Hashing* sensível à Localidade).

O *hashing* perceptivo possui uso no campo do RIS, gerando valores de *hashes* semelhantes para imagens com características visuais parecidas (ZHU, Z. *et al.*, 2022). A CNN (*Convolutional Neural Network* – Rede Neural Convolutacional) é uma tecnologia pertencente à inteligência artificial, sendo amplamente utilizada no processamento de imagens (GOODFELLOW; BENGIO; COURVILLE, 2016), podendo ser utilizada no campo da RIS. É possível por meio dessa tecnologia extrair as características das imagens com base nas camadas de treinamento de um modelo de rede previamente treinado e assim retornar seus valores para comparação com a figura alvo da pesquisa (TZELEPI; TEFAS,

¹ Disponível em: <https://www.bing.com/images>

² Disponível em: <https://lens.google.com>

Figura 1 – Possíveis alterações visuais nas imagens.



Fonte: Autora (2024).

2017).

O LSH (*Locality Sensitive Hashing* – *Hashing* sensível à Localidade) é um algoritmo que possui aplicação em espaços com alta dimensionalidade, tendo como objetivo reduzir o custo computacional (RAJARAM; SCHOLZ, 2008).

Considerando os desafios enfrentados no campo da RIS, será abordado em critérios de desempenho e assertividade o uso das tecnologia de *hashing* perceptivo e CNN, levando em conta imagens digitalmente modificadas, conforme Figura 1. Para diminuir o espaço de busca e a fim de melhorar a eficiência do resultado da pesquisa, é utilizado o LSH juntamente com a CNN, particionando os dados a serem pesquisados.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

O objetivo do presente trabalho é comparar os métodos pertencentes à RIS no campo do *hash* perceptivo e da CNN em termos de desempenho de tempo e assertividade. Além disso, explora-se a aplicação do LSH aleatório nos vetores de características gerados pela CNN, com a finalidade de melhorar a taxa de acertos e o tempo da busca.

1.1.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

1. Demonstrar o uso do *hashing* perceptivo para representar a informação em formato *hash*, abrangendo o algoritmo DCT (*Discrete Cosine Transformation* – Transformada Discreta do Cosseno), juntamente com técnicas para calcular proximidade entre as imagens resultantes.
2. Explorar o uso da CNN para o campo da RIS, abrangendo como os vetores de características calculados pela rede são usados na busca reversa de imagens e como é calculada a similaridade com base em experimentos implementados.
3. Explorar a técnica de utilização do LSH (*Locality Sensitive Hashing* – *Hashing* sensível à Localidade) para particionar o espaço de busca proveniente da técnica CNN, reduzindo o espaço de busca e visando aumentar a eficiência computacional.
4. Comparar os métodos implementados utilizando o *hashing* perceptivo, a CNN e a junção do LSH com o espaço de busca proveniente da CNN, considerando métricas de desempenho de tempo e assertividade dos resultados.

1.2 JUSTIFICATIVA

De acordo com pesquisa efetuada por Domo (2022), a cada minuto do dia: 5,9 milhões de buscas são feitas no Google, 66 mil fotos são enviadas ao Instagram e 1,7 milhões de dados são compartilhados no Facebook. Levando em consideração o aumento dos dados armazenados na *Internet*, especialmente imagens, juntamente com a quantidade de pesquisas efetuadas aos mecanismos de pesquisa, é necessária a implementação de algoritmos de pesquisa que sejam assertivos em seus resultados e que tenham um desempenho de tempo aceitável.

O campo da RIS (*Reverse Image Search* – Busca reversa de imagens) explora a recuperação de dados com base no conteúdo das imagens (WAN; LIU, 2008), possuindo aplicações em diversas áreas, como, por exemplo: detecção de violação de direitos autorais (GEORGE; BASKAR; PANDEY, 2024), detecção de fraudes (WEKESA; DECUSATIS; ZHU, A., 2023), reconhecimento de prática de *catfish* em redes sociais (ESCAP *et al.*, 2024), entre outros cenários.

Considerando o cenário de modificações a imagens, é importante que a técnica de RIS mantenha sua taxa de acertos alta mesmo com estas alterações. Conforme citado por Gaillard (2017), há modificações que dificultam o reconhecimento das imagens na busca reversa, como a operação de corte ou rotação.

A presente pesquisa abrange a exploração de diferentes casos de modificações visuais nas figuras, tais quais: desfoque gaussiano, dessaturação, alteração de resolução, realce

de nitidez, rotação, espelhamento, corte e inserção de elementos gráficos, ilustrados na Figura 1. Assim, são adicionadas novas possibilidades aos estudos já efetuados, listados na Seção 1.4.

1.3 METODOLOGIA

Para definir a metodologia é seguido a caracterização dos tipos de pesquisa científica estabelecidos por Wazlawick (2020), onde tem-se três critérios de classificação: em relação à sua natureza, seus objetivos e seus procedimentos técnicos.

No que diz respeito à natureza da pesquisa o presente trabalho se classifica em pesquisa primária, pois mesmo sendo um tópico já explorado por trabalhos citados nesse documento são aplicadas diferentes variações dos cenários de testes, como a variação dos efeitos de transformações sobre as imagens utilizadas.

Quanto aos objetivos a pesquisa se enquadra em exploratória, pois possui o objetivo de dar uma visão ampla da busca reversa de imagens, considerando técnicas específicas e alterações nas figuras.

Em relação aos procedimentos técnicos, a pesquisa se enquadra como bibliográfica e experimental. Ela é bibliográfica pois utiliza artigos e livros para fornecer a base teórica do trabalho. Também é experimental, pois são feitos experimentos durante a monografia visando avaliar critérios de acurácia e de tempo de execução.

No que diz respeito à pesquisa experimental, os experimentos são implementados na linguagem de programação Python, em sua versão 3.12.1, estando disponíveis para consulta no repositório do Github³, de maneira *online*.

Todos os experimentos utilizam do conjunto de imagens disponibilizado por Piosenka (2022), onde há 100 espécies de borboletas e mariposas. Esse conjunto foi escolhido por ser de domínio público e por fornecer imagens que diversificam o conjunto de pesquisa da busca reversa. As imagens possuem o padrão de tamanho de $224 \times 224 \times 3$, no formato JPEG (*Joint Photographic Experts Group*). Foi escolhido o conjunto com 500 figuras, de modo que, para cada imagem, foram criadas oito novas variações, cada uma com uma das transformações visuais propostas, sendo estas:

- Dessaturação (preto e branco)
- Desfoque Gaussiano (raio=2)
- Alteração de resolução (128×128)
- Realce de nitidez
- Rotação (90° em sentido anti-horário)
- Espelhamento (horizontal)
- Corte — canto inferior direito ($112 \times 112 \times 3$)

³ https://github.com/SandyHoffmann/BuscaReversaImagens_CNN-DCT-LSH

- Inserção de elemento gráfico (quadrado) em ponto aleatório da figura

As operações de desfoque gaussiano, dessaturação em preto e branco, realce de nitidez, rotação e corte foram selecionadas por representarem exemplos básicos de alterações visuais frequentemente encontradas na internet (GAILLARD, 2017). O espelhamento horizontal, por sua vez, é amplamente empregado, especialmente como técnica de aumento de dados no treinamento de redes neurais (ALOMAR; AYSEL; CAI, 2023). Já a inserção de elementos gráficos foi escolhida por ser uma prática comum em anúncios fraudulentos ou conteúdos ofensivos (SAMANTA; JAIN, 2021).

A escolha do fator de acurácia se deve pela importância do mecanismo de busca conseguir trazer as imagens similares corretamente, sendo um tópico recorrente na área da cibersegurança, onde é utilizado para reconhecer pessoas, objetos, localizações, e ver questões de veracidade (WEKESA; DECUSATIS; ZHU, A., 2023).

O desempenho é um fator importante para a busca reversa de imagens, pois o volume de dados encontrado na internet é enorme, e dependendo do algoritmo utilizado o resultado esperado pode não ser retornado a um tempo razoável para o usuário. Por isso, será observado o tempo de resposta do programa, fazendo uma média das execuções. A eficiência do algoritmo é essencial para que seu uso seja viável em conjuntos de dados em larga escala (DOUZE *et al.*, 2021).

Os experimentos são divididos em duas partes:

- **Experimento 1:** É realizada uma comparação entre a aplicação do *hashing* perceptivo baseado em DCT e o uso de CNN no contexto da RIS em termos de acurácia e desempenho de tempo. O DCT implementado utiliza do método estabelecido pela biblioteca OpenCV⁴. O modelo pré treinado escolhido para o experimento utiliza da CNN disponível no *framework* Keras⁵, sendo o modelo VGG16 (*Convolutional Network for Classification and Detection*).
- **Experimento 2:** Combinação do LSH com a CNN, afim de particionar a busca, diminuindo o espaço de pesquisa. Para a execução do experimento, é implementado o LSH aleatório, utilizando das configurações de 100 tabelas e comprimento de 8 *bits*.

1.4 REVISÃO BIBLIOGRÁFICA

No estudo de Gaillard (2017), é apresentado um método para busca reversa em grandes conjuntos de dados, no qual as redes neurais convolucionais (CNN) são empregadas para extrair características das imagens. Essas características são então compactadas por meio do *hash* perceptivo, gerando códigos binários que preservam as similaridades visuais entre as imagens. Para os experimentos, o estudo utiliza um conjunto de ima-

⁴ https://docs.opencv.org/4.x/d2/de8/group__core__array.html

⁵ <https://keras.io/api/applications/vgg/>

gens modificadas, que incluem desfoque, dessaturação para preto e branco, alterações de resolução, realce de nitidez, rotação e corte.

Em relação ao CBIR, os autores Qazanfari, AlyanNezhadi e Khoshdaregi (2023) abordam a lacuna semântica, onde características como cor e textura podem ser interpretadas de forma diferente pela máquina e pelo usuário. Assim como Gaillard (2017), destacam a capacidade da CNN de gerar representações mais robustas contra alterações comuns nas imagens, especialmente nas camadas superiores. Embora o estudo de Gaillard (2017) mostre que a CNN tem um desempenho superior ao do *hash* perceptivo, ela enfrenta desafios relacionados à dimensionalidade dos vetores de busca, exigindo a implementação de técnicas adicionais para melhorar o desempenho, especialmente em cenários com grandes volumes de dados.

Nesse contexto, Wekesa, DeCusatis e Andy Zhu (2023) e Gaillard (2017) mencionam a utilização do algoritmo LSH para agrupar dados de maneira eficiente. O LSH mapeia as características das imagens em códigos binários, facilitando a busca em grandes volumes de dados e preservando a precisão dos resultados obtidos pela CNN.

Dessa forma, os estudos revisados destacam o grande potencial da CNN e das técnicas de *hashing* para a busca reversa de imagens, ao mesmo tempo em que ressaltam a necessidade de soluções que equilibrem precisão e eficiência em cenários de grande escala.

Em relação aos trabalhos mencionados, o presente estudo apresenta dois diferenciais importantes. Primeiro, além das transformações estudadas por Gaillard (2017), foi incluída a avaliação das operações de espelhamento e inserção de elementos gráficos. Segundo, foi feita uma análise comparativa direta entre as técnicas de *hash* perceptivo, CNN e LSH, considerando a taxa de acerto e o desempenho de tempo em processamento para diferentes tamanhos de conjuntos de pesquisa. Assim, é possível ter uma visão mais detalhada sobre as vantagens e limitações de cada técnica.

2 FUNDAMENTAÇÃO TEÓRICA

Nessa seção é abordada a parte teórica da pesquisa, explicando os conceitos e aplicações da RIS (*Reverse Image Search* – Busca reversa de imagens) juntamente com os algoritmos de *hashing* perceptivo e CNN.

2.1 BUSCA REVERSA DE IMAGENS

A RIS (*Reverse Image Search* – Busca reversa de imagens) é uma técnica que possibilita identificar e relacionar imagens com base em suas características visuais. Essa abordagem acelera tarefas como a verificação de autenticidade, o monitoramento de uso indevido e a análise em larga escala de conteúdo visual (WAN; LIU, 2008). Essa técnica pertence ao campo do CBIR, possuindo o objetivo de encontrar a imagem original em uma coleção de figuras, mesmo que com pequenas alterações na imagem base (GAILLARD, 2017).

Exemplos práticos do uso da RIS incluem a detecção de casos de *copyright* em redes sociais, onde é alterado parte do conteúdo da imagem para ser repostada em outro lugar. Um exemplo desse tipo de situação pode ser visto na Figura 2a, onde uma tirinha é repostada sem as informações do autor original e com uma alteração visual. Plataformas com reconhecimento facial como PimEyes¹ e Yandex² também atuam na RIS, utilizando algoritmos para localizar imagens semelhantes ou idênticas em grandes bases de dados. Essas ferramentas são utilizadas em áreas como privacidade, segurança e detecção de uso indevido de imagens pessoais. Um exemplo prático de sua aplicação pode ser observado nas Figuras 2c (Yandex) e 2b (PimEyes), onde uma mesma foto foi inserida nos mecanismos de busca, resultando no retorno de variações da imagem da mesma pessoa.

Google Lens³ e Microsoft Bing⁴ são exemplos de ferramentas populares classificadas como RIS, sendo mecanismos de busca. Ambas aceitam uma imagem como entrada do motor de busca e retornam como resultado figuras semelhantes, conforme pode ser visto na Figura 3, onde ambos os mecanismos de busca recebem uma mesma imagem como parâmetro de busca. A Figura 3a, feita no Google Lens, mostra que como resultado há tanto a imagem do gato com a modificações no rosto, quanto variações, inclusive sem alterações visuais. Já a Figura 3b exibe a fonte da imagem e imagens de gatos com características similares à pesquisada na ferramenta Bing.

O campo da RIS apresenta diversos desafios, entre os quais se destaca a lacuna semântica, um problema que surge devido à dificuldade de transmitir à máquina as características que um ser humano percebe ao fazer a interpretação de uma imagem (QAZANFARI; ALYANNEZHADI; KHOSHDAREGI, 2023). A escalabilidade é outro fator

¹ Disponível em: <https://pimeyes.com/en>

² Disponível em: <https://yandex.com/>

³ Disponível em: <https://lens.google/>

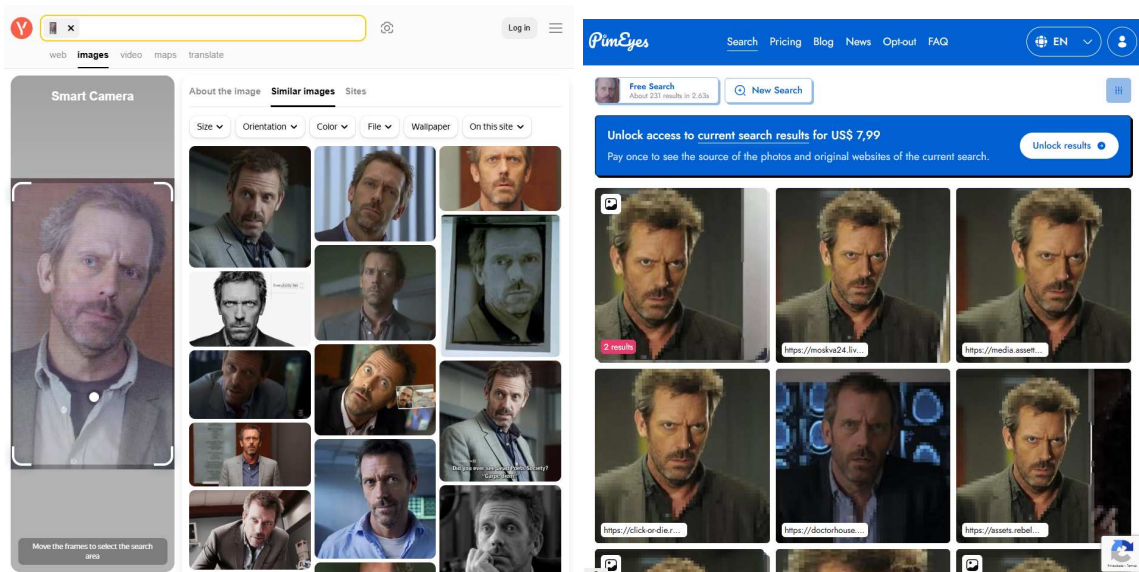
⁴ Disponível em: <https://www.bing.com/images>

Figura 2 – Exemplos práticos de RIS.



(a) Exemplo de violação de direitos autorais.

Fonte: Ohi (2020).



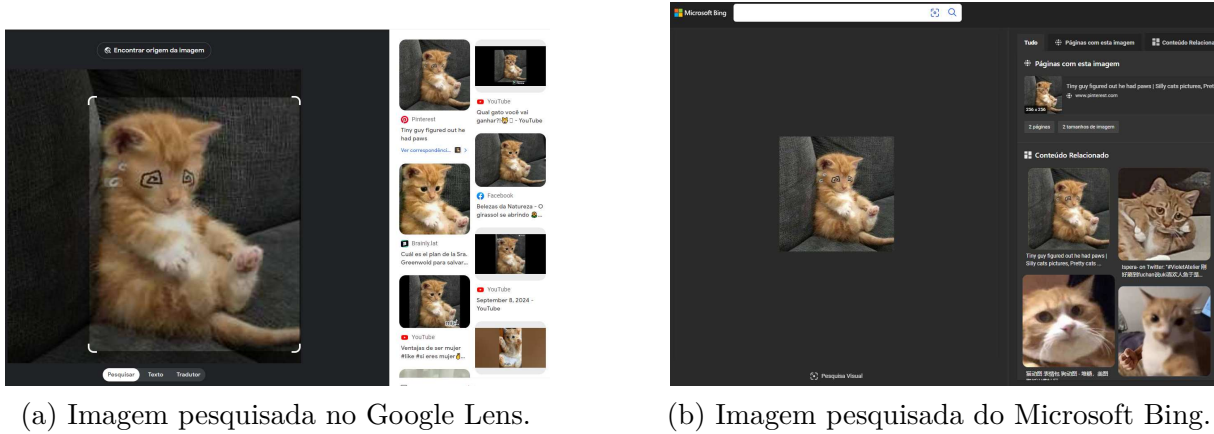
(b) Foto pesquisada no mecanismo Yandex. (c) Foto pesquisada no mecanismo PimEyes.

Fonte: Autora (2024).

Fonte: Autora (2024).

crítico para um algoritmo RIS, pois à medida que aumenta o volume de dados no espaço de pesquisa, a eficiência e a assertividade do sistema tendem a diminuir (QAZANFARI; ALYANNEZHADI; KHOSHDAREGI, 2023). Reconhecer imagens que sofreram alterações visuais, como rotação e corte, também é um desafio para o algoritmo de RIS (GAILLARD, 2017).

Figura 3 – Exemplos de mecanismos de busca utilizando imagens.



Fonte: Autora (2024).

2.2 TÉCNICAS HASHING

O uso do *hash* se encontra na computação com diferentes aplicações, como armazenamento de dados sensíveis, integridade de dados, entre outros cenários. No campo de retorno de imagens, o *hashing* é utilizado para obter maior eficiência e menor consumo de memória (CHI; ZHU, X., 2017).

A função *hash* H pode ser definida como qualquer função que possa ser usada para mapear dados de um tamanho arbitrário para um intervalo fixo $[0, m]$ (CHI; ZHU, X., 2017). Os autores Chi e Xingquan Zhu (2017) dividem o grupo de funções *hash* em duas classificações principais, sendo estas: *hashing* orientado a dados, onde o objetivo principal é usar o *hash* para aumentar a velocidade do processo de retorno de dados e o *hashing* orientado a segurança, onde o *hash* é utilizado para gerar assinaturas e garantir integridade de dados.

No caso do *hashing* perceptivo e do *hashing* sensível à localidade, ambos são classificados como *hashing* orientados a dados (CHI; ZHU, X., 2017) (WENG; JHUO; CHENG, 2019). Assim, ambos possuem o objetivo de alterar a representação dos dados, de maneira a diminuir sua dimensionalidade. As verificações de busca por *hash* são geralmente mais simples, sendo assim mais rápidas (WENG; JHUO; CHENG, 2019), ajudando a melhorar o desafio da escalabilidade, quando aplicado ao campo da RIS.

2.2.1 Hashing Perceptivo

O *hashing* perceptivo possui o objetivo de gerar uma representação *hash* mantendo valores próximos, mesmo se sua entrada for exposta a pequenas mudanças (DU; HO; CONG, 2020). Assim, é possível utilizar esse tipo de *hash* para fazer comparações entre as representações geradas medindo a similaridade entre dois *hashes* distintos (GAILLARD, 2017).

De acordo com Meixner e Uhl (2006) há quatro propriedades desejáveis para o *perceptual hash*. Considerando P como a probabilidade, Hp como a função *hash*, X a imagem alvo, X' sendo semelhante a X , Y como uma imagem visualmente distinta de X , α e β sendo valores de *hash*, e $\{0, 1\}^L$ como strings binárias de tamanho L .

- Robustez perceptual:

$$P(Hp(X) = \alpha) \approx \frac{1}{2^L}, \forall \alpha \in \{0, 1\}^L. \quad I$$

- Independência de par entre valores de duas figuras perceptualmente diferentes:

$$P(Hp(X) = \alpha | Hp(Y) = \beta) \approx P(Hp(X) = \alpha). \quad II$$

É importante notar que $P(A|B)$ é a operação de probabilidade condicional, dada por $\frac{P(A \cap B)}{P(B)}$.

- Invariância para imagens visualmente semelhantes:

$$P(Hp(X) = Hp(X')) \approx 1. \quad III$$

- Distinção para imagens visualmente diferentes:

$$P(Hp(X) = Hp(Y)) \approx 0. \quad IV$$

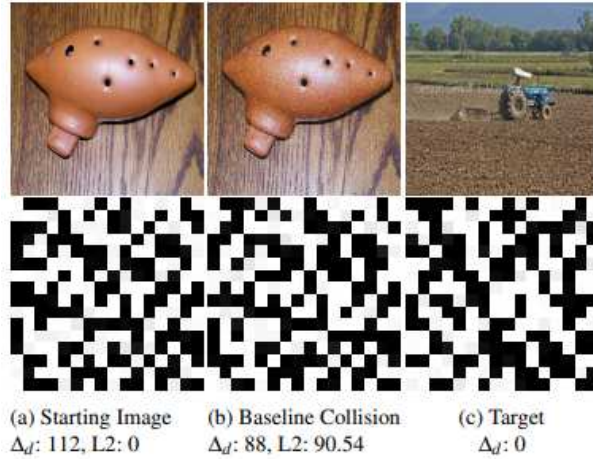
A Propriedade *I* diz que os valores de *hash* calculados por Hp devem ter chances iguais de serem gerados, ou seja, deve-se ter uma distribuição uniforme no espaço de *hash*. Isto é importante para não haver colisões frequentes. Conhecer o valor *hash* de uma entrada Y não deve fornecer informações sobre o *hash* que será formado por X , conforme afirmado na Propriedade *II*.

A Propriedade *II* é importante nas questões de segurança e robustez do algoritmo, pois não se deve conseguir manipular o *hash* de imagens distintas a fim de fazê-las coincidirem. Um exemplo desse tipo de ataque pode ser visto na Figura 4, onde é aplicada a técnica de geração de segunda pré-imagem direcionada. Na figura, Δ_d representa a diferença no espaço *hash* e L_2 remete à distância perceptiva entre as imagens. Na Figura 4a, há a imagem original, que serve como ponto de partida. Já na Figura 4b, é apresentada uma colisão gerada pela técnica, onde o *hash* perceptivo foi manipulado para coincidir com o da imagem alvo, mesmo havendo diferenças visuais.

Na Figura 4c, é exibida a imagem alvo, cuja representação no espaço *hash* é idêntica a da imagem original ($\Delta_d = 0$), mas distinta em termos visuais. Essa técnica tenta gerar uma imagem que colide no espaço *hash* permanecendo perceptualmente distante da original (PROKOS *et al.*, 2023), ou seja, gera códigos *hashes* iguais para imagens com conteúdos visuais diferentes. Tal abordagem pode causar falsos positivos em sistemas que utilizam o *hash* perceptivo para detectar conteúdos ilícitos, comprometendo sua confiabilidade.

As Propriedades *III* e *IV* são peças-chave para o campo da busca reversa de imagens, ao ser desejado que entradas similares resultem em valores de *hash* com proximidade de representação, enquanto entradas diferentes tenham representações distintas.

Figura 4 – Exemplo de ataque de segunda pré-imagem direcionada.



Fonte: Prokos *et al.* (2023).

2.2.1.1 DCT (Discrete Cosine Transformation – Transformada Discreta do Cosseno)

O DCT (*Discrete Cosine Transformation* – Transformada Discreta do Cosseno) é um algoritmo de *hashing* perceptivo que utiliza as funções do cosseno para somar senoides com frequências e amplitudes diferentes (ZAUNER, 2010). Esta função é derivada da DFT (*Discrete Fourier Transformation* – Transformada Discreta de Fourier), cujo objetivo é mapear dados do domínio de tempo para o domínio de frequência (STRANG, 1999). Seu uso se aplica no cenário do *perceptual hash* pela propriedade de estabilidade a modificações nas imagens que os seus coeficientes de baixa frequência provêm (GAILLARD, 2017). De acordo com Zauner (2010) a variação mais comum do DCT é o tipo II.

Para formar a matriz DCT c considera-se N como seu tamanho, n como índice da linha e m como índice da coluna, obtendo a Equação (1) (ZAUNER, 2010).

$$c[n, m] = \sqrt{\frac{2}{N}} \cdot \cos\left(\frac{(2m + 1) \cdot n\pi}{2N}\right). \quad (1)$$

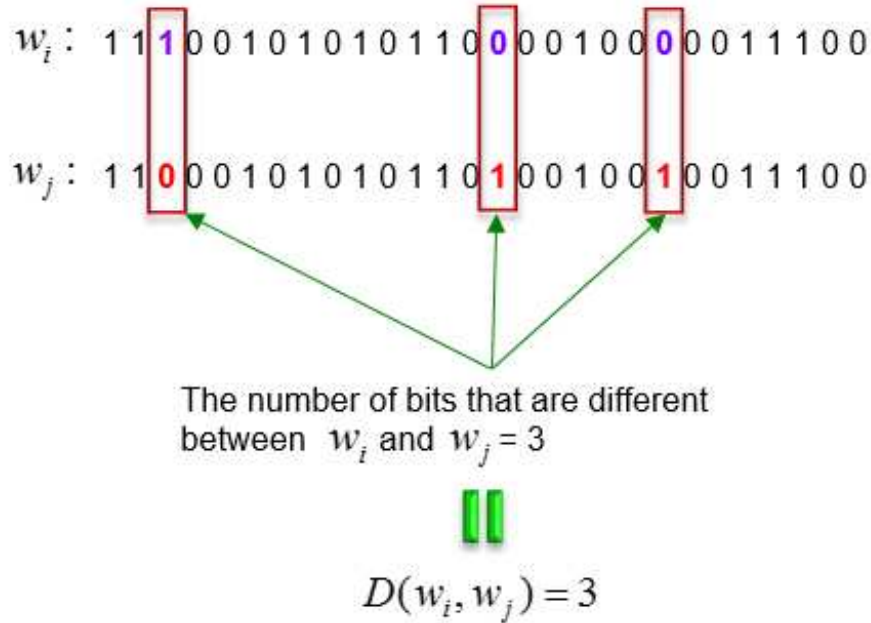
2.2.1.2 Distância de Hamming

O cálculo de distância no campo do *hashing* perceptivo pode ser feito com a fórmula de Hamming, mostrada na Equação (2), onde a função de Hamming dh opera sobre duas palavras binárias a e b , ambas possuindo o mesmo comprimento n , sendo a distância computada como o somatório de i até n utilizando o operador binário XOR (*Exclusive or* – Ou Exclusivo). Assim, a distância é definida como a quantidade de *bits* iguais a um obtida a partir da operação XOR entre duas palavras binárias de mesmo comprimento.

$$d_h(a, b) = \sum_{i=1}^n (a_i \oplus b_i). \quad (2)$$

Na Figura 5, é ilustrada a aplicação da distância de Hamming, comparando duas palavras binárias *bit a bit* e contabilizando as posições em que os *bits* diferem. A soma dessas diferenças determina a distância de Hamming entre as duas palavras.

Figura 5 – Exemplo de aplicação da distância de Hamming.



Fonte: Yan (2024).

2.2.1.3 Uso do Hashing Perceptivo no campo da RIS

A aplicação do *hashing* perceptivo no campo da RIS ocorre criando uma nova representação *hash* da imagem, e comparando com os *hashes* disponíveis das demais figuras do espaço de busca. Assim, é possível utilizar o cálculo da distância de Hamming (visto na Seção 2.2.1.2).

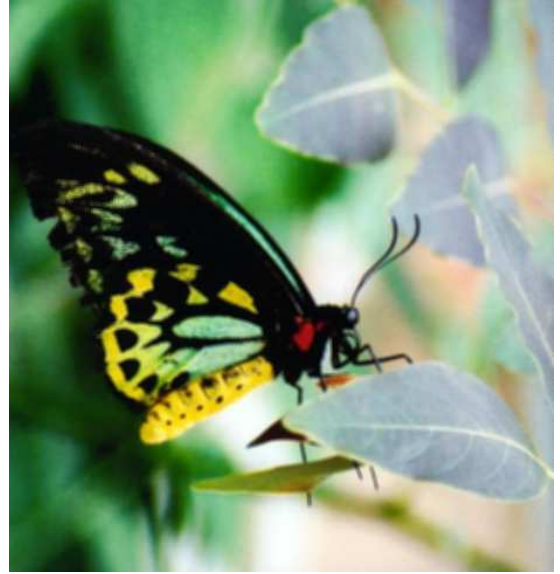
A Figura 6 ilustra esse processo, mostrando a geração de *hash* perceptivo DCT da mesma imagem com quatro variações de efeitos visuais. Assim, entradas similares são mapeados para códigos binários próximos, fornecendo representações mais compactas e mais fáceis de serem computadas pelo computador (GAILLARD, 2017).

A Tabela 1 demonstra como os códigos *hash* computados na Figura 6 seriam calculados, se considerassem a distância de cada uma das quatro variações com a imagem sem modificações. Na busca, quando deparada com a própria imagem, a distância seria 0. Para imagens alteradas com desfoque Gaussiano ou alteração de resolução, a distância seria 6. A imagem com o espelhamento horizontal gerou um código *hash* bem distante da imagem sem modificações, com uma distância de 62 pontos. O *hashing* perceptivo pode não ser robusto quando exposto a algumas transformações nas imagens, como o corte e a rotação (GAILLARD, 2017).

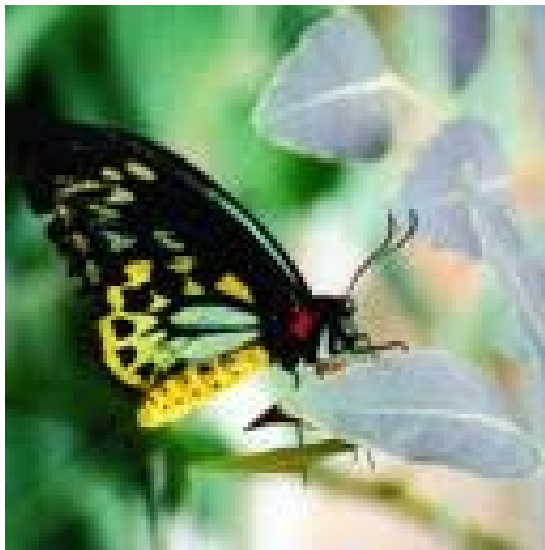
Figura 6 – Exemplos de geração de DCT em imagens modificadas.



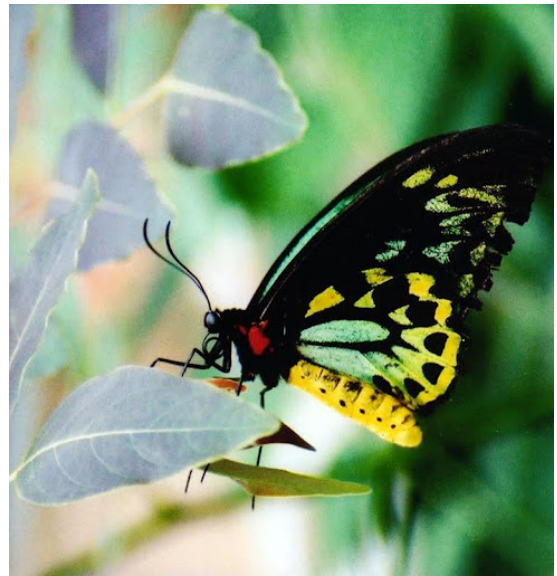
(a) Imagem sem alterações visuais:
0xc60a6c04cc1a7bcbfc13e33f119e400.



(b) Imagem com desfoque Gaussiano:
0xc6086804cc5a7b8bcfc43e33f119e400.



(c) Imagem com alteração de resolução:
0xc6086804cc5a7b8bcfc43e33f119e400.



(d) Imagem com espelhamento horizontal:
0x934f3979990b2e9e9ab36b66a44cb10d.

Fonte: Autora (2024).

2.2.2 *Hashing* Sensível à Localidade

O LSH (*Locality Sensitive Hashing* – *Hashing* sensível à Localidade) é um método baseado em *hashing* que tem a finalidade de encontrar os vizinhos próximos de um conjunto de dados com dimensões variadas projetando seus dados em um ou mais espaços com baixa dimensionalidade utilizando as funções *hash* (JAFARI *et al.*, 2021). As funções tradicionais de achar os vizinhos próximos em espaços com alta dimensionalidade possuem alto custo computacional, e o LSH possui o intuito de diminuí-lo (RAJARAM; SCHOLZ,

Tabela 1 – Distâncias de Hamming entre as imagens da Figura 6.

Imagem	Distância para a imagem original
Inalterada	0
Desfocada	6
Redimensionada	6
Espelhada	62

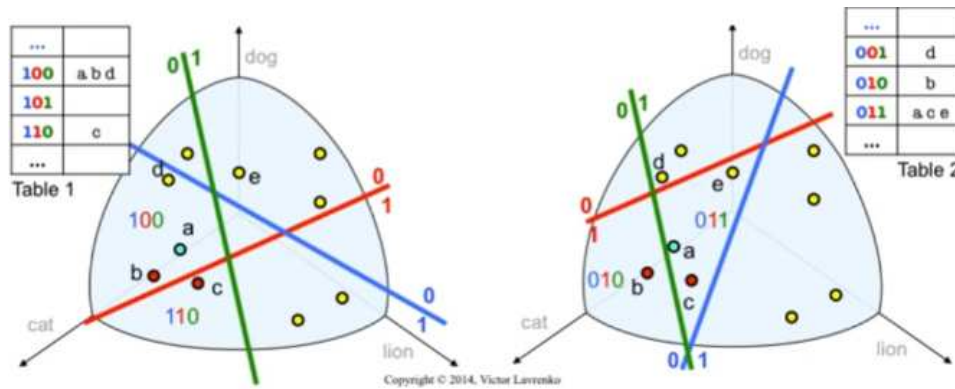
Fonte: Autora (2024).

2008).

Para isso é feito um particionamento dos pontos em grupos, ou *buckets*, partindo do princípio que há uma alta probabilidade de pontos próximos serem mapeados a um mesmo *bucket*, e pontos distantes serem mapeados para *buckets* diferentes. O LSH baseado em ângulo consiste na divisão do seu espaço de dados em projeções aleatórias múltiplas ajustando seu ângulo de maneira ortogonal.

A Figura 7 ilustra esse processo, onde os pontos são os dados do conjunto de busca e as linhas são os hiperplanos aleatórios. É exemplificada a criação de duas tabelas *hash* nesse modelo e, dependendo da aleatoriedade dos hiperplanos gerados, os pontos podem pertencer a um mesmo grupo (gerando o mesmo *hash*) ou não, assim limitando o espaço de busca final.

Figura 7 – Representação do particionamento dos dados utilizando LSH aleatório.



Fonte: Lavrenko (2015).

A probabilidade P de dois vetores A e B colidirem⁵, ao passarem pela função *hash* h , se dá pela Fórmula (3) (CHARIKAR, 2002), se baseando no ângulo θ entre eles. Vetores com ângulos menores entre si apresentam uma maior probabilidade de colisão.

$$P[h(A) = h(B)] = 1 - \frac{\theta}{\pi}, \text{ onde } \theta = \arccos \left(\frac{|A \cap B|}{\sqrt{|A| \cdot |B|}} \right). \quad (3)$$

⁵ Uma colisão em uma função *hash* ocorre quando duas entradas distintas x_1 e x_2 geram o mesmo valor de hash, ou seja, $x_1 \neq x_2$ e $h(x_1) = h(x_2)$ (KOMARGODSKI; NAOR; YOGEV, 2018).

2.2.2.1 Uso do Hashing Sensível à Localidade no campo da RIS

O uso do LSH no campo da RIS é geralmente visto em conjunto com a CNN (apresentada na seção a seguir), de modo a particionar o espaço de busca de seus vetores. Assim, seu uso é recomendado para atingir melhorias de acurácia e eficiência na busca (SCHIAVO *et al.*, 2021).

O LSH transforma os vetores de característica fornecidos pela CNN e utiliza as projeções aleatórias para converter esses valores para o formato *hash* e colocar em tabelas *hash* que armazenam vetores que compartilham o mesmo código *hash*. Antes da busca reversa por imagens ser feita, o *hash* gerado a partir da imagem alvo é comparado com os códigos disponíveis nas tabelas, sendo procurados vetores correspondentes, e assim diminuindo o espaço de busca. Após isso, é feito o cálculo com base na similaridade do cosseno, retornando as instâncias mais similares.

A similaridade do cosseno é uma medida que estima a similaridade entre dois vetores, calculando o cosseno do ângulo θ entre eles. Essa medida é dada pelo produto escalar de x e y , dividido pelo produto de suas normas L2, conforme apresentado na Equação (4) (GAILLARD, 2017).

$$\cos \theta(x, y) = \frac{x \cdot y}{\|x\|_2 \|y\|_2}. \quad (4)$$

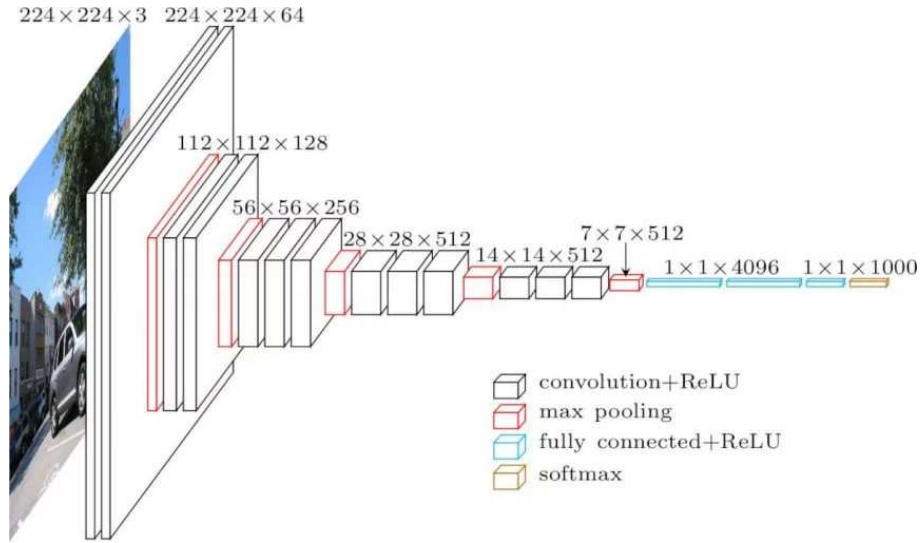
2.3 REDE NEURAL CONVOLUCIONAL

A CNN (*Convolutional Neural Network* – Rede Neural Convolutacional) é um tipo de inteligência artificial responsável por processar dados que possuem uma topologia conhecida de grade (GOODFELLOW; BENGIO; COURVILLE, 2016), sendo vastamente utilizada como arquitetura no campo de processamento de imagens (WEIDMAN, 2019). Segundo Goodfellow, Bengio e Courville (2016), o conceito de CNN pode ser dado por uma rede neural que utiliza convolução ao invés da matriz geral de multiplicação em no mínimo uma camada.

2.3.1 Arquitetura

Na Figura 8, pode ser vista a arquitetura do modelo VGG16 (SIMONYAN; ZISSERMAN, 2015), sendo uma CNN projetada para tarefas de classificação e detecção de imagens. São vistas quatro classes de camadas, sendo elas: convolucionais, *max pooling*, camadas totalmente conectadas (ou densas), e as funções de ativação *softmax* e ReLU. Ao longo desta seção, serão explorados os conceitos dessas camadas e sua influência no desempenho e comportamento da CNN.

Figura 8 – Organização das camadas da rede VGG16.



Fonte: Hassan (2018).

2.3.1.1 Camadas de Convolução

A CNN utiliza da operação de convolução, sendo denominada uma operação matemática entre duas funções com valores reais (GOODFELLOW; BENGIO; COURVILLE, 2016). De acordo com Goodfellow, Bengio e Courville (2016), muitas redes neurais implementam a função de correlação cruzada sendo a mesma que a convolução, porém sem inverter o *kernel*. Considerando K como um *kernel*⁶ bi-dimensional, I sendo a matriz representativa da imagem e S sendo a nova função resultante, tem-se então a equação geral da convolução na CNN como (GOODFELLOW; BENGIO; COURVILLE, 2016):

$$S(i, j) = (K * I)(i, j) = \sum_m \sum_n I(i + m, j + n) K(m, n). \quad (5)$$

Um exemplo simples de aplicação de convolução em imagens é o *box blur*, operação cujo objetivo é aplicar um filtro para suavizar o resultado da figura. Esse filtro utiliza um *kernel* que permite fazer a média dos valores selecionados por meio da convolução. Esse processo pode ser observado na Equação (6), onde é aplicado um *kernel* 3x3 a um conjunto de pixels de uma determinada imagem retornando a média geral da soma dos termos.

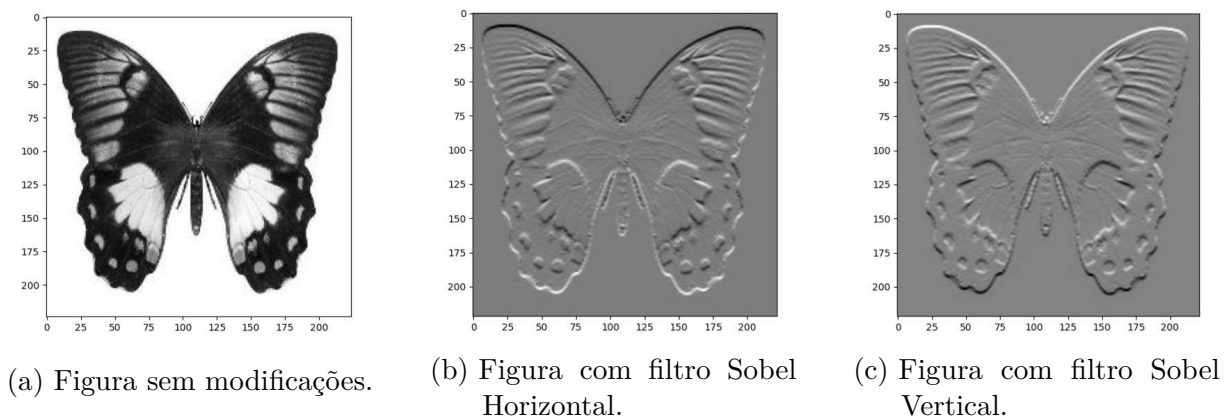
$$\begin{bmatrix} 1 & 1 & 1 \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ 1 & 1 & 1 \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ 1 & 1 & 1 \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix} * \begin{bmatrix} 2 & 2 & 2 \\ 4 & 4 & 4 \\ 6 & 6 & 6 \end{bmatrix} = \left[\frac{1}{9} \cdot 2 + \frac{1}{9} \cdot 2 + \dots + \frac{1}{9} \cdot 6 \right] = 4. \quad (6)$$

⁶ *Kernel*, no contexto da convolução matemática, é uma matriz utilizada como filtro para modificar a matriz de entrada.

Para uma camada convolutiva, geralmente são necessários múltiplos *kernels* (WU, 2017). Isso ocorre porque cada *kernel* pode destacar diferentes características da imagem, como bordas horizontais, verticais ou em outras direções. Como exemplo, pode-se utilizar os filtros de Sobel para realçar as bordas, conforme demonstrado na Figura 9. É possível na CNN utilizar essa qualidade da convolução para aprender características que ativam as bordas para ângulos diferentes, assim, as próximas camadas podem ser ativadas por contornos que tenham significados específicos (WU, 2017).

A aplicação da convolução na CNN melhora o processo de aprendizado, promovendo iterações esparsas, com o armazenamento de menos parâmetros nos *kernels* (GOODFELLOW; BENGIO; COURVILLE, 2016). Também melhora o compartilhamento de parâmetros, utilizando a mesma configuração para mais de uma função no modelo, e representações equivariantes, sendo que se a entrada for deslocada, a saída também será, de maneira correspondente (GOODFELLOW; BENGIO; COURVILLE, 2016).

Figura 9 – Efeito do filtro de Sobel sobre as imagens.



Fonte: Autora (2024).

2.3.1.2 Camadas de Pooling

As funções *Pooling*, ou de agrupamento, formam uma camada geralmente utilizada na CNN, visando reduzir a representação dos mapas de característica utilizando uma convolução (WEIDMAN, 2019). Para isso, a função *Pooling* substitui a saída da rede com uma estatística resumida das saídas próximas (GOODFELLOW; BENGIO; COURVILLE, 2016).

Seu uso auxilia a rede a aumentar sua invariância a pequenas translações (GOODFELLOW; BENGIO; COURVILLE, 2016), permitindo que o modelo mantenha sua acurácia independente da localização exata dos objetos na imagem. Seu uso auxilia também a redução dos números de parâmetros da rede e otimiza o número de computações necessárias para treinar o modelo (WEIDMAN, 2019).

Um tipo de função *Pooling* é a *max-pooling*, onde é extraído o maior valor de saída de uma seleção de vizinhos próximos (GOODFELLOW; BENGIO; COURVILLE, 2016). Outra função utilizada é o *average-pooling*, ou *pooling* médio, onde é feito a média dos valores daquela seleção escolhida (WEIDMAN, 2019).

2.3.1.3 Camadas de Ativação

As funções de ativação são utilizadas para satisfazer alguma especificidade do problema que o neurônio está tentando resolver (DEMUTH *et al.*, 2014). Geralmente, as funções utilizadas na maioria das redes neurais são não lineares, e suas saídas são chamadas de ativações (WEIDMAN, 2019).

2.3.1.3.1 Softmax

A função Softmax, ou função de normalização exponencial, é um tipo de função de ativação utilizada para prevenir o *Overflow* e o *Underflow* nas saídas das camadas da rede (GOODFELLOW; BENGIO; COURVILLE, 2016). Ambos são problemas de arredondamento numérico, onde o *Underflow* ocorre quando números próximos a zero são arredondados para zero e o *Overflow* ocorre quando os números se aproximam do ∞ ou $-\infty$ (GOODFELLOW; BENGIO; COURVILLE, 2016). De acordo com Goodfellow, Bengio e Courville (2016), considerando x_i como a entrada associada à i -ésima classe e n como o número total de classes, tem-se a fórmula do Softmax definida pela Equação (7).

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_{j=1}^n \exp(x_j)}. \quad (7)$$

As saídas da função Softmax podem ser consideradas um vetor de probabilidades ligadas a cada classe de um problema, onde cada saída varia de 0 a 1, e a soma das saídas é igual a 1 (DEMUTH *et al.*, 2014).

2.3.1.3.2 ReLU

A ReLU (*Rectified Linear Unit* – Unidade Linear Retificada) é uma função de ativação que possui o objetivo de aumentar a não linearidade em casos de redes CNN (WU, 2017). Considerando y como o valor de saída da função ReLU, x o valor de entrada, i , j e d como índices que representam a posição no mapa de ativação, a fórmula da ReLU é representada pela Equação (8) (WU, 2017). Os índices i e j correspondem, respectivamente, à linha e à coluna, enquanto d se refere à profundidade do mapa de ativação.

$$y_{i,j,d} = \max\{0, x_{i,j,d}\}. \quad (8)$$

Seu uso é popular em arquiteturas de redes neurais densas por produzir gradientes grandes (WEIDMAN, 2019), facilitando o treinamento das redes e permitindo uma atualização mais eficiente dos pesos.

2.3.1.4 Camadas Densas

Na camada densa, também chamada de camada totalmente conectada, todos os neurônios de entrada contribuem para o cálculo de cada neurônio de saída (WEIDMAN, 2019). Essa característica é interessante para a CNN, especialmente nas camadas finais, pois se deseja combinar todas as características aprendidas anteriormente para construir representações mais fortes (WU, 2017).

No caso da CNN, partindo de uma camada convolutiva, sua saída será um vetor tridimensional, onde há o formato de m canais de cores \times largura da imagem \times altura da imagem. Logo, há a necessidade de utilizar a operação de achatamento (*flatten*) para que o mapa de característica tenha apenas uma dimensão e seja possível passar a camada totalmente conectada para calcular as probabilidades (no caso de classificação) (WEIDMAN, 2019).

2.3.2 Treinamento

O treinamento de uma rede visa adaptar parâmetros internos do modelo de modo que ele tenha um bom desempenho em futuros dados não vistos (DEISENROTH; FAISAL; ONG, 2020). Para isso, é utilizado um conjunto de dados como entrada do modelo, que possui uma função que avalia o quão bem a rede prevê os dados de treinamento (DEISENROTH; FAISAL; ONG, 2020). Entre os parâmetros alterados estão os pesos, que precisam ser configurados corretamente para a rede funcionar (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.3.2.1 Preparação dos Dados

Alcançar uma boa performance, ou seja, alta capacidade de generalização em novas entradas, é um dos principais desafios na área de aprendizado de máquina (GOODFELLOW; BENGIO; COURVILLE, 2016). De acordo com Goodfellow, Bengio e Courville (2016), para avaliar o desempenho de um modelo, é comum calcular duas métricas de erro: o erro de treinamento, que trata do desempenho do modelo nos dados utilizados no treinamento, e o erro de teste, que mede sua capacidade de generalização para dados inéditos. O objetivo é que ambos os valores sejam baixos (GOODFELLOW; BENGIO; COURVILLE, 2016), assim indicando que o modelo aprendeu os padrões sem ter se ajustado apenas à configuração dos dados de treinamento.

O conjunto de dados é geralmente dividido em três partes, 70% como treinamento, 15% validação e 15% teste (DEMUTH *et al.*, 2014). De acordo com Demuth *et al.* (2014),

o conjunto de treinamento é formado por exemplos usados para ajustar os pesos da rede durante cada iteração. O conjunto de validação, por sua vez, monitora o desempenho do modelo ao longo do treinamento, decidindo quando o treinamento deve ser parado. Já o conjunto de teste é utilizado como indicativo de como a rede irá se comportar no futuro, avaliando sua capacidade de generalização.

Uma das práticas implementadas no campo de aprendizado de máquina para aumentar a generalização do modelo é o aumento de dados, ou *augmentation*, que cria dados falsos para adicionar ao conjunto de treinamento (GOODFELLOW; BENGIO; COURVILLE, 2016). Em caso de imagens, as variações podem ser facilmente simuladas, incluindo rotações ou mudanças de tamanho (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.3.2.2 Propagação Direta e Propagação Reversa

O processo de propagação direta, ou *forward pass*, é dado quando uma rede neural aceita uma entrada x e produz uma saída y de modo que a informação é passada diretamente pela rede, até produzir um custo escalar $J(\theta)$ (GOODFELLOW; BENGIO; COURVILLE, 2016). O processo de propagação reversa, ou *backpropagation*, permite que as informações do custo voltem à rede para computar o gradiente (GOODFELLOW; BENGIO; COURVILLE, 2016). O algoritmo propagação reversa permite o ajuste dos pesos em redes neurais, utilizando o método de descida do gradiente (MITCHELL; MITCHELL, 1997). A descida do gradiente é o processo de minimizar funções, de modo a mover os parâmetros em pequenos passos na direção oposta ao sinal da derivada de maneira gradativa, até alcançar um valor de mínimo (GOODFELLOW; BENGIO; COURVILLE, 2016).

Um método de descida de gradiente comumente utilizado é o algoritmo SGD (*Stochastic Gradient Descent* – Descida Estocástica do Gradiente), que ao invés de calcular o gradiente exato em relação a todo o conjunto dos dados de treinamento, utiliza o conceito de *mini-batch*, que seria um sub-conjunto menor de dados de treinamento (GOODFELLOW; BENGIO; COURVILLE, 2016). Utilizar grandes conjuntos de dados para alimentar a rede é muito custoso computacionalmente, por isso essa técnica é utilizada para estimar o gradiente e assim utilizar apenas parte do conjunto de dados a cada passo (MITCHELL; MITCHELL, 1997).

O cálculo do gradiente pode ser aprimorado com o uso do *Momentum*, que possui o objetivo de acelerar o aprendizado, introduzindo uma variável v de velocidade (GOODFELLOW; BENGIO; COURVILLE, 2016). Essa variável representa a direção e a rapidez que os parâmetros são ajustados no espaço de parâmetros, refinando a atualização deles (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.3.3 Uso da CNN no campo da RIS

A aplicação da CNN no campo da RIS se dá pela extração dos vetores de parâmetros de uma das camadas da rede. Assim, são construídas novas formas de representação para as figuras, onde pode ser calculada a proximidade por métricas de distância, como a distância Euclidiana.

Sua aplicação provê robustez na RIS, mostrando resultados promissores contra alterações visuais na figura, como cortes e mudanças de resolução (GAILLARD, 2017). Essa resiliência ocorre por conta das fases da preparação de dados descritas na Seção 2.3.2.1, o que inclui processos como o pré-processamento e o aumento de dados, submetendo as imagens a diversas transformações visuais e forçando o modelo a aprender a lidar com esse tipo de representação (GAILLARD, 2017). O compartilhamento de parâmetros entre as camadas também auxilia a rede a ter uma maior invariação à translação, fazendo com que a mesma característica seja computada em diferentes locais da entrada (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.3.3.1 Distância Euclidiana

A distância Euclidiana é a distância em linha reta entre dois pontos no espaço (GAILLARD, 2017). Considerando a distância Euclidiana de dois pontos x e y , n como o número de dimensões no espaço e i como índice, tem-se a fórmula da distância Euclidiana na Equação (9).

$$d_{\text{Euclidiana}}(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}. \quad (9)$$

3 EXPERIMENTOS

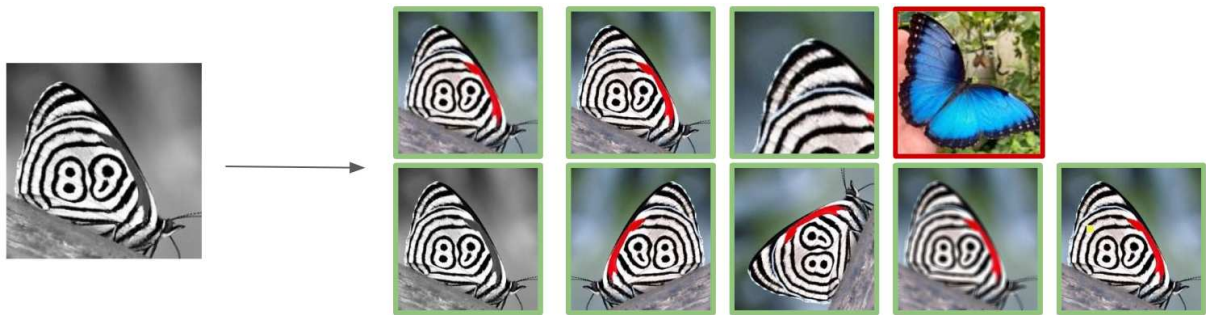
Esta seção apresenta os experimentos propostos e especificados na Seção 1.3, detalhando as métricas selecionadas, a máquina utilizada, e a quantidade de testes realizados. Também estão explicadas as implementações realizadas, bem como são apresentados os resultados obtidos acompanhados de uma análise.

No que diz respeito às métricas avaliadas em ambos os experimentos, foram utilizadas a **acurácia** e o **desempenho** em termos de tempo.

A forma de avaliação da taxa de acertos segue o seguinte princípio: para cada imagem consultada, espera-se o retorno das suas nove variações, com as oito alterações visuais possíveis mais a sua versão sem modificações. Para cada retorno correto conta-se um acerto; caso contrário, conta-se como erro. O espaço de busca foi composto por 4500 imagens, após aplicar as transformações visuais propostas (as 500 imagens originais acrescidas das suas oito variações).

Esse processo é ilustrado na Figura 10, sendo selecionada para o mecanismo de busca uma imagem em preto e branco, e retornando os nove resultados considerados mais similares. Os resultados com contorno verde são variações da imagem pesquisada, sendo contados como acertos, enquanto a figura com a borda vermelha corresponde a outro tipo de borboleta, sendo assim considerada um erro.

Figura 10 – Forma de avaliação de acurácia.



Fonte: Autora (2024).

Para o critério de desempenho de tempo foi considerado o tempo de execução dos programas. São seguidas as mesmas configurações de parâmetros do experimento de acurácia, considerando imagens submetidas às transformações visuais propostas. O experimento é realizado retratando o desempenho do programa quando busca por 100 imagens de cada transformação (900 no total), 300 imagens de cada transformação (2700 no total) e 500 imagens de cada transformação (4500 no total). Foram utilizadas diferentes quantidades de imagens para permitir a análise da escalabilidade das tecnologias de busca, verificando como o tempo de execução se comporta conforme o volume de dados aumenta. Os experimentos foram realizados em um computador com as especificações descritas na Tabela 2.

Tabela 2 – Configurações do Computador.

Especificação	Detalhes
Processador	Intel(R) Core(TM) i5-4440 CPU @ 3.10GHz
Placa de Vídeo	NVIDIA GeForce GTX 970 (4 GB GDDR5)
Memória RAM	8 GB DDR3
Sistema Operacional	Microsoft Windows 10 Professional (x64)

Fonte: Autora (2024).

Os experimentos são realizados em três execuções independentes, registrando os resultados de acurácia e tempo de execução. Os valores finais são obtidos por meio da média aritmética simples dos resultados obtidos em cada repetição.

3.1 EXPERIMENTO 1

Para o experimento 1 tem-se a comparação direta dos resultados alcançados utilizando a técnica DCT (vista na Seção 2.2.1.1) com o uso da CNN (vista na Seção 2.3).

3.1.1 Implementação da Técnica RIS utilizando DCT

O processo de RIS utilizando o DCT começa com o tratamento das figuras, convertendo-as para preto e branco. A conversão de cor é feita para reduzir a quantidade de informações, utilizando apenas a intensidade dos píxeis.

Após isso, a figura passa por uma operação de desfoque, visando reduzir o ruído. O cálculo do DCT é feito utilizando a biblioteca OpenCV, retornando a matriz de coeficientes.

Para formar o código *hash* são considerados os primeiros coeficientes da matriz DCT (no formato 8×8). Esses coeficientes, que representam as componentes de baixa frequência, são escolhidos por serem mais estáveis diante de modificações nas imagens (GAILLARD, 2017).

Assim, para formar o *hash* é percorrida a matriz de coeficientes e estabelecida a seguinte condição: se o valor for maior ou igual à mediana dos valores da matriz DCT, o *bit* mais a direita é definido como 1, caso contrário permanece 0.

Para determinar a distância entre os códigos gerados é utilizada a distância de *Hamming*, vista na Seção 2.2.1.2.

3.1.2 Implementação da Técnica RIS utilizando CNN

Para a aplicação no campo da RIS, é necessário alimentar a rede com o banco de imagens selecionado para a busca e extrair as novas representações das figuras. Para a CNN, foi escolhido o modelo VGG16, sendo uma rede pré-treinada aplicada em problemas de classificação de imagens, extração de características para outros conjuntos de dados, localização de objetos e transferência de aprendizado (SIMONYAN; ZISSERMAN, 2015).

É configurado para que o modelo VGG16 forneça como saída da rede os valores de parâmetros estabelecidos na camada *fc1*, sendo a penúltima camada da rede e possuindo 4096 neurônios. Essa camada foi selecionada por pertencer às camadas superiores da rede, conhecidas por gerar ativações capazes de capturar descritores de alto nível do conteúdo visual da imagem (GAILLARD, 2017).

Dessa forma, o vetor de características de cada imagem do conjunto de dados é extraído e salvo em arquivos no formato da biblioteca NumPy¹, escolhida por sua eficiência no armazenamento e carregamento desse tipo de dado.

Para calcular a similaridade entre as entradas, é utilizado o cálculo da distância Euclidiana, vista na Seção 2.3.3.1.

3.1.3 Análise dos Resultados do Experimento 1

Os resultados apresentados na Tabela 3 indicam que a técnica DCT obteve melhores taxas de acerto nos casos de desfoque gaussiano e dessaturação, obtendo com uma diferença de, respectivamente, 4,02% e 6,85%, se comparada à CNN. A CNN apresentou um aumento significativo na acurácia em sete das nove transformações avaliadas. Os resultados incluem um incremento de 22,51% nos casos de busca por imagens com o método original, 21,58% para mudanças de tamanho, 25,56% em imagens com realce de nitidez e 21,60% em casos onde elementos gráficos foram adicionados.

As operações de corte, rotação e espelhamento resultaram em ganhos significativos na acurácia da CNN. Comparada à técnica de DCT, a acurácia aumentou de 13,55% para 27,62% com o corte, de 13,77% para 69,15% com a rotação, e de 15,13% para 92,42% com o espelhamento. Esses resultados são consistentes com os obtidos no trabalho de Gaillard (2017), evidenciando a robustez da CNN a operações como rotação e corte. Além disso, conforme destacado por Qazanfari, AlyanNezhadi e Khoshdaregi (2023), a CNN demonstra maior resiliência a operações comuns de manipulação de imagens. É adicionada a essas observações a alta taxa de acertos da CNN, se comparada ao DCT, nas operações de espelhamento e adição de elementos gráficos, que constituem as transformações adicionais analisadas neste estudo.

No estudo de desempenho de tempo, conforme apresentado na Tabela 4, é demonstrado a média de tempo de execução dos algoritmos DCT e CNN conforme aumento na quantidade de imagens de busca. Pode-se constatar que a técnica DCT obteve tempos de execução mais rápidos que a CNN, reduzindo o tempo de execução conforme a quantidade de imagens aumenta, chegando a ter uma taxa de diminuição de 78,49% na pesquisa por 4500 imagens. Isso pode ser explicado devido às vantagens que o *hash* possui como forma de representação, tanto em critérios de tamanho, quanto de cálculo de distância (GAILLARD, 2017).

¹ <https://numpy.org/>

Tabela 3 – Porcentagem de acertos obtidos com as técnicas DCT e CNN (Total de 4500 imagens).

Transformação	DCT - Taxa de Acertos	CNN - Taxa de Acertos
Imagem Original	67,55%	90,06%
Desfoque Gaussiano	67,44%	63,42%
Dessaturação	67,73%	60,88%
Alteração de Resolução	67,53%	89,11%
Realce de Nitidez	67,48%	93,04%
Rotação (90°)	13,77%	69,15%
Espelhamento Horizontal	15,13%	92,42%
Corte	13,55%	27,62%
Inserção de Elemento Gráfico	67,37%	88,97%

Fonte: Autora (2024).

Tabela 4 – Média de tempo de execução obtida com as técnicas DCT e CNN.

Técnica	Tempo de execução (em segundos)		
	900 Imagens	2700 Imagens	4500 Imagens
DCT	34,06	38,85	66,14
CNN	63,55	165,07	307,60

Fonte: Autora (2024).

3.2 EXPERIMENTO 2

Para o experimento 2 tem-se a comparação direta dos resultados alcançados utilizando a técnica da CNN (vista na Seção 2.3) com a aplicação da técnica do LSH² (vista na Seção 2.2.2).

3.2.1 Implementação da Técnica RIS utilizando LSH

O LSH utiliza como entrada os vetores de características extraídos da CNN, conforme estabelecido na Seção 3.1.2. Sua estrutura armazena uma coleção de hiperplanos gerados aleatoriamente, onde cada hiperplano está associado a uma tabela de *hash*. Para cada vetor de busca, calcula-se a projeção em relação aos hiperplanos, gerando um código binário no qual cada posição é definida como 1 se a projeção for positiva e 0 caso contrário.

Assim, ao realizar uma busca, o vetor de consulta é projetado em relação aos hiperplanos, gerando um código binário. Em seguida, a tabela de *hash* é consultada para identificar os possíveis candidatos que podem ser similares. Os candidatos similares são selecionados com base no fato de possuírem o mesmo código de *hash* gerado pela projeção do vetor de busca.

² Sempre que o algoritmo LSH for mencionado nos experimentos, considera-se que ele é aplicado diretamente aos vetores de características extraídos pela CNN.

O cálculo de distância é feito utilizando a similaridade do cosseno, visto na Seção 2.2.2.1.

3.2.2 Análise dos Resultados do Experimento 2

Conforme resultados apontados na Tabela 5, nos casos de realce de nitidez e espelhamento, a CNN tem resultados próximos ao LSH, conseguindo mais acertos por uma margem de 1,62% e 0,94%, respectivamente. O LSH conseguiu maiores taxas de acertos se comparado com a CNN em sete das nove transformações, obtendo destaque nas operações de rotação, dessaturação, desfoque gaussiano e corte, com um aumento de, respectivamente, 8,62%, 18,03%, 22,89% e 14,82%.

Dessa forma, o uso do LSH demonstrou ser eficaz para melhorar a taxa de acurácia da CNN. Esses resultados suportam a teoria de Gaillard (2017), que fala que a aplicação do LSH manteria a acurácia dos resultados obtidos pela CNN.

Tabela 5 – Porcentagem de acertos obtidos com as técnicas CNN e LSH (Total de 4500 imagens).

Transformação	CNN - Taxa de Acertos	LSH - Taxa de Acertos
Imagem Original	90,06%	92,73%
Desfoque Gaussiano	63,42%	86,31%
Dessaturação	60,88%	78,91%
Alteração de Resolução	89,11%	91,55%
Realce de Nitidez	93,04%	91,42%
Rotação (90°)	69,15%	77,77%
Espelhamento Horizontal	92,42%	91,48%
Corte	27,62%	42,44%
Inserção de Elemento Gráfico	88,97%	92,51%

Fonte: Autora (2024).

Na média de tempo de execução o LSH obteve melhores resultados se comparado com a CNN, conforme demonstrado na Tabela 6. Essa redução tornou-se ainda mais evidente à medida que o número de imagens aumentou, atingindo uma diminuição de 29,7% no caso de 4500 imagens. Esses resultados demonstram a eficiência do LSH na redução do custo computacional da busca em um grande volume de dados.

Tabela 6 – Média de tempo de execução obtida com as técnicas CNN e LSH.

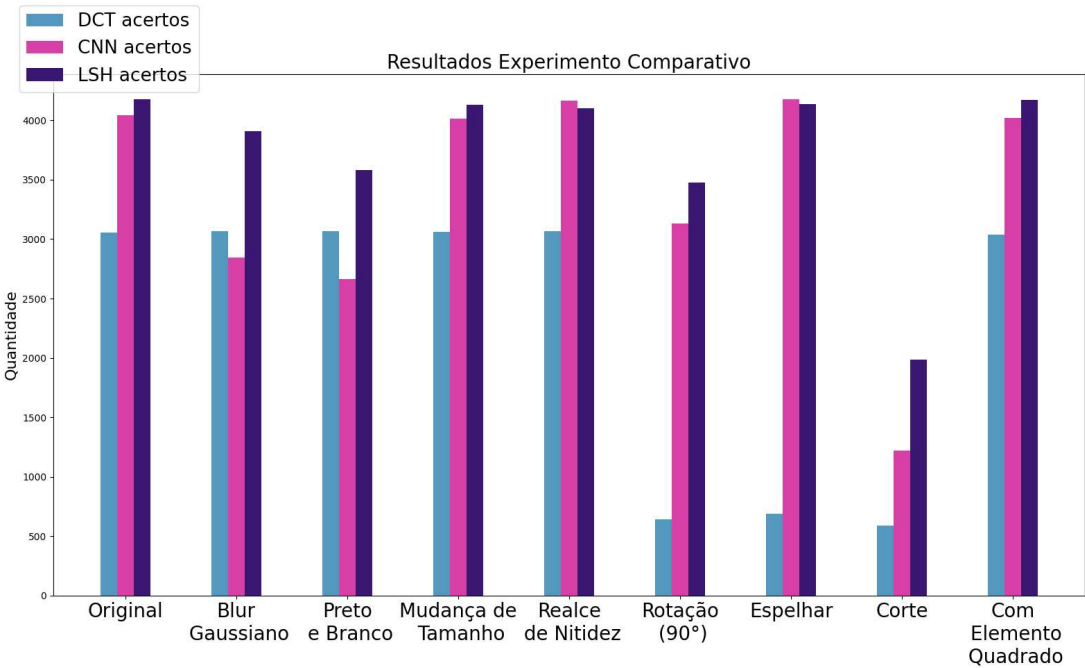
Técnica	Tempo de execução (em segundos)		
	900 Imagens	2700 Imagens	4500 Imagens
CNN	63,55	165,07	307,60
LSH	51,00	123,59	216,15

Fonte: Autora (2024).

3.3 COMPARAÇÃO DAS TÉCNICAS DCT, CNN E LSH

O gráfico disponível na Figura 11 representa a relação da taxa de porcentagens de acertos entre as três técnicas – DCT, CNN e LSH – considerando os resultados vistos no Experimento 1 e Experimento 2. De forma geral, o LSH apresentou o melhor desempenho em várias categorias, se mostrando resiliente às alterações aplicadas, seguido pela CNN, enquanto o DCT apresentou resultados com menor taxa de acurácia.

Figura 11 – Porcentagem de acertos obtidos com as técnicas DCT, CNN e LSH.



Fonte: Autora (2024).

A Tabela 7 apresenta a comparação dos tempos de execução das três técnicas analisadas com diferentes volumes de imagens. Observa-se que o DCT é a técnica com o menor tempo de execução, porém também apresenta o pior desempenho em termos de acurácia, conforme evidenciado nos resultados do gráfico. O LSH, que obteve a maior porcentagem de acertos, apresenta tempos de execução menores em comparação com a CNN, obtendo um equilíbrio entre desempenho de tempo e acurácia.

Tabela 7 – Média de tempo de execução obtida com as técnicas DCT, CNN e LSH.

Técnica	Tempo de execução (em segundos)		
	900 Imagens	2700 Imagens	4500 Imagens
DCT	34,06	38,85	66,14
CNN	63,55	165,07	307,60
LSH	51,00	123,59	216,15

Fonte: Autora (2024).

4 CONCLUSÃO

Com a presente pesquisa, foi possível observar e comparar técnicas para realizar a busca reversa por imagens, destacando-se o uso da CNN e do *hashing* perceptivo DCT. Além disso, foi utilizado o LSH aleatório em conjunto com a CNN, uma técnica para particionamento do espaço de busca, que contribui para melhorar a assertividade dos resultados em grandes bases de dados.

A acurácia dessas técnicas frente a modificações nas imagens foi medida, simulando cenários típicos de um mecanismo de busca. O desempenho de tempo também foi comparado, visto que a busca reversa de imagens geralmente opera sob um grande conjunto de imagens.

A busca reversa de imagens se destaca como uma ferramenta valiosa em diversas áreas, incluindo mecanismos de pesquisa e soluções voltadas à segurança computacional. Suas aplicações vão desde a detecção de fraudes e violação de direitos autorais até a identificação de falsificação de documentos.

Os resultados indicam que, quando comparada ao *hashing* perceptivo DCT, a CNN apresenta um desempenho mais robusto. Essa robustez pode ser atribuída ao processo de aumento de dados durante a fase de treinamento, onde são realizadas diversas alterações visuais nas figuras (GAILLARD, 2017). Além disso, a utilização do LSH aleatório para reduzir o espaço de busca da CNN manteve a precisão dos resultados, ao mesmo tempo em que aprimorou a taxa de acertos e reduziu o tempo médio de busca.

Para futuros estudos, recomenda-se ampliar os experimentos realizados neste estudo, incorporando análises estatísticas mais robustas. Além disso, a utilização de um conjunto de dados maior e mais diversificado é uma alternativa promissora para aumentar a confiabilidade das análises. Outro ponto a ser explorado é a adoção de diferentes técnicas, como novos algoritmos de *hashing* perceptivo e variações do LSH, além de avaliar outros *frameworks* de CNN.

Outro aspecto promissor para pesquisas futuras é a aplicação dos algoritmos de busca reversa na detecção de *deepfakes*, com o objetivo de verificar se um vídeo foi manipulado a partir de um conteúdo original ou se mantém sua autenticidade.

REFERÊNCIAS

ALOMAR, Khaled; AYSEL, Halil Ibrahim; CAI, Xiaohao. Data Augmentation in Classification and Segmentation: A Survey and New Strategies. **Journal of Imaging**, v. 9, n. 2, 2023. ISSN 2313-433X.

CHARIKAR, Moses S. Similarity estimation techniques from rounding algorithms. *In: PROCEEDINGS of the thirty-fourth annual ACM symposium on Theory of computing.* [S.l.: s.n.], 2002. P. 380–388.

CHI, Lianhua; ZHU, Xingquan. Hashing Techniques: A Survey and Taxonomy. **ACM Comput. Surv.**, Association for Computing Machinery, New York, NY, USA, v. 50, n. 1, abr. 2017. ISSN 0360-0300.

DEISENROTH, Marc Peter; FAISAL, A. Aldo; ONG, Cheng Soon. **Mathematics for Machine Learning**. [S.l.]: Cambridge University Press, 2020.

DEMUTH, Howard B.; BEALE, Mark H.; DE JESS, Orlando; HAGAN, Martin T. **Neural Network Design**. 2nd. Stillwater, OK, USA: Martin Hagan, 2014. ISBN 0971732116.

DOMO. **Data Never Sleeps 10.0**. [S.l.: s.n.], 2022. Accessed: 2024-10-26. Disponível em: <https://www.domo.com/data-never-sleeps#data>.

DOUZE, Matthijs *et al.* The 2021 image similarity dataset and challenge. **arXiv preprint arXiv:2106.09672**, 2021.

DU, Ling; HO, Anthony T.S.; CONG, Runmin. **Perceptual hashing for image authentication: A survey**. [S.l.]: ELSEVIER, 2020.

ESCAP, UN *et al.* Trust and Security Using Digital Technologies. UN. ESCAP, 2024.

GAILLARD, Mathieu. **Perceptual Hashing using Convolutional Neural Networks for Large Scale Reverse Image Search**. [S.l.]: INSA, 2017.

GEORGE, A Shaji; BASKAR, T; PANDEY, Digvijay. Establishing Global AI Accountability: Training Data Transparency, Copyright, and Misinformation, 2024.

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

HASSAN, Muneeb ul. **VGG16 – Convolutional Network for Classification and Detection**. [S.l.: s.n.], nov. 2018. <https://neurohive.io/en/popular-networks/vgg16/>. Accessed: 2024-11-22.

JAFARI, Omid; MAURYA, Preeti; NAGARKAR, Parth; ISLAM, Khandker Mushfiqul; CRUSHEV, Chidambaram. A survey on locality sensitive hashing algorithms and their applications. **arXiv preprint arXiv:2102.08942**, 2021.

KOMARGODSKI, Ilan; NAOR, Moni; YOGEV, Eylon. Collision Resistant Hashing for Paranoids: Dealing with Multiple Collisions. *In*: NIELSEN, Jesper Buus; RIJMEN, Vincent (Ed.). **Advances in Cryptology – EUROCRYPT 2018**. Cham: Springer International Publishing, 2018. P. 162–194.

LAVRENKO, Victor. **LSH.9 Locality-sensitive hashing: how it works**. [S.l.: s.n.], 2015. YouTube. Accessed: 2024-10-07. Disponível em: <https://youtu.be/Arni-zkqMBA?si=u-zEdudQaWhuJhh5>.

MEIXNER, Albert; UHL, Andreas. Robustness and security of a wavelet-based CBIR hashing algorithm. *In*: PROCEEDINGS of the 8th Workshop on Multimedia and Security. Geneva, Switzerland: Association for Computing Machinery, 2006. (MM&Sec '06), p. 140–145.

MITCHELL, Tom M; MITCHELL, Tom M. **Machine learning**. [S.l.]: McGraw-hill New York, 1997. v. 1.

OHI, Debbie Ridpath. **Copyright Infringement and Sharing Art Online**. [S.l.: s.n.], 2020. Accessed: 2024-12-01. Disponível em: <https://www.inkygirl.com/inkygirl-main/2020/9/18/copyright-infringement-and-sharing-art-online.html>.

PIOSENKA, Gerald. **Butterfly and Moths Image Classification 100 species**. [S.l.: s.n.], 2022. Accessed: 2024-10-04. Disponível em: <https://www.kaggle.com/datasets/gpiosenka/butterfly-images40-species/data>.

PROKOS, Jonathan; FENDLEY, Neil; GREEN, Matthew; SCHUSTER, Roei; TROMER, Eran; JOIS, Tushar; CAO, Yinzhi. Squint Hard Enough: Attacking Perceptual Hashing with Adversarial Machine Learning. *In*: 32ND USENIX Security

Symposium (USENIX Security 23). Anaheim, CA: USENIX Association, ago. 2023. P. 211–228.

QAZANFARI, Hamed; ALYANNEZHADI, Mohammad M.; KHOSHDAREGI, Zohreh Nozari. **Advancements in Content-Based Image Retrieval: A Comprehensive Survey of Relevance Feedback Techniques**. [S.l.: s.n.], 2023. arXiv: [2312.10089](https://arxiv.org/abs/2312.10089) [cs.CV]. Disponível em: <https://arxiv.org/abs/2312.10089>.

RAJARAM, Shyamsundar; SCHOLZ, Martin. Client-friendly classification over random hyperplane hashes. *In*: SPRINGER. JOINT European Conference on Machine Learning and Knowledge Discovery in Databases. [S.l.: s.n.], 2008. P. 250–265.

SAMANTA, Priyanka; JAIN, Shweta. Analysis of Perceptual Hashing Algorithms in Image Manipulation Detection. **Procedia Computer Science**, v. 185, p. 203–212, 2021. Big Data, IoT, and AI for a Smarter Future. ISSN 1877-0509.

SCHIAVO, Alessio; MINUTELLA, Filippo; DAOLE, Mattia; GOMEZ, Marsha Gomez. Sketches image analysis: Web image search engine using LSH index and DNN InceptionV3. **arXiv preprint arXiv:2105.01147**, 2021.

SIMONYAN, Karen; ZISSERMAN, Andrew. Very Deep Convolutional Networks for Large-Scale Image Recognition. **arXiv preprint arXiv:1409.1556**, 2015. arXiv: [1409.1556](https://arxiv.org/abs/1409.1556) [cs.CV].

STRANG, Gilbert. The Discrete Cosine Transform. **SIAM Review**, v. 41, n. 1, p. 135–147, 1999. eprint: <https://doi.org/10.1137/S0036144598336745>.

TZELEPI, Maria; TEFAS, Anastasios. **Deep convolutional learning for Content Based Image Retrieval**. [S.l.]: ELSEVIER, 2017.

WAN, Gary Gang; LIU, Zao. Content-based information retrieval and digital libraries. **Information Technology and Libraries**, 2008. ISSN 0730-9295, 2163-5226.

WAZLAWICK, Raul Sidnei. **Metodologia de Pesquisa para Ciência da Computação**. [S.l.]: GEN LTC, 2020. ISBN 9788595151093.

WEIDMAN, S. **Deep Learning from Scratch: Building with Python from First Principles**. [S.l.]: O'Reilly Media, Incorporated, 2019. ISBN 9781492041412.

WEKESA, Esther Nanjala; DECUSATIS, Casimer; ZHU, Andy. A Black Box Comparison of Machine Learning Reverse Image Search for Cybersecurity OSINT Applications. **Electronics**, v. 12, n. 23, 2023. ISSN 2079-9292.

WENG, Li; JHUO, I-Hong; CHENG, Wen-Huang. Perceptual Hashing for Large-Scale Multimedia Search. *In: BIG Data Analytics for Large-Scale Multimedia Search*. [S.l.]: John Wiley & Sons, Ltd, 2019. cap. 9, p. 239–265. ISBN 9781119376996. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781119376996.ch9>.

WU, Jianxin. Introduction to convolutional neural networks. **National Key Lab for Novel Software Technology. Nanjing University. China**, v. 5, n. 23, p. 495, 2017.

YAN, Chris. **Understanding Hamming Distance: A Measure of Similarity**. [S.l.: s.n.], 2024. <https://chrisyandata.medium.com/understanding-hamming-distance-a-measure-of-similarity-698ae2cb0ef6>. Accessed: 2024-11-08.

ZAUNER, Christoph. **Implementation and Benchmarking of Perceptual Image Hash Functions**. v. 43. [S.l.], 2010. Disponível em: <https://api.semanticscholar.org/CorpusID:17075066>.

ZHU, Zhiying; ZHOU, Hang; XING, Siyuan; QIAN, Zhenxing; LI, Sheng; ZHANG, Xinpeng. **Perceptual Hash of Neural Networks**. [S.l.]: Symmetry, 2022.