Universidade de São Paulo
Escola Politécnica
Programa de Educação Continuada em Engenharia
Especialização em Inteligência Artificial

Rafael Rocha Fernandes

Inspeção visual em linha de produção utilizando visão computacional

RAFAEL ROCHA FERNANDES

Inspeção visual em linha de produção utilizando visão computacional

— Versão Original —

Monografia apresentada ao Programa de Educação Continuada em Engenharia da Escola Politécnica da Universidade de São Paulo como parte dos requisitos para conclusão do curso de Especialização em Inteligência Artificial.

Orientador: Profa. Dra. Larissa Driemeier

Catalogação-na-publicação

Fernandes, Rafael Rocha
Inspeção Visual em linha de produção utilizando visão computacional – São Paulo, 2024.
150p.

Monografia (Especialização em Inteligência Artificial) – Escola Politécnica da Universidade de São Paulo. PECE – Programa de Educação Continuada em Engenharia.

1. Manufatura 2. Produção 3. Qualidade 4. Inteligência artificial 5. Aprendizado de máquina 6. Visão Computacional.
I. Universidade de São Paulo. Escola Politécnica. PECE – Programa de

Agradecimentos

Primeiramente à minha família, minha esposa Thainá e minha filha Laura que me proporcionam momentos de felicidades que é o combustível para seguirmos sempre em frente, e especialmente à minha esposa por cuidar da nossa filha no período em que estava na universidade, possibilitando meus estudos.

Aos meus pais Ademir e Márcia por me educar e me incentivar nos estudos desde criança.

À profa. Dra. Larissa Driemeier e ao prof. Dr. Thiago de Castro Martins, coordenadores do curso de Inteligência Artificial, pelo excelente curso e pelos ensinamentos durante as aulas.

Aos colegas do curso pela participação e apoio durante o período em que estivemos juntos.

Aos outros professores e funcionários do Programa de Educação Continuada da Poli-USP que são um exemplo para o nosso país, me senti muito privilegiado por fazer parte desse ambiente.

Sumário

Resumo

FERNANDES, Rafael Rocha. *Inspeção visual em linha de produção utilizando visão computacional* 2024. Monografia (Especialização em Inteligência Artificial) – Escola Politécnica da Universidade de São Paulo. PECE – Programa de Educação Continuada em Engenharia. Universidade de São Paulo, São Paulo, 2024.

A indústria necessita cada vez mais de recursos tecnológicos para aprimorar sua qualidade, produtividade, reduzir custos e melhorar sua eficiência energética contribuindo com o meio ambiente ao consumir menos recursos e emitir menos poluentes. O uso da inteligência artificial pode certamente contribuir para esses desafios da indústria. O aprendizado de máquina, que é a utilização de dados para treinamento de modelos é cada vez mais utilizado para aprimorar processos industriais. O presente trabalho tem como objetivo utilizar visão computacional e aprendizado de máquina para realizar tarefas de inspeções de qualidade, onde é necessário carregar um conjunto de imagens conhecidas, de classes diferentes, treinar o modelo para então executar a tarefa de inspeção utilizando fotos reais e realizar inferência no modelo já treinado. Foi utilizada a arquitetura MobileNetV2 devido ao seu pequeno tamanho e baixo consumo de memória pode ser utilizado em qualquer dispositivo móvel ou equipamento industrial. O modelo em si só pode ser acessado através de algum script ou software, então para isso foi desenvolvido um sistema em nuvem para carregar as imagens, processar o treinamento em uma máquina virtual específica, gerenciar os diversos modelos, tudo através de uma interface web amigável e de fácil utilização pelo usuário. Para realizar a inferência não é necessário estar conectado à nuvem, pois o modelo pode rodar localmente no dispositivo, a nuvem só é utilizada para gerenciar as imagens e os modelos. Como foi utilizada uma rede neural convolucional pré treinada, mesmo com um pequeno conjunto de dados e treinamento realizado com poucas épocas foi possível obter resultados muito bons.

Palavras-chave: Aprendizado de máquina. AWS. Indústria 4.0. Inteligência artificial. Manufatura. MobileNetV2. Produção. Qualidade. Redes Neurais Convolucionais. SageMaker. Visão Computacional.

Abstract

FERNANDES, Rafael Rocha. *Quality inspection in production line using computer vision*. 2024.. Monografia (Specialization in Artificial Intelligence) – Escola Politécnica da Universidade de São Paulo. PECE – Programa de Educação Continuada em Engenharia. University of São Paulo, São Paulo, Brazil. 2024.

The industry needs technological resources to improve its quality, productivity, reduce costs and use energy more efficiently, contributing to the environment and carbon footprint by consuming less resources. The usage of artificial intelligence can certainly contribute to these industry challenges. Machine learning, which is the use of data to train models, is used to improve industrial processes. The current work aims to use computer vision and machine learning to perform quality inspection tasks, where it is necessary to load a set of known images, from different classes, train the model to then perform the inspection task using real photos and execute the inference on the trained model. The MobileNetV2 architecture was used due to its small size and low memory consumption and can be used on any mobile device or industrial equipment. The model itself can only be accessed through some script or software, so for this purpose a cloud system was developed to load the images, process the training in a specific virtual machine, manage the different models, all through a friendly web interface and easy to use by the user. To perform inference, it is not necessary to be connected to the cloud, as the model can run locally on the device, or on premises environment, the cloud is only used to manage the images and models. As a pre-trained convolutional neural network was used, even with a small set of data and training done with a few epochs, it was possible to obtain very good results.

Keywords: Artificial Intelligence. AWS. Computer Vision. CNN. Convolutional Neural Networks. Industry 4.0 Machine Learning. Manufacturing. MobileNetV2. Production. Quality. SageMaker.

Capítulo **1**

Introdução

A Indústria desempenha um papel muito importante na sociedade pois é responsável pela produção de produtos essenciais para a sobrevivência e conforto das pessoas na vida moderna. Alimentos, medicamentos, vestuário, materiais de construção, eletrônicos e veículos são produzidos em larga escala devido aos avanços da indústria nos últimos anos.

Além disso a indústria também contribui para a economia e desempenha outros papeis importantes na sociedade como geração de empregos, desenvolvimento regional e também inovação tecnológica.

Desde a revolução industrial, que permitiu a produção em larga escala, a indústria passou por muitas transformações com a eletrificação, avanço das telecomunicações, computação, automação industrial, armazenamento de dados e inteligência artificial.

Também existem muitos desafios para que exista uma produção sustentável com qualidade nos processos e produtos, segurança e bem estar dos colaboradores, diminuição de custos e melhora na eficiência energética, com responsabilidade social e respeito ao meio ambiente.

A inteligência artificial pode certamente contribuir com os desafios da indústria, e o aprendizado de máquina pode utilizar dados para realizar tarefas que são complexas, reduzir ou eliminar desperdícios realizando atividades simples ou repetitivas, possibilitando às pessoas realizarem tarefas que agregam mais valor ao negócio.

Com a utilização de visão computacional combinado com aprendizado de máquina é possível treinar modelos para detectar objetos ou pessoas, classificar ou identificar problemas de qualidade em produtos. O modelo pode ser treinado com um conjunto de imagens controladas e um algoritmo específico de acordo com o problema. Uma vez treinado, o modelo está pronto para fornecer o resultado de uma nova imagem que é submetida a ele.

1.1. Motivação

Os modelos de visão computacional e aprendizado de máquina são ferramentas que podemos usar para entender e processar imagens. Geralmente, precisamos escrever scripts para acessá-los, o que requer ferramentas, conhecimento de programação e desenvolvimento de software.

No entanto, para atividades diárias ou em empresas, precisamos de soluções simples e diretas. Não faz sentido ter que escrever um script ou usar um ambiente de desenvolvimento apenas para tarefas básicas, como carregar imagens, treinar um modelo ou comparar imagens.

O objetivo deste trabalho é mostrar o funcionamento de um sistema desenvolvido para inspecionar a qualidade em um processo de produção utilizando um modelo de visão computacional e aprendizado de máquina. Esse sistema pode ser utilizado por operadores de produção ou controle de qualidade para identificar automaticamente montagens incorretas na linha de produção ou em postos de verificação de qualidade. Isso pode ser feito de forma eficiente e sem a necessidade de conhecimento avançado em programação.

Capítulo 2

Referencial teórico

2.1 Visão Computacional

Visão computacional é um campo de inteligência artificial que se destaca por permitir aos algoritmos detectar e processar características em imagens, possibilitando análise e interpretação de dados visuais. A visão computacional vem se desenvolvendo rapidamente nos últimos anos, boa parte do crescimento deve-se à evolução das redes neurais profundas, também conhecidas como Deep Learning.

Até poucos anos atrás eram utilizados algoritmos comuns de visão computacional para realizar tarefas simples, mas como o avanço das redes neurais convolucionais com camadas profundas os modelos se tornaram mais eficientes e atingindo melhores resultados.

Visão computacional é a base do atual trabalho que tem como objetivo identificar características nas imagens para poder processar dados matematicamente identificando padrões processando em cada camada da rede neural.

Com a visão computacional é possível efetuar tarefas de classificação de imagens, onde as imagens são classificadas em classes. Também é possível efetuar tarefas de detecção e segmentação. Ainda existem as IAs Generativas que conseguem gerar novas imagens à partir de textos ou imagens.

2.2 Rede Neural Convolucional

Rede Neural Convolucional (ou CNN, do inglês Convolutional Neural Networks) é uma classe de rede neural artificial que vem sendo muito utilizada e aprimorada nos últimos anos na área de classificação de imagens.

A rede neural convolucional recebe a imagem, converte em 3 vetores de bytes, um para cada cor, com escalas de vermelho (R, do inglês red), verde (G, do inglês green) e preto ou escala de cinza (B, do inglês black). Nas imagens preto e branco é utilizado somente um vetor cinza. Então são aplicados filtros e realizadas convoluções que resultam em um novo vetor de dados. Cada filtro possui uma característica diferente e irá produzir um resultado diferente. No final esses vetores são processados em camadas densas treinadas para a atividade específica. A Figura 2.2.1 demonstra como funciona um filtro executando a convolução.

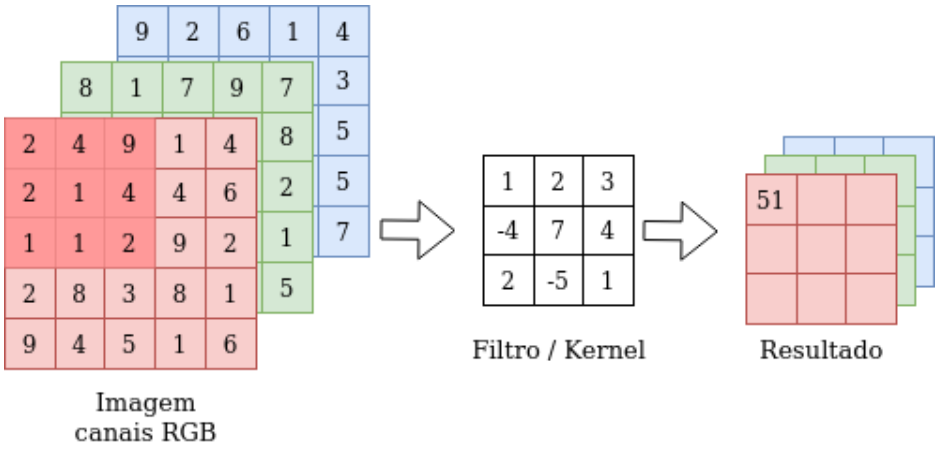

Figura 2.2.1: Exemplo de aplicação de um filtro de convolução. Fonte: RODRIGUES (2020)

A Figura 2.2.2 demonstra uma arquitetura genérica de uma rede neural convolucional que contém a camada de convolução, que é responsável por extrair características da imagem gerando novos vetores com os resultados de saída dos filtros. A camada pooling pode ser utilizada para simplificar o vetor, reduzindo sua dimensão para um processamento mais eficiente, com menor consumo de memória e maior velocidade de processamento. A camada totalmente conectada é responsável por receber os vetores com múltiplas dimensões e gerar na saída um vetor de uma única dimensão que é processado na camada densa que realiza a classificação.

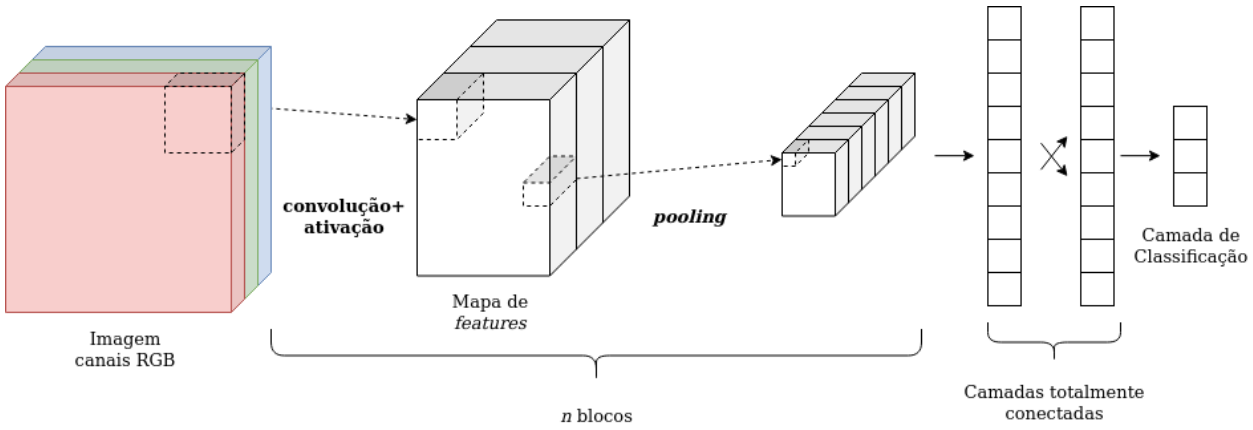

Figura 2.2.2: Arquitetura genérica de uma CNN. Fonte: RODRIGUES (2020)

2.3 Indústria 4.0

Indústria 4.0, também conhecida como quarta revolução industrial é um termo que se refere à utilização de novas tecnologias e sistemas para controle de processos e automação industrial. A Figura 2.3.1 demonstra a quarta revolução industrial, e também as revoluções anteriores.

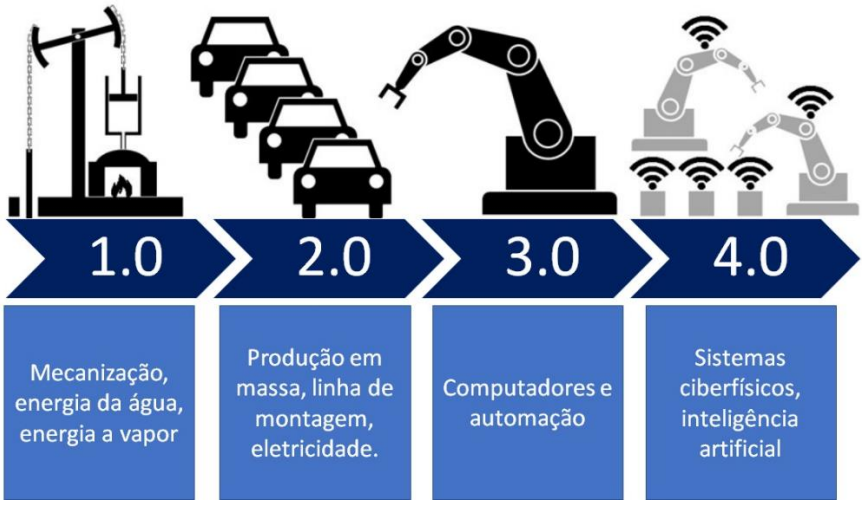

Figura 2.3.1: A quarta revolução industrial. Fonte: MUSSATO (2018)

Graças ao poder de processamento é possível desenvolver sistemas de controle com eletrônica embarcada ou CLPs (controladores lógicos programáveis), estes podem se conectar a outros equipamentos utilizando uma rede de campo, ou protocolos abertos como Modbus, OPC, ou comunicar com um broker através do protocolo MQTT.

As redes também evoluíram bastante permitindo comunicação dentro e fora da empresa, com toda segurança, protocolos de transporte como o TLS (Transport Layer Security) e certificados podem criptografar o conteúdo da mensagem onde somente o produtor e o consumidor podem entender o conteúdo. Firewalls e outros dispositivos de segurança adicionam mais uma camada de segurança a rede.

Muitos dispositivos também estão preparados para comunicação de rede sem fio Wifi-6 ou 5G permitindo mobilidade em robôs e veículos de transporte autônomos. Protocolos como o iPCF-2 podem tornar uma rede sem fio determinística trazendo toda segurança necessária para as máquinas de comunicarem na rede sem fio.

Com o avanço das redes, aumento do poder de processamento e armazenamento é possível criar um ambiente de dados analíticos que possibilitam o uso de algoritmos de inteligência artificial para treinar modelos e resolver problemas cada vez mais complexos.

O uso de computação em nuvem também possibilita uma enorme capacidade de armazenamento e processamento.

Capítulo 3

Métodos

3.1 Implementação

No contexto da manufatura, a montagem e inspeção dos produtos são etapas fundamentais, caracterizadas por um planejamento meticuloso e uma execução precisa, resultando em um conhecimento prévio do resultado final. Dessa forma, optou-se por empregar um método de aprendizado supervisionado para classificação. Esta escolha se justifica pela capacidade de comparar o resultado obtido durante a inspeção

Foi adotada a plataforma em nuvem da AWS (Amazon Web Services) para armazenamento de imagens, treinamento e armazenamento dos modelos, pois já existem serviços dedicados para cada necessidade dentro da plataforma, fornecendo maior flexibilidade na solução, e evita instalação de servidores físicos no local.

Sistemas industriais utilizados por operadores em linha de produção normalmente necessitam de uma rede estável e com baixa latência, por esse motivo a inferência deve ocorrer no próprio dispositivo, sem dependência de serviços externos ou de nuvem.

Os estudos de ISUYAMA e ALBERTINI (2019) demonstram a comparação de performance entre as diferentes arquiteturas de redes neurais convolucionais com modelos Tensor Flow (TF) e Tensor Flow Light (TFL), apresentados na Tabela 3.1.1.

A coluna CNN contém cada arquitetura de rede neural convolucional. A coluna Top-1 Accuracy significa a acurácia, ou seja, a porcentagem do número de acertos divido pelo número total de testes. A coluna Inference Time (ms) indica a média de tempo que o modelo levou para classificar todo o conjunto de testes em milissegundos. A coluna Overall Memory Usage indica consumo médio de memória durante a inferência em Megabytes. A coluna Model File Size contém o tamanho do arquivo do modelo treinado em Kilobytes.

CNN	Top-1 Accuracy.		Inference Time (ms).		Overall Memory Usage (MB).		Model File Size (KB)	
	TF	TFL	TF	TFL	TF	TFL	TF	TFL
DenseNet121	98.18	98.18	1746	168	214.8	46.25	28249	27289
DenseNet169	96.36	96.36	2251	208	298	67.87	50521	48949
DenseNet201	98.18	98.18	2727	252	393.3	90.37	72907	70889
EfficientNetB0	94.54	94.54	1127	145	74.3	26.68	16348	15721
EfficientNetB1	95.75	95.75	1607	197	146.6	37.82	26360	25511
EfficientNetB2	95.75	95.75	1581	210	130.6	45.16	31037	30145
EfficientNetB3	96.96	96.96	1589	273	180.2	62.62	42894	41830
EfficientNetB4	94.54	94.54	2311	396	268.5	92.44	69969	68570
EfficientNetB5	93.93	93.93	2760	556	452.4	139.47	112493	110685
EfficientNetB6	89.09	89.09	4142	725	606.7	192.39	161274	159048
EfficientNetB7	91.51	91.51	4975	946	905.7	280.25	251896	248989
InceptionResNetV2	92.72	92.72	2762	329	737	248.18	213611	212262
InceptionV3	89.09	89.09	1062	156	267.5	110.88	85904	85173
MobileNet	95.15	95.15	415	182	40	29.76	12900	12547
MobileNetV2	94.54	94.54	648	29	25	17.42	9271	8713
ResNet101	95.15	95.15	1545	335	546.2	191.25	167488	165986
ResNet101V2	93.93	93.93	1457	334	541.1	191.81	167343	166253
ResNet152	92.72	92.72	2342	464	783.3	247.92	229158	227051
ResNet152V2	92.12	92.12	2251	475	743.7	251.45	228978	227443
ResNet50	94.54	94.54	799	191	276.5	118.05	92703	91854
ResNet50V2	92.12	92.12	856	188	279	118.38	92599	91979
VGG16	91.51	91.51	208	508	166.7	175.74	57607	57510
VGG19	88.48	88.48	263	649	228.8	196.1	78357	78253
Xception	89.09	89.09	2451	201	269.8	124.23	81990	81309

Tabela 3.1.1: Comparação das redes neurais convolucionais. Fonte: ISUYAMA e ALBERTINI (2019)

Para a solução atual foi escolhida uma rede neural convolucional pré treinada, com isso é possível aplicar o treinamento somente na última camada, o que exige menos épocas e uma menor quantidade de dados.

A arquitetura MobileNetV2 foi adotada pois apresenta o menor consumo de memória, e o modelo com menor tamanho, ainda assim apresenta excelente acurácia e baixo tempo de resposta na inferência, o que torna a MobileNetV2 ideal para utilização em dispositivos móveis.

3.2 Máquina de estados

Para o treinamento do modelo foi criada uma máquina de estados, também conhecida como Step Function no ambiente da AWS. Para cada estado existe um conjunto de parâmetros de entrada que serão utilizados para a execução, e após a execução é gerado um resultado de saída com informações adicionais na forma de logs.

A Figura 3.2.1 demonstra a máquina de estados que foi criada utilizando o módulo Step Function da AWS.

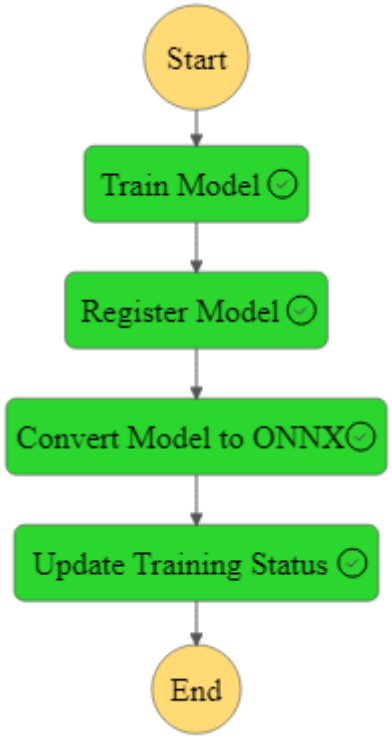

Figura 3.2.1: Máquina de estados para treinamento do modelo.

Existe uma interface Web para o usuário carregar as imagens e disparar o processo de treinamento. O primeiro estado, Train Model é encarregado de configurar a máquina virtual designada para realizar o treinamento do modelo. Nesse estágio, é selecionada uma máquina virtual com especificações adequadas para a tarefa de treinamento do modelo.

A Tabela 3.2.1 demonstra os parâmetros utilizados para instanciar a máquina virtual de treinamento. O parâmetro InstanceType significa a configuração de máquina virtual que será utilizada, o valor ml.c5.2xlarge resulta em uma máquina com 8 CPUs virtuais e 16GB de memória RAM. Esta configuração apresenta um equilíbrio entre performance e custo.

O parâmetro Model contém o modelo escolhido para realizar o treinamento. O valor train-pytorch-ic-mobilenet-v2 é utilizado para selecionar o modelo MobileNetV2 implementado na plataforma PyTorch para a tarefa de classificação.

A máquina virtual opera com base no conceito de contêiner, que representa um tipo específico de virtualização. Para configurá-la adequadamente, é essencial indicar uma imagem específica no parâmetro denominado TrainingImage, que contém todas as dependências necessárias para executar o treinamento do modelo. Nesse contexto, optou-se pela imagem pytorch-training:1.10.0-cpu-py38, uma vez que ela oferece o ambiente necessário para conduzir o treinamento do modelo de maneira eficaz. Também é necessário definir a estrutura de hardware no parâmetro InstanceType, e o valor adotado foi ml.c5.2xlarge, que significa uma máquina com 8 CPUs virtuais e 16GB de memória RAM. Esta configuração apresenta um equilíbrio entre performance e custo.

Parâmetro	**Valor**
InstanceType	ml.c5.2xlarge
Model	train-pytorch-ic-mobilenet-v2
TrainingImage	pytorch-training:1.10.0-cpu-py38

Tabela 3.2.1: Parâmetros da máquina virtual de treinamento.

Para garantir o funcionamento correto do modelo MobileNetV2, é necessário ajustar alguns hiperparâmetros específicos. A Tabela 3.2.2 lista os hiperparâmetros mais relevantes para este propósito. O hyperparâmetro epochs indica o número de épocas do processo de treinamento, esse número não precisa ser alto pois já está sendo utilizado um modelo pré treinado. O hyperparâmetro batch-size contém o número de imagens em cada lote de treinamento, adam-learning-rate é a taxa de aprendizado do otimizador Adam. O hyperparâmetro train-only-top-layer significa que somente a última camada do modelo será treinada e os pesos das camadas anteriores ficam congelados. Desta maneira é possível utilizar os pesos de uma rede pré treinada, e somente os pesos da última camada serão alterados para o conjunto de dados específico.

Hiperparâmetro	**Valor**
epochs	20
batch-size	4
adam-learning-rate	0.05
train-only-top-layer	True

Tabela 3.2.2: Hiperparâmetros de treinamento.

Todo processo de treinamento, incluindo modelos de aprendizagem de máquina e visão computacional estão presentes no módulo SageMaker da AWS.

No estado Register Model, o modelo é copiado e armazenado após a conclusão do treinamento. Cada modelo recebe uma identificação única, que é associada à data e hora do treinamento, juntamente com as métricas de acurácia obtidas.

Em seguida, o estado Convert Model to ONNX converte o modelo PyTorch para o formato ONNX, uma especificação aberta que representa modelos de aprendizado de máquina.

Por fim, o último estado, Update Training Status, atualiza a tabela com informações sobre o modelo treinado. Ele atribui uma identificação única, registra a hora de término do processo de treinamento e atualiza os links para acessar ou baixar o modelo treinado.

3.3 Interface Visual

O ambiente em nuvem da AWS é restrito aos desenvolvedores de sistemas e profissionais de TI (Tecnologia da Informação), qualquer alteração em ambiente de produção necessita passar por um fluxo específico de mudança.

Os usuários da aplicação efetuam o acesso por meio de uma interface simples e amigável para interagir com os dados. Foi desenvolvida uma página web que pode ser acessada à partir de qualquer navegador moderno, como Microsoft Edge ou Google Chrome. O desenvolvimento foi feito com a linguagem HTML (Hypertext Markup Language) e o framework Angular JS. Os dados são obtidos da plataforma em nuvem AWS através de uma API (Application Programming Interface), essa API expõe os dados para a página com todo controle e segurança necessários.

A Figura 3.3.1 demonstra a tela inicial do sistema onde é possivel criar novos projetos ou visualizar os projetos existentes.

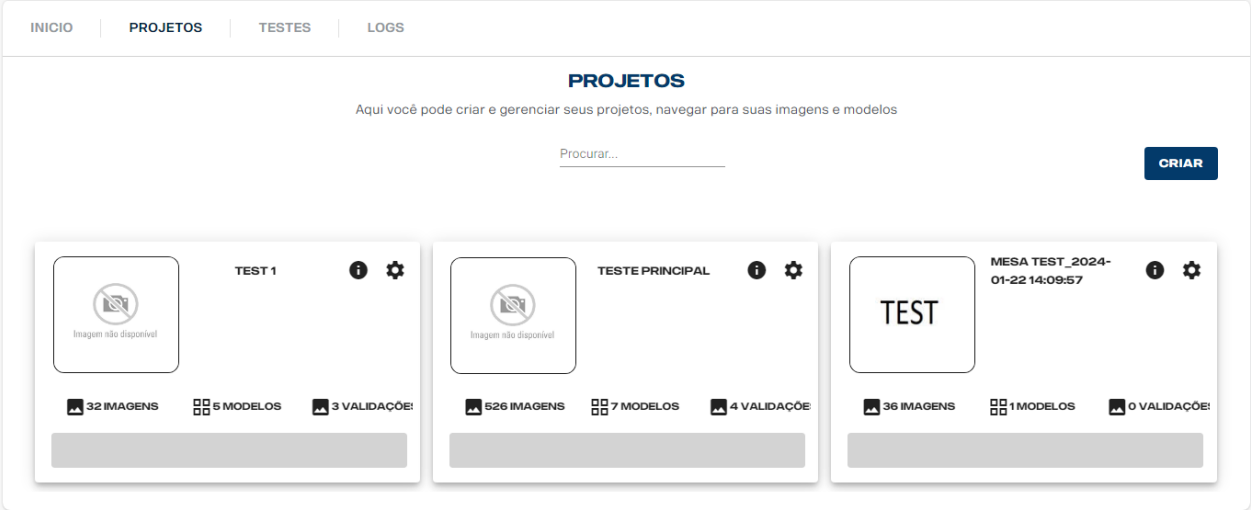

Figura 3.3.1: Interface visual com o usuário para gerenciamento de modelos e imagens

Para demonstração da interface visual do sistema foi criado o projeto chamado classificação livros, conforme indicado na Figura 3.3.2. O projeto contém 108 imagens divididas em 3 classes.

Figura 3.3.2: Visualização de um projeto no sistema.

A Figura 3.3.3 demonstra a primeira classe que contém 36 imagens do livro Deep Learning with Python.

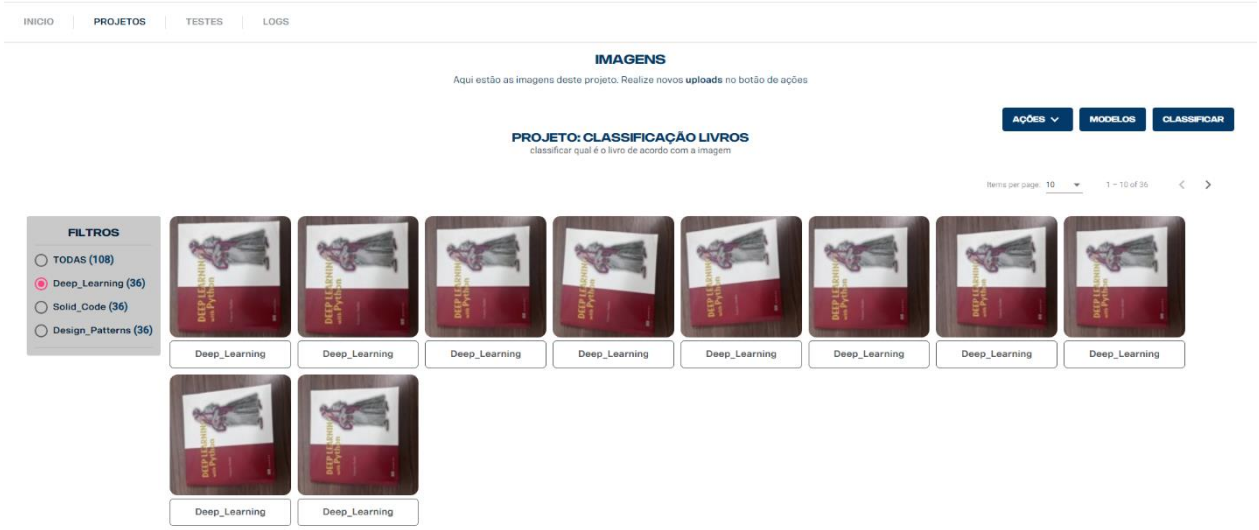

Figura 3.3.3: Imagens primeira classe, livro Deep Learning with Python.

A Figura 3.3.4 exibe a segunda classe que contém 36 imagens do livro Solid Code.

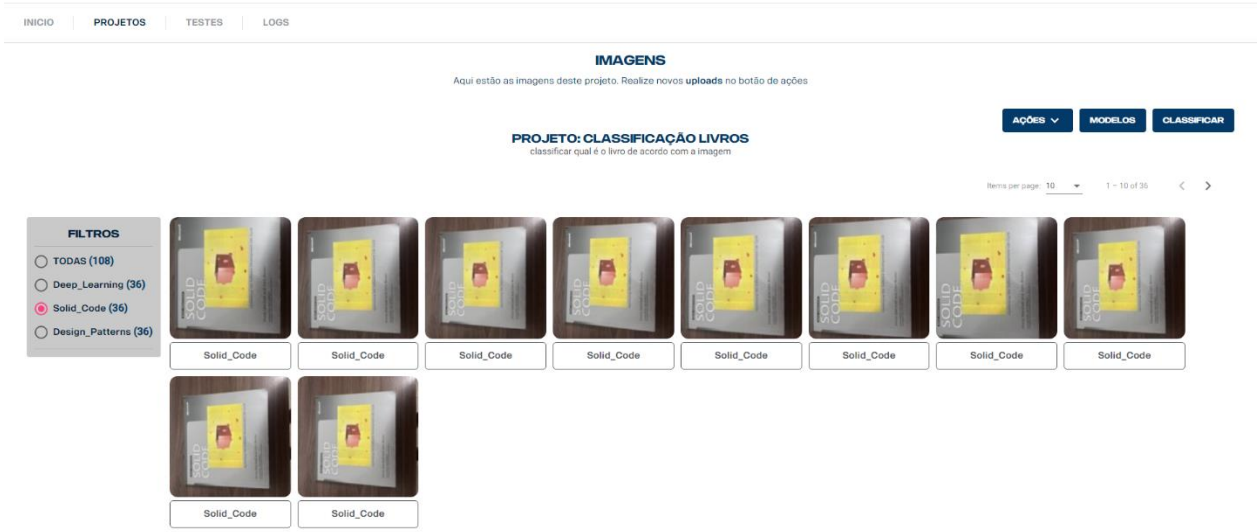

Figura 3.3.4: Imagens da segunda classe, livro Solid Code.

E por fim, a terceira classe contém 36 imagens do livro Design Patterns, indicada na Figura 3.3.5.

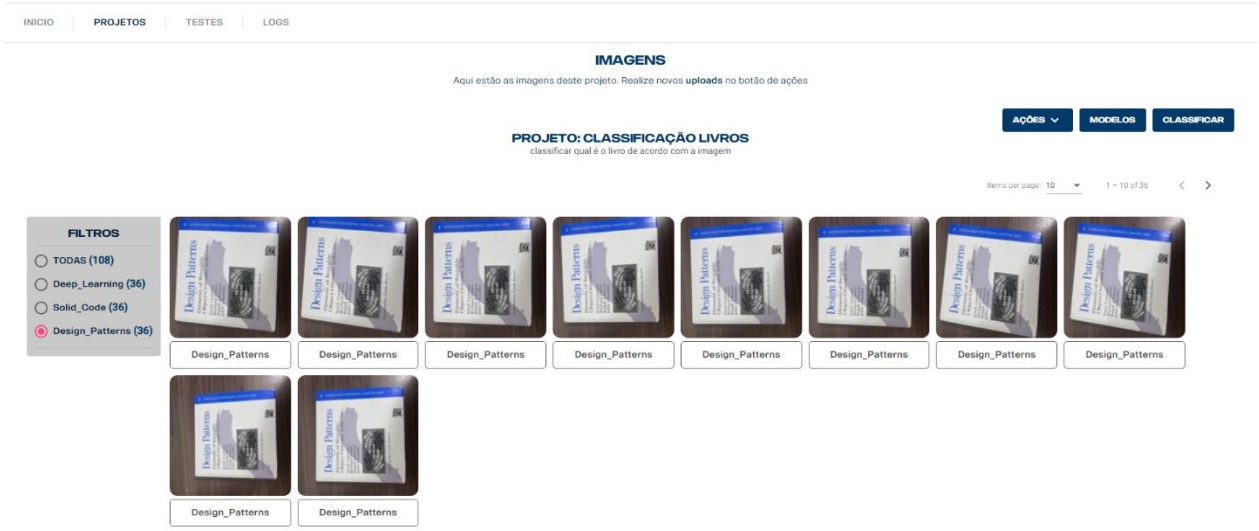

Figura 3.3.5: Imagens da terceira classe, livro Design Patterns.

Após carregar as imagens no projeto, e classificar as imagens nas classes correspondentes, deve-se gerar um novo modelo que será treinado com 80% do conjunto de imagens, reservando 20% das imagens para validação. A Figura 3.3.6 exibe informações sobre o resultado do processo de treinamento para o modelo gerado no projeto classificação livros.

Figura 3.3.6: Treinamento do modelo classificação livros

A Figura 3.3.7 demonstra a página utilizada para testes, onde o usuário pode carregar o modelo treinado e escolher uma imagem, assim a página irá retornar o resultado da classificação conforme o usuário configurou.

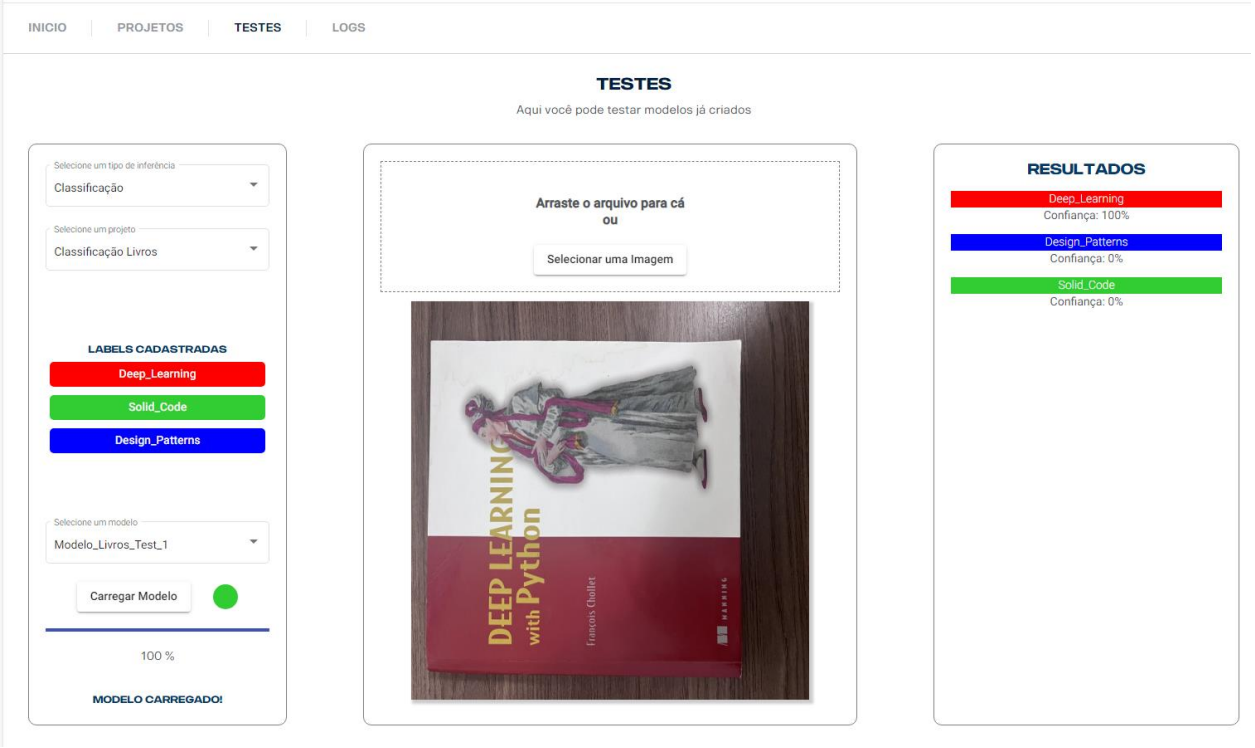

Figura 3.3.7: Interface visual para testar uma imagem com o modelo treinado

Capítulo 4

Resultados

4.1 Resultados

Após a publicação do sistema de gerenciamento de inspeções em ambiente de desenvolvimento, o mesmo já está disponível para utilização. Com o objetivo de demonstrar que não é necessário o conhecimento avançado em programação para criar e treinar um novo modelo, foi criado um projeto de teste e disponibilizado aos controladores de qualidade, que por sua vez, alimentaram o sistema com fotos reais. Após inserir as imagens e classificá-las como OK e NOT OK, o próprio usuário criou um modelo e efetuou o treinamento do mesmo.

No projeto de teste foram inseridas 526 imagens reais, onde 459 são da classe OK e 66 são da classe NOT OK, pode-se notar que as classes não estão balanceadas para o conjunto de imagens. O processo de treinamento foi realizado com 80% das imagens, e 20% das imagens foram utilizadas para validação. O tempo total de treinamento do modelo foi aproximadamente 9 minutos em uma máquina virtual com 8 CPUs e 16GB de memória RAM. A exatidão do modelo atingiu 96.9% com imagens reais de validação.

A Figura 4.1.1 demonstra a visualização do treinamento no sistema, com o nome e identificação do modelo, data de término e o valor de exatidão com dados de validação.

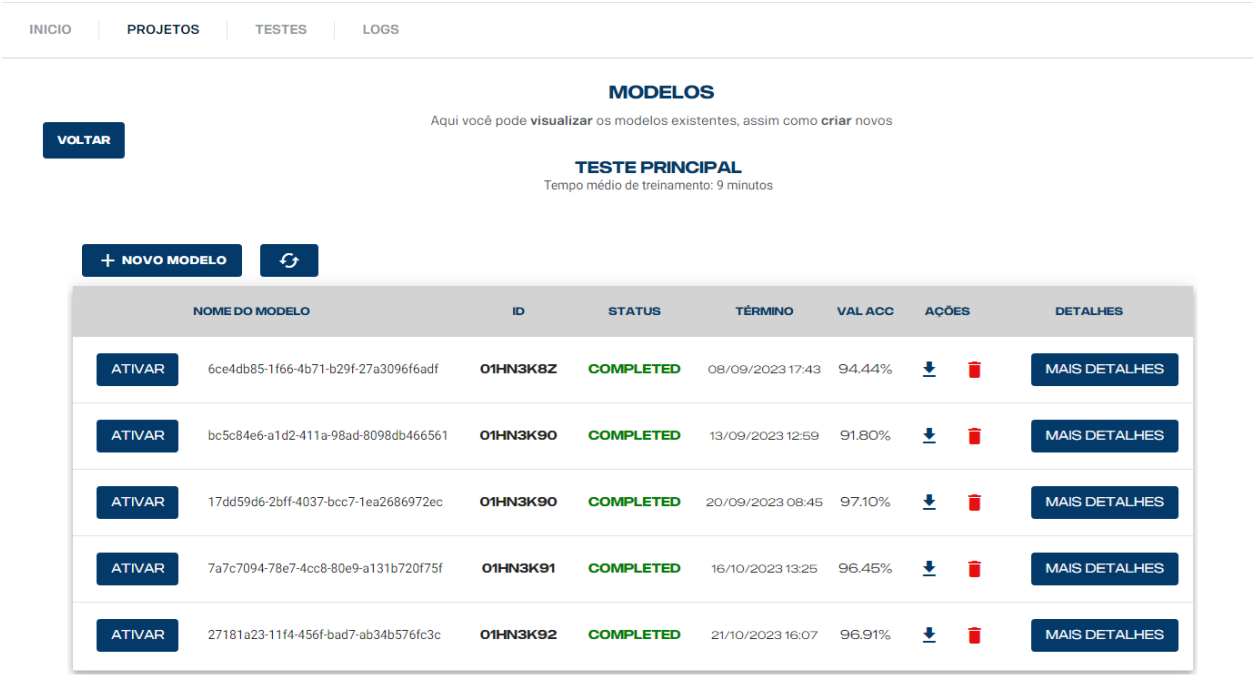

Figura 4.1.1: Resultado do treinamento no sistema

A Tabela 4.1.1 contém as métricas de custo e exatidão durante o treinamento para dados de treinamento e validação em cada época.

Época	Custo treinamento	Exatidão treinamento	Custo validação	Exatidão validação
0	4.669	0.8636	1.4516	0.9227
1	1.6972	0.9035	1.0394	0.9381
2	1.3002	0.9151	0.641	0.9588
3	1.4111	0.9163	0.4103	0.9485
4	1.0146	0.9356	1.0959	0.9485
5	0.9868	0.9266	0.8518	0.9433
6	1.5157	0.9266	0.554	0.9691
7	1.2445	0.9344	0.765	0.9536
8	1.1849	0.9279	1.4655	0.9278
9	1.21	0.9344	1.0625	0.9536
10	1.3216	0.9215	0.8701	0.9536
11	1.4141	0.9099	0.4714	0.9691
12	1.2811	0.9125	0.8257	0.9381
13	1.202	0.9266	0.4689	0.9588
14	1.263	0.9254	1.1104	0.9227
15	1.2469	0.9292	0.6828	0.9485
16	1.353	0.9176	0.5645	0.9639
17	1.4112	0.9099	0.5176	0.9691
18	1.5086	0.9163	0.7165	0.9588
19	1.2531	0.9318	0.7094	0.9588

Tabela 4.1.1: Parâmetros da máquina virtual de treinamento.

Os dados também podem ser visualizados na forma de gráficos, a Figura 4.1.2 demonstra o gráfico da função de custo por época, e a Figura 4.1.3 demonstra o gráfico da métrica de exatidão por época.

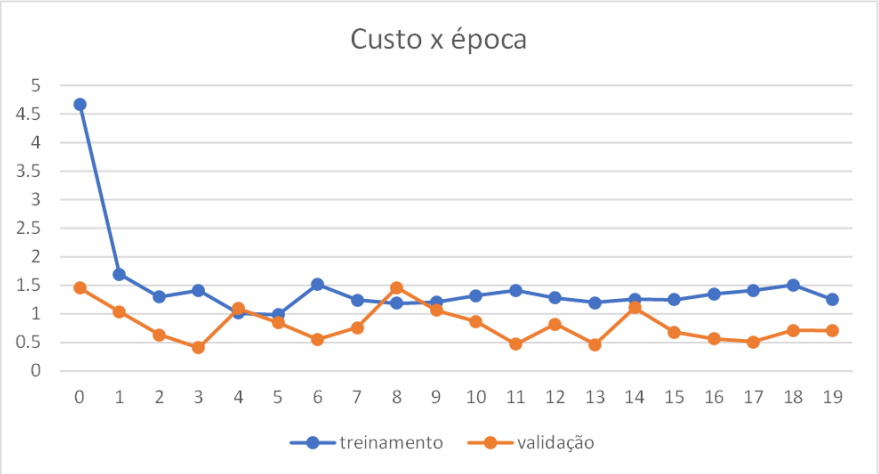

Figura 4.1.2: Gráfico da função de custo por época.

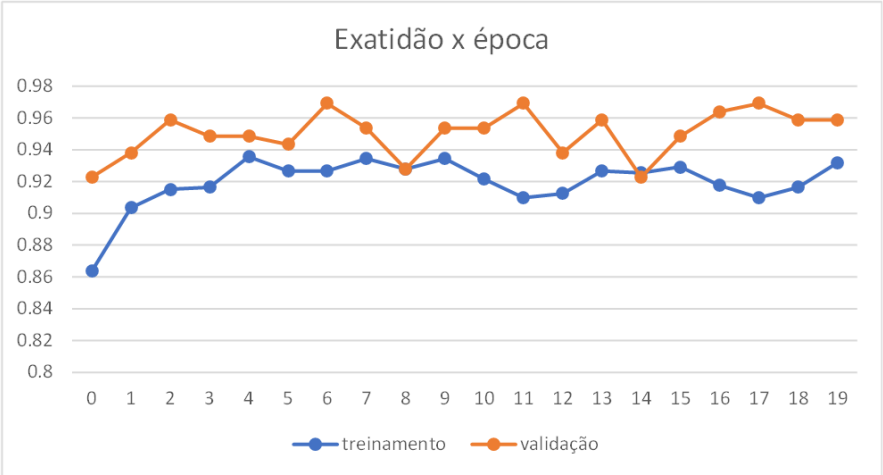

Figura 4.1.3: Gráfico da métrica de exatidão por época.

De acordo com os dados demonstrados o comportamento do treinamento foi balanceado, não apresentou subajuste (do inglês *underfitting*) ou sobreajuste (do inglês *overfitting).*

4.2 Discussão

Embora a exatidão do modelo com dados de treinamento foi excelente, é possível melhorar ainda mais incluindo uma etapa de data augmentation, para gerar novas imagens aplicando pequenas alterações como rotação, corte, espelhamento ou outras transformações nas imagens originais, aumentando assim a quantidade de dados.

Também é possível gerar novas imagens utilizando tecnologia de Inteligência Artificial Generativa.

A utilização do modelo treinado em tarefa de inspeção de qualidade foi realizada manualmente, e as fotos foram tiradas a partir de um dispositivo móvel. Porém é possível descarregar o modelo em câmeras industriais que aceitam modelos ONMX, ou efetuam conversão para o modelo específico da câmera. Desta maneira a câmera pode tirar fotos e fazer inferência automaticamente sendo disparada por algum sensor de presença, ou até mesmo embutir a câmera no braço de um robô. A vantagem de utilizar uma câmera fixa é obter sempre a mesma distância e ângulo da imagem, trazendo o foco sempre para o objeto alvo.

Também é possível enviar resultados para outros equipamentos ou sistemas através de APIs, protocolos de comunicação como OPC-UA ou publicar em um broker utilizando o protocolo MQTT. Os também modelos podem ser escolhidos automaticamente recebendo dados de sistemas ERP (Enterprise Resources Planning), MES (Manufacturing Executing systems) ou SCADA (Supervisory Control and Data Acquisition).

O método utilizado para gerenciar modelos e imagens também pode ser adaptado para outras tarefas complexas como detecção de objetos, onde pode-se detectar desde presença de pessoas ou animais em ambientes até objetos em caixas, caso muito comum na área logística. Também é possível incluir modelos específicos para detecção de anomalias.

Capítulo 5

Conclusão

O principal objetivo do trabalho era demonstrar o funcionamento de um sistema capaz de gerenciar modelos de visão computacional, que possa ser usado por qualquer profissional que possui o conhecimento para classificar imagens, sem a necessidade desenvolver scripts complexos.

O sistema desenvolvido possui uma interface simples, e tem a capacidade de gerenciar diversos projetos, cada um com seu conjunto de imagens e modelo próprio. Vale a pena destacar que foi utilizado todo o poder de computação da nuvem para armazenamento e treinamento das imagens, mas o modelo pode rodar em um dispositivo móvel na rede local, trazendo maior segurança e estabilidade no processo de verificação.

Desta maneira é possível automatizar uma inspeção visual utilizando aprendizado de máquina e visão computacional utilizando somente um dispositivo móvel. Os resultados demonstraram que o controlador da qualidade teve autonomia para gerenciar suas imagens e efetuar o treinamento do modelo, ainda conseguiu atingir um ótimo resultado com as imagens de validação.

Referências

ALMEIDA, Carlos Diego F.; LIMA, Jean Phelipe de O.; OLIVEIRA, João Victor M. de; OLIVEIRA, Raimundo Corrêa de; OLIVEIRA, Edson Farias. Arquitetura de sistema em nuvem para apoio à implantação de visão computacional em linhas de produção na Indústria 4.0. 2021. DOI: https://doi.org/10.20906/sbai.v1i1.2862

BURRA, Lakshmi Ramani; KARUNA, A; TUMMA, Srinivasarao; MARLAPALLI, Krishna; TUMULURU, Praveen. MobileNetV2-based Transfer Learning Model with Edge Computing for Automatic Fabric Defect Detection. 2023, Vol 82 No 1.
DOI: https://doi.org/10.56042/jsir.v82i1.69928

CAMELO, Leonardo Yuto Suzuki. Sistema de inspeção de etiquetas por visão computacional e aprendizado profundo na indústria 4.0, 2023.
url: http://repositorioinstitucional.uea.edu.br//handle/riuea/4770

GIRELLI, Cassio; CORSO, Leandro. Detecção de falhas em contentores plásticos utilizando Redes Neurais Convolucionais. SCIENTIA CUM INDUSTRIA, V. 8, N. 2, PP. 156 — 163, 2020
DOI: http://dx.doi.org/10.18226/23185279.v8iss2p156 .

ISUYAMA, Vivian Kimie; ALBERTINI, Bruno de Carvalho. Comparison of Convolutional Neural Network Models for Mobile Devices. *In*: WORKSHOP EM DESEMPENHO DE SISTEMAS COMPUTACIONAIS E DE COMUNICAÇÃO (WPERFORMANCE). 2021, Sociedade Brasileira de Computação, p. 73-83. ISSN 2595-6167.
DOI: https://doi.org/10.5753/wperformance.2021.15724.

KE, Dong; CHENGJIE, Zhou; YIHAN, Ruan; YUZHI, Li. MobileNetV2 Model for Image Classification. 2020. DOI: https://doi.org/10.1109/ITCA52113.2020.00106

ZHOU, Longfei; ZHANG, Lin; KONZ, Nicholas. Computer Vision Techniques in Manufacturing. 2023 in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 53, no. 1, pp. 105-117, DOI: https://doi.org/10.1109/TSMC.2022.3166397

MAUAD, Hugo Ferreira; GUILHERMINO Neto, Guilherme. Deep Learning para inspeção visual em peças fundidas, 2023. url: https://repositorio.ifes.edu.br/handle/123456789/3272

MIJAILOVIĆ, Đorđe; DJORDJEVIC, Aleksandar; STEFANOVIC, Miladin; ERIĆ, Milan. Quality control in the manufacturing industry based on the application of computer vision. 2023. ISBN 978-86-6335-104-2. url: https://scidar.kg.ac.rs/handle/123456789/18388

MUSSATO, André. Revolução 4.0 e a educação profissional. 2018.
url: https://www.fatecjales.edu.br/publicacoes/fatecnologia/474-revolucao-4-0-e-a-educacao-profissional

RODRIGUES, João Victor. Classificação de peças mecânicas a partir de visão computacional e aprendizado de máquina utilizando imagens sintéticas. 2020.
url: http://hdl.handle.net/10183/217435

SANTOS, Erick Erate dos. Automatização de um processo de inspeção visual multiclasse com o uso de um sistema de visão computacional baseado em deep learning. 2022.
url: https://repositorio.uninter.com/handle/1/1388