SISTEMA DE VISÃO COMPUTACIONAL APLICADO À INSPEÇÃO INDUSTRIAL[1]

FIGUEREDO, Fernando Luiz de[2]
MARTINS, Hugo Magalhães[3]

RESUMO

Este estudo de caso apresenta a elaboração e teste de um sistema de visão computacional capaz de classificar objetos com base em características visuais. O sistema tem como principal objetivo a aplicação em etapas de inspeção de qualidade em diferentes setores produtivos, como indústrias de manufatura, alimentícia e agronegócio. O sistema foi elaborado para ser adaptável a diferentes necessidades, permitindo que o próprio usuário colete imagens do objeto a ser identificado, defina categorias e treine o modelo utilizando uma interface visual amigável em uma aplicação *web* baseada em Python Flask. Dois estudos de caso foram conduzidos para validar sua aplicabilidade: o primeiro consistiu na classificação de maçãs entre vermelhas e verdes, e o segundo na identificação de prendedores de madeiras, entre inteiros e incompletos. A solução utiliza processamento de imagens, redes neurais convolucionais (CNN) e *transfer learning* baseado na arquitetura *MobileNetV2*. Durante os testes, o sistema foi avaliado com conjuntos de imagens de diferentes quantidades e imagens capturadas por câmeras de telefone celular e em diferentes resoluções, a fim de verificar o impacto do volume de dados e da resolução de imagem no desempenho do sistema. Os resultados demonstram que o sistema é eficiente e versátil, com potencial para ser utilizado em diferentes contextos que demandem inspeção visual automatizada de baixo custo e alta flexibilidade.

Palavras-chave: Visão Computacional, Processamento de Imagens, Inspeção industrial, Inteligência Artificial

INTRODUÇÃO

A inspeção visual é um elemento crucial em diferentes indústrias, sendo indispensável para o reconhecimento de uma variedade de defeitos, não conformidades e variações em elementos como cor, quantidade, posicionamento e posição. Este processo é fundamental para garantir que as demandas de qualidade da empresa e de seus clientes sejam atendidas. (Silva et al. 2018)

A Visão Computacional une conceitos de processamento digital de imagens e Inteligência Artificial para automatizar as inspeções visuais. Esta abordagem tem o potencial de melhorar significativamente os fatores de qualidade e produtividade nas indústrias onde são aplicados. Ao substituir a inspeção humana pela inspeção automatizada, é possível reduzir a ocorrência de erros e aumentar a eficiência do processo de inspeção. (Barelli, 2018)

De acordo com Silva et al. (2018), o desenvolvimento de um sistema de inspeção visual envolve etapas fundamentais baseadas em processamento de imagens e inteligência artificial: coleta das imagens por sensores ópticos (geralmente câmeras), pré-

[1] Trabalho de Conclusão de Curso (TCC) para a pós-graduação Lato Sensu de Especialização em Controle e Automação.

[2] Pós-graduando Lato Sensu de Especialização em Controle e Automação; IFSP; São Paulo; São Paulo; fernando.figueredo@aluno.ifsp.edu.br.
[3] Mestre em Automação e Controle; IFSP; São Paulo; São Paulo; hugo.magalhaes@ifsp.edu.br.

processamento para realce e redução de ruídos, segmentação da imagem em regiões de interesse, extração de características visuais como formas, cores e texturas e, por fim, a classificação, etapa em que as imagens são rotuladas por algoritmos, com destaque para o uso de redes neurais artificiais na distinção entre itens.

Neste trabalho, utilizou-se a arquitetura MobileNetV2 como modelo de classificação, o que permitiu que o sistema fosse treinado com poucas amostras de dados, o que é possível porque essa rede já é previamente treinada em grandes bases de dados (*ImageNet)*, facilitando o uso do chamado *transfer learning*, onde a rede neural transfere o aprendizado prévio para uma aplicação, reduzindo a necessidade de treinamentos com conjuntos maiores. (Dong, 2020)

Foram testadas as influências de variáveis na etapa de coleta (resolução da câmera) e volume de dados de treinamento (tamanho do dataset) no desempenho final do sistema. Além disso, uma aplicação *web* com interface amigável foi desenvolvida a fim de facilitar a utilização do sistema por usuários.

OBJETIVOS

Desenvolver um sistema de visão computacional que possa ser treinável para diferentes objetos e seja capaz de classificá-los entre duas categorias (Conforme e Não Conforme)

Realizar experimentos para testar a influência das variáveis tamanho de conjunto de dados e resolução da imagem capturada no desempenho geral desse sistema

Desenvolver e disponibilizar ao uso público o conjunto de dados utilizado nos testes (maçãs vermelhas e maçãs verdes e prendedores de madeira completos e incompletos em diferentes resoluções de imagem).

METODOLOGIA

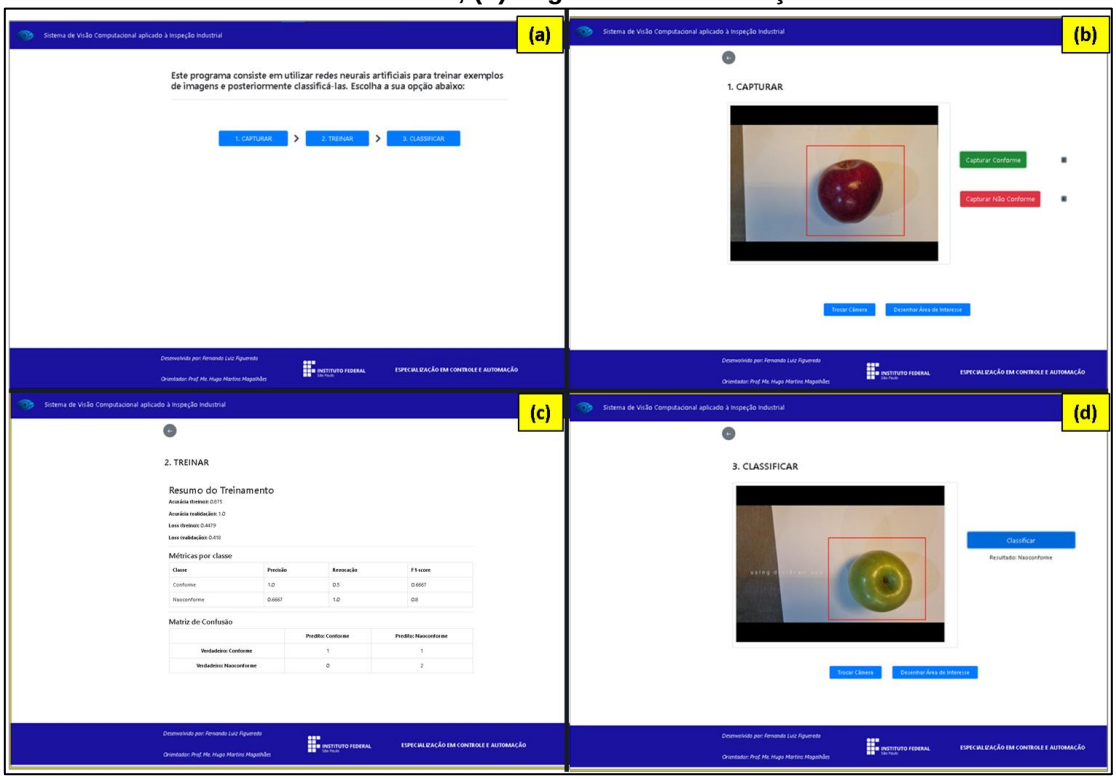
O sistema desenvolvido tem seu funcionamento dividido em três etapas Captura, Treinamento e Classificação. A primeira etapa, captura, é onde o usuário vai definir o objeto a ser reconhecido e alimentará o sistema com imagens das categorias conforme e não conforme, onde “conforme” é um exemplo do objeto esperado e o “não conforme” é o objeto com alguma diferença e imperfeição que se deseja reconhecer. As imagens são capturadas por uma câmera conectada ao computador que está executando o sistema e é importante que o conjunto de cada categoria possua uma mesma quantidade de imagens, evitando o desbalanceamento do sistema. (Barelli, 2018)

A segunda etapa, treinamento, processa o conjunto de imagens fornecido na etapa 1 e realiza operações de pré-processamento dessas imagens (normalização e remoção de ruídos), reduzindo o tempo necessário para o treinamento para, em seguida, treinar um modelo de rede neural artificial baseado na arquitetura *MobileNetV2.* Este modelo é treinado e avaliado em quesitos de acurácia, erro (Loss) e F1-Score, métricas comuns para avaliação desse tipo de rede neural, como indicado por Junior et al. (2022).

A terceira e última etapa, classificação, faz o carregamento do modelo treinado na etapa 2 para classificar dados que não foram usados no treinamento, de forma que a câmera capture imagens pare serem classificadas de acordo com as categorias Conforme e Não Conforme.

O sistema é encapsulado em uma aplicação web de interface amigável, baseada em *Flask*, um *microframewor*k da linguagem Python específico para desenvolvimento de aplicações leves e com possibilidade de escalabilidade devido a sua grande quantidade de extensões que podem ser instaladas ou não, a depender do uso. (Grinberg, 2018). A aplicação possui quatro páginas, sendo elas Página Inicial, Página de Captura, Página de Treinamento e Página de Classificação, conforme ilustrado pela Figura 1.

Figura 1 - Páginas da aplicação *web* (a) Página Inicial; (b) Página de Captura; (c) Página de Treinamento; (d) Página de Classificação



Fonte: Os autores

A Página Inicial atua como ponto de navegação entre as demais páginas da aplicação. Na página de Captura, o usuário pode visualizar o feed da câmera em tempo real, selecionar o dispositivo de captura, desenhar uma área de interesse da câmera que será capturada e registrar as imagens para as categorias "Conforme" e "Não Conforme". A página de Treinamento executa o processo de aprendizado do modelo, exibindo ao final métricas como acurácia, erro (loss) e F1-score. Já a Página de Classificação também apresenta o feed da câmera e as opções de configuração, similar a Página de Captura, porém com um botão para classificar as imagens capturadas, com base no modelo treinado.

Para os testes de validação do sistema, foram utilizados 2 cenários. O primeiro voltado a reconhecimento de geometria com prendedores de madeira inteiros (“Conforme”) e quebrados (“Não Conforme”) e o segundo voltado a diferenciação por cor, com maçãs vermelhas ("Conforme") e verdes ("Não Conforme"). O sistema utiliza por padrão uma proporção de 80% das imagens para treinamento e 20% das imagens para validação

Os testes verificaram a influência de fatores no desempenho do modelo, sendo eles: quantidade de imagens por categoria (10 imagens, 50 imagens e 100 imagens) e resolução da imagem capturada (480p, 720p e 1080p). Em ambos os testes, as imagens foram capturadas com um smartphone Samsung Galaxy S21 5G. O resultado desses testes pode ser observado na sessão de Resultados e Discussão. A Figura 2 demonstra a diferença de imagem entre as 3 resoluções testadas.

Figura 2 – Diferença entre as três resoluções testadas (1080p, 720p e 480p)

Fonte: Os autores

RESULTADOS E DISCUSSÃO

As métricas utilizadas são a Acurácia (%), que calcula o percentual de predições acertadas em comparação com o total de valores; Erro (*Loss*), calculado por meio da função de perda de *categorical cross-entropy*, que mede a diferença entre a distribuição de probabilidade verdadeira e a estimada. E o chamado F-1 Score, uma média harmônica entre precisão (quantos dos positivos preditos estavam certos) e recall (quantos dos positivos foram corretamente identificados). (Keras, 2025)

A Tabela 1 e a Tabela 2 mostram resultado de testes realizados para conjunto de dados com diferentes quantidades de imagens por categoria (10, 50 e 100 imagens) para medir o impacto na capacidade de reconhecimento do modelo. Ambos os teste utilizaram imagens com uma resolução padrão de 720p.

Tabela 1 – Métricas medidas para diferentes tamanhos de conjunto de dados (Prendedores)

N° de imagens por categoria	Acurácia (%)	Erro (Loss)	F1-Score
10	87,50	0,3363	0,6667
50	95,00	0,2214	0,9474
100	94,74	0,1254	0,9737

Tabela 2 - Métricas medidas para diferentes tamanhos de conjunto de dados (Maçãs)

N° de imagens por categoria	Acurácia (%)	Erro (Loss)	F1-Score
10	62,50	0.4479	0,6667
50	92,35	0,1696	0,9332
100	97,50	0,1187	0,9876

A Tabela 3 e a Tabela 4 mostram resultado de testes realizados para conjunto de dados com diferentes resoluções de imagens (480p, 720p e 1080p), buscando medir o impacto da resolução de imagem no reconhecimento do modelo. Ambos os teste utilizaram um conjunto com 50 imagens por categoria.

Tabela 3 - Métricas medidas para diferentes resoluções de imagem (Prendedores)

Resolução das imagens	Acurácia (%)	Erro (Loss)	F1-Score
480p	65,00	0,5613	0.4286
720p	93,50	0,2575	0,8871
1080p	95,00	0,2308	0,9242

Tabela 4 - Métricas medidas para diferentes resoluções de imagem (Maçãs)

Resolução das imagens	Acurácia (%)	Erro (Loss)	F1-Score
480p	90,55	0,1344	0,9430
720p	94,89	0,1596	0,9596
1080p	98,89	0,1033	0,9912

Os resultados obtidos indicam que o sistema mostra um melhor desempenho na diferenciação de objetos com atributos baseados em cor do que em geometria, como evidenciado pela acurácia e F1-score superiores nos testes com maçãs (vermelhas e verdes), sobretudo para conjuntos maiores de dados.

A sensibilidade à diferentes resoluções foi maior verificada na diferenciação de geometria (prendedores de madeira), sugerindo que a identificação de contornos e formas é mais impactada pela baixa qualidade das imagens, diferentemente da diferenciação de cores das maçãs, que se saiu melhor mesmo para imagens de baixa qualidade (480p).

Os testes também mostraram que o ganho de desempenho entre conjuntos com 10 e 50 imagens foi significativamente maior do que entre 50 e 100 imagens, demonstrando que depois de uma certa quantidade de imagens no conjunto dados, o sistema tende a ficar estável e adicionar novas imagens tem um impacto reduzido.

De forma similar, a diferença de desempenho do sistema entre conjuntos de dados de treinamento com resolução de 480p para 720p foi maior (sobretudo para os prendedores) do que entre a resolução de 720p e 1080p, mostrando que o sistema atinge um certo patamar de desempenho mesmo com imagens com resolução maior.

CONSIDERAÇÕES FINAIS/CONCLUSÃO

O sistema desenvolvido obteve resultados robustos (média de acurácia superior a 90%, Erro (loss) inferior a 0,15 e F1-Score superior a 0,90) mesmo utilizando câmeras com limitações de hardware e conjuntos de imagens relativamente pequenos (até 100 imagens), o que mostra que o *transfer learning* aliado a arquitetura MobileNetV2 funcionaram corretamente para a aplicação proposta.

Para trabalhos futuros, recomenda-se o teste desse sistema em ambientes com variações de iluminação, ruído ou com objetos parcialmente ocultos, a implementação do treinamento incremental, onde o usuário alimenta o sistema com novas imagens ao longo do tempo sem a necessidade de recomeçar o processo do início, além de testes com mais categorias (não apenas duas) e testes com outras arquiteturas de rede. O sistema ainda pode ser integrado uma esteira com separador que automatizaria a separação de objetos nas duas categorias reconhecidas sem necessidade de intervenção humana.

REFERÊNCIAS

BARELLI, Felipe. **Introdução à visão computacional: Uma abordagem prática com Python e OpenCV**. Editora Casa do Código, 2018.

DONG, Ke et al. MobileNetV2 model for image classification. In: **2020 2nd International Conference on Information Technology and Computer Application (ITCA)**. IEEE, 2020. p. 476-480.

GRINBERG, Miguel. **Flask web development: developing web applications with python**. "O'Reilly Media, Inc.", 2018.

JUNIOR, Guanis B. Vilela et al. Métricas utilizadas para avaliar a eficiência de classificadores em algoritmos inteligentes. **Revista CPAQV–Centro de Pesquisas Avançadas em Qualidade de Vida| Vol**, v. 14, n. 2, p. 2, 2022.

KERAS. **Classification metrics**. Disponível em: https://keras.io/api/metrics/classification_metrics/. Acesso em: 05 ago. 2025.

SILVA, Ricardo Luhm et al. **Machine vision systems for industrial quality control inspections. In: Product Lifecycle Management to Support Industry 4.0:** 15th IFIP WG 5.1 International Conference, PLM 2018, Turin, Italy, July 2-4, 2018, Proceedings 15. Springer International Publishing, 2018. p. 631-641.