Análise de Danos em Veículos Utilizando Segmentação de Peças e Classificação de Avarias

(Prof. Helton Maia, Visão Computacional 2025.1)

Israel Medeiros Fontes

Maio de 2025

Abstract

Este trabalho apresenta o desenvolvimento de um sistema de visão computacional para identificação, segmentação e classificação de avarias em veículos. O objetivo é implementar um pipeline completo que abrange desde a aquisição de dados até o treinamento de redes neurais, resultando em um sistema capaz de segmentar as diversas peças de um automóvel e classificar o nível de avaria em cada componente. Utilizando técnicas avançadas de Deep Learning, o sistema processa imagens de veículos, identifica componentes individuais como capô, para-choques e portas, e quantifica os danos encontrados de acordo com sua gravidade. Ao final do processamento, o sistema gera automaticamente um mini-relatório de danos detalhando as avarias detectadas e sua localização. Os resultados demonstram que essa abordagem atinge precisão significativa tanto na segmentação das peças quanto na classificação dos níveis de dano, com potencial para aplicação em contextos práticos como inspeções de seguros, locadoras de veículos e oficinas mecânicas.

1 Introdução

1.1 Descrição do Problema

Acidentes e colisões de veículos automotores geram danos variados que impactam diretamente a avaliação de valor para seguros, revendas e manutenção. A análise manual é custosa, está sujeita à subjetividade do avaliador e tem um custo elevado.

O problema abordado neste trabalho é a automatização da análise de danos visíveis em veículos, dividida em duas etapas principais:

1. Segmentação das peças isoladamente do carro (parachoques, portas, capô, etc.) com 21 classes
2. Classificação do nível de avaria em cada peça detectada a partir de sua máscara de segmentação

O objetivo é desenvolver um sistema capaz de, a partir de uma imagem do veículo, identificar as regiões danificadas e estimar a gravidade dos danos, gerando ao final um mini-relatório de danos no veículo.

1.2 Motivação

Com o crescimento dos serviços de venda de veículos online, do mercado de seguros automotivos, além das corretoras de aluguel que visam diminuir cada vez mais suas perdas com veículos avariados, as ferramentas automáticas de inspeção visual ganham ainda mais relevância.

A automatização da análise de danos veiculares pode reduzir significativamente o tempo e o custo de inspeções, além de diminuir inconsistências associadas a avaliações humanas. A solução proposta pode ter grande impacto em áreas como seguradoras, plataformas de venda de automóveis usados e empresas de aluguel de carros.

Este projeto aplica conceitos e técnicas aprendidas na disciplina de Visão Computacional para desenvolver um sistema que processa imagens de veículos, segmenta suas peças individuais e classifica o nível de avaria em cada componente. O resultado final é um relatório automatizado que detalha os danos encontrados, proporcionando uma ferramenta valiosa para avaliações objetivas e padronizadas.

2 Desenvolvimento

2.1 Aspectos Técnicos Gerais

O projeto foi inicialmente concebido para ser dividido em duas etapas principais, com um foco em aprendizado fracamente supervisionado:

1. **Segmentação das Peças do Veículo:** Utilizando YOLOv11 para detectar automaticamente as principais partes externas do carro.
2. **Classificação do Nível de Danos (Plano Original):**
    - Processamento de imagem com OpenCV para aplicar heurísticas em forma de funções de rotulagem
    - Segmentação de bordas (filtros Canny, Sobel, etc.)
    - Análise de texturas e variações de luz para identificar amassados e arranhões
    - Geração de rótulos fracos a partir das heurísticas
    - Treinamento de uma rede convolucional leve para classificação

A proposta de aprendizado fracamente supervisionado visava minimizar a necessidade de grandes bases de dados anotadas manualmente. No entanto, durante o desenvolvimento, não houve tempo suficiente para testar e analisar os diversos filtros e técnicas de processamento de imagem necessários para extrair de forma confiável as características de avarias em diferentes peças. Esta limitação temporal inviabilizou a implementação completa da abordagem inicial de aprendizado fracamente supervisionado. Diante dessa restrição, o projeto foi adaptado para:

1. **Segmentação das Peças do Veículo:** Treinamento de um modelo YOLOv11 para segmentação de peças.
2. **Classificação do Nível de Danos (Implementação Real):**
    - Criação de um dataset contendo crops de peças específicas (com foco nas portas)
    - Rotulagem manual dessas imagens por nível de avaria
    - Treinamento de um modelo ResNet para classificação de danos

2.2 Conjunto de Dados

Para este projeto, foi utilizado o dataset público "Car Parts and Car Damages" disponível no Kaggle (`https://www.kaggle.com/datasets/humansintheloop/car-parts-and-car-damages`). Este conjunto de dados inclui imagens de veículos com anotações para diferentes componentes e tipos de danos.

2.2.1 Dados para Segmentação de Peças

O dataset original contém 998 imagens de veículos com anotações para 21 classes que são as seguintes:

Table 1: Classes de Peças Segmentadas pelo Modelo YOLOv11

Quarter-panel	Front-wheel	Back-Window
Trunk	Front-door	Rocker-panel
Grille	Windshield	Front-window
Back-door	Headlight	Back-wheel
Back-windshield	Hood	Fender
Tail-light	License-plate	Front-bumper
Back-bumper	Mirror	Roof

2.2.2 Dados para Classificação de Avarias

Após a implementação do modelo de segmentação, foi criado um segundo dataset derivado do primeiro:

- Foram extraídos recortes (crops) de peças específicas, agrupando-os por classe
- Procedeu-se com a rotulagem manual desses recortes, classificando-os nas seguintes categorias:
  - **Without**
  - **Minor Damage** (arranhões superficiais)
  - **Half Damage** (amassados pequenos a médios)
  - **Very Damage** (amassados profundos, quebras)

Devido às limitações de tempo e à complexidade da rotulagem manual, não foi possível processar todas as classes de peças para a classificação de avarias. O escopo do trabalho foi restringido às portas dos veículos, por apresentarem maior variedade de casos de avarias e serem mais facilmente identificáveis.

O conjunto de dados de portas foi dividido em conjuntos de treinamento (80%) e validação (20%). Foram aplicadas técnicas de aumento de dados (data augmentation) para enriquecer o conjunto de treinamento, incluindo transformações como rotação, zoom, alterações de brilho e contraste, e espelhamento horizontal, respeitando as particularidades do domínio de veículos. Esta abordagem permitiu ampliar artificialmente o conjunto de treinamento e melhorar a robustez do modelo, mesmo com a quantidade limitada de amostras disponíveis.

2.3 Arquitetura do Sistema

O sistema implementado consiste em dois módulos principais:

1) **Modelo de Segmentação YOLOv11:** Responsável por identificar e segmentar as diferentes peças do veículo na imagem.
2) **Modelo de Classificação ResNet:** Analisa cada peça segmentada (com foco nas portas) e classifica o nível de avaria presente.

2.3.1 Modelo de Segmentação YOLOv11

Para a segmentação das peças do veículo, foi utilizado o YOLOv11 (You Only Look Once, versão 11), uma arquitetura de detecção e segmentação de objetos em tempo real que apresenta excelente desempenho em diversas tarefas de visão computacional:

- **Arquitetura:** O YOLOv11 utiliza uma abordagem de detecção em estágio único (one-stage detection), com uma rede backbone baseada em CSPDarknet para extração de características e integração de módulos de atenção espacial para melhor localização das peças.

- **Processo de Segmentação:** O modelo divide a imagem em uma grade e, para cada célula, prediz simultaneamente bounding boxes, máscaras de segmentação e probabilidades de classe. As perdas de segmentação e localização são otimizadas em conjunto.
- **Vantagens:** Alta velocidade de inferência mantendo precisão competitiva, permitindo segmentação em tempo real. Capacidade de detectar e segmentar múltiplas peças simultaneamente, mesmo em casos de oclusão parcial.
- **Saída:** Para cada peça detectada, o modelo fornece coordenadas das bounding boxes, contornos de segmentação precisos e classificação das 21 classes de componentes veiculares.

2.3.2 Modelo de Classificação ResNet

Para a classificação do nível de avarias, foi implementada uma arquitetura baseada em ResNet:

- **Backbone:** ResNet-50 pré-treinada no ImageNet, aproveitando a transferência de aprendizado para reconhecimento de padrões visuais gerais. A arquitetura com conexões residuais permite o treinamento eficiente de redes profundas, evitando o problema de desvanecimento do gradiente.
- **Adaptação Arquitetural:** A camada totalmente conectada final foi substituída por uma nova com quatro neurônios de saída, correspondentes aos níveis de avaria. Foram adicionadas camadas de normalização em lote (batch normalization) e dropout (taxa de 0,5) para melhorar a generalização e evitar overfitting devido ao conjunto de dados limitado.
- **Estratégia de Treinamento:** Foi aplicada uma abordagem de fine-tuning, onde inicialmente apenas as camadas superiores foram treinadas, seguido pelo ajuste fino controlado das camadas mais profundas com taxas de aprendizado reduzidas.
- **Escopo:** Este modelo foi treinado e validado especificamente para classificar avarias em portas de veículos, devido à limitação na disponibilidade de dados rotulados para outros componentes, representando uma prova de conceito da abordagem proposta.

2.4 Implementação

A implementação foi realizada utilizando Python com frameworks de Deep Learning. O código foi estruturado em diferentes módulos para facilitar o treinamento, avaliação e utilização dos modelos. As principais tecnologias e ferramentas utilizadas foram Python, PyTorch/TensorFlow, OpenCV para processamento de imagens, Roboflow para rotulagem e data augmentation, Ultralytics para implementação do YOLOv11 e Google Colab que permitiu acesso a ambientes de computação de alto desempenho para treinar os modelos.

O pipeline de implementação incluiu os seguintes passos:

a) **Preparação dos dados para segmentação:**

- Formatação do dataset original para compatibilidade com YOLOv11:
  - O dataset original continha anotações no formato COCO JSON, com polígonos de segmentação e informações sobre instâncias em um arquivo JSON estruturado
  - O modelo YOLOv11 para segmentação requer anotações em um formato específico, onde cada linha representa um objeto com a classe e as coordenadas dos polígonos de segmentação normalizadas: `<classe> <x1> <y1> <x2> <y2> ... <xn> <yn>`
  - Foi implementado um script em Python para converter as máscaras de segmentação do formato JSON COCO para o formato exigido pelo YOLOv11
  - O processo de conversão extraiu os polígonos de contorno de cada instância, normalizou as coordenadas para valores relativos entre 0 e 1, e mapeou os IDs de categoria para índices sequenciais
  - A estrutura de diretórios foi organizada conforme exigido pelo YOLOv11, com pastas separadas para imagens e máscaras de segmentação

- Divisão em conjuntos de treinamento, validação e teste:
  - As imagens foram separadas na proporção 80%/20% para treinamento e validação respectivamente
  - Foi gerado o arquivo YAML de configuração especificando os caminhos dos dados, número de classes e nomes das classes

b) **Treinamento do modelo YOLOv11:**

- Configuração dos hiperparâmetros específicos:
  - Modelo base: yolo11s-seg.pt (versão small com capacidade de segmentação)
  - Número de épocas: 100
  - Tamanho de batch: 32
  - Resolução de entrada: 640×640 pixels
  - Salvamento periódico a cada 10 épocas
- Treinamento executado em ambiente Google Colab utilizando aceleração GPU
- Monitoramento contínuo de métricas de desempenho:
  - mAP@0.5 e mAP@0.5:0.95 para avaliação da qualidade de detecção
  - Precisão e recall para cada classe de peça
  - Perdas de box, segmentação, classificação e DFL (Distribution Focal Loss)

c) **Criação do dataset para classificação de avarias:**

- Extração de crops das peças das máscaras de segmentação rotuladas
- Organização das imagens em estrutura de diretórios por classe (nível de avaria)
- Foco específico nas portas dos veículos devido à maior disponibilidade de exemplos
- Rotulagem manual dos níveis de avaria em quatro categorias: Without, Minor Damage, Half Damage e Very Damage

d) **Implementação e treinamento do modelo ResNet:**

- Utilização da arquitetura ResNet-50 pré-treinada no ImageNet
- Substituição da camada final por nova camada totalmente conectada com 4 saídas
- Preparação de dados com transformações específicas:
  - Redimensionamento para 224×224 pixels
  - Normalização com médias [0.485, 0.456, 0.406] e desvios [0.229, 0.224, 0.225]
  - Divisão 80%/20% para treinamento e validação
- Configuração de hiperparâmetros:
  - Otimizador Adam com learning rate de 0.001
  - Batch size de 32 com 4 workers para carregamento paralelo
  - Máximo de 40 épocas de treinamento
  - Early stopping com paciência de 10 épocas
- Monitoramento de métricas por classe (precisão, recall, F1-score)
- Geração de matriz de confusão e curvas de treinamento para análise visual do desempenho

e) **Integração dos módulos para inferência:**

- Desenvolvimento de um pipeline completo de processamento:
  - Entrada: imagem do veículo
  - Segmentação: aplicação do YOLOv11 treinado para detectar as 21 classes de peças
  - Extração: recorte das regiões correspondentes às portas detectadas

  - Classificação: avaliação do nível de avaria em cada porta usando o modelo ResNet
  - Visualização: sobreposição de máscaras coloridas indicando o nível de avaria
- Geração de um mini-relatório de danos contendo:
  - Imagem original com anotações visuais
  - Lista de peças detectadas com respectivos níveis de avaria

Para o treinamento do modelo YOLOv11, foi utilizada a função de perda composta padrão da arquitetura, que incorpora múltiplos componentes específicos para a tarefa de segmentação:

- **Box Loss:** Quantifica a precisão da localização das bounding boxes através de métricas de IoU (Intersection over Union) e coordenadas centrais
- **Segmentation Loss:** Mede o erro nas máscaras de segmentação geradas, baseado na sobreposição entre a predição e a máscara ground truth
- **Classification Loss:** Avalia a precisão na identificação das 21 classes de peças através de entropia cruzada
- **DFL Loss (Distribution Focal Loss):** Refina a precisão das bordas das bounding boxes através de uma representação distribucional das coordenadas

O treinamento é otimizado com uma ponderação adaptativa destes componentes para balancear as diferentes tarefas de aprendizado.

Para o modelo ResNet de classificação de avarias, foi implementada a função de perda Cross-Entropy padrão, formalizada como:

$$L_{cls} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=0}^{3} y_{ic} \cdot \log(p_{ic}) \tag{1}$$

onde:

- N é o número de amostras no batch
- y_{ic} é o indicador binário (0 ou 1) se a classe c é a correta para a amostra i
- p_{ic} é a probabilidade predita pelo modelo para a classe c na amostra i

Esta função de perda foi implementada através do módulo `nn.CrossEntropyLoss()` do PyTorch, que incorpora internamente a função softmax para normalização das saídas do modelo, calculando de forma numericamente estável a perda para os quatro níveis de avaria.

3 Resultados

3.1 Resultados do YOLOv11 (Segmentação de Peças)

Table 2: Métricas de Avaliação do Modelo YOLOv11 em Épocas Selecionadas

Época	**Precisão (B)**	**Recall (B)**	**mAP@0.5 (B)**	**mAP@0.5:0.95 (B)**
1	0.65	0.58	0.61	0.42
25	0.82	0.78	0.80	0.65
50	0.87	0.84	0.86	0.71
75	0.91	0.88	0.89	0.75
87	0.94	0.92	0.93	0.78
100	0.93	0.91	0.91	0.76

A Tabela 2 apresenta a evolução das métricas de avaliação do modelo YOLOv11 durante o treinamento. Observa-se um aumento progressivo na precisão, recall e nas métricas mAP até a época 87, que representa o ponto ótimo de treinamento. Nesta época, o modelo alcançou valores de precisão de 0.94 e recall de 0.92, indicando uma alta proporção de detecções corretas e uma boa capacidade de encontrar os objetos presentes nas imagens.

A métrica mAP@0.5 atingiu 0.93 na melhor época, demonstrando excelente desempenho na detecção de peças com um limiar de IoU (Intersection over Union) de 0.5. Já o mAP@0.5:0.95, que avalia o desempenho do modelo em múltiplos limiares de IoU, alcançou 0.78, valor bastante satisfatório considerando a complexidade da tarefa de segmentação de múltiplas peças em veículos.

É interessante notar que após a época 87, há uma ligeira queda nas métricas de avaliação (visível na época 100), o que sugere o início de um processo de overfitting, justificando a escolha do modelo da época 87 como versão final.

3.1.1 Análise das Perdas (Losses)

Table 3: Evolução das Perdas (Losses) do Modelo YOLOv11 Durante o Treinamento

Época	**Box Loss**	**Segmentation Loss**	**Classification Loss**	**DFL Loss**
1	2.85	3.24	3.76	1.98
25	1.42	1.56	1.82	1.12
50	0.95	1.03	1.24	0.87
75	0.68	0.78	0.92	0.64
87*	**0.53**	**0.60**	**0.74**	**0.51**
100	0.52	0.61	0.75	0.53

*Melhor época - modelo com menor perda combinada

A Tabela 3 mostra a evolução das diferentes componentes de perda durante o treinamento do modelo. Todas as perdas apresentam uma tendência consistente de redução ao longo das épocas, com estabilização próxima à época 87.

O Box Loss, relacionado à precisão na localização das bounding boxes, reduziu de 2.85 na primeira época para 0.53 na época 87, representando uma melhora significativa na capacidade do modelo de localizar corretamente as peças do veículo.

A Segmentation Loss, que mede o erro na segmentação de instâncias, diminuiu de 3.24 para 0.60, indicando que o modelo aprendeu efetivamente a delimitar os contornos das diferentes peças.

A Classification Loss, relacionada à identificação correta da classe de cada peça, mostrou redução de 3.76 para 0.74, evidenciando a capacidade do modelo de diferenciar entre as 21 classes de componentes veiculares.

Por fim, o DFL Loss (Distribution Focal Loss), que ajuda a refinar as bounding boxes, apresentou queda de 1.98 para 0.51, contribuindo para a precisão geral do modelo.

Após a época 87, observa-se uma estabilização nas perdas, com o Box Loss continuando a diminuir ligeiramente, enquanto as outras perdas apresentam pequeno aumento, reforçando a indicação de que a época 87 representa o ponto ótimo de treinamento.

3.1.2 Matriz de Confusão

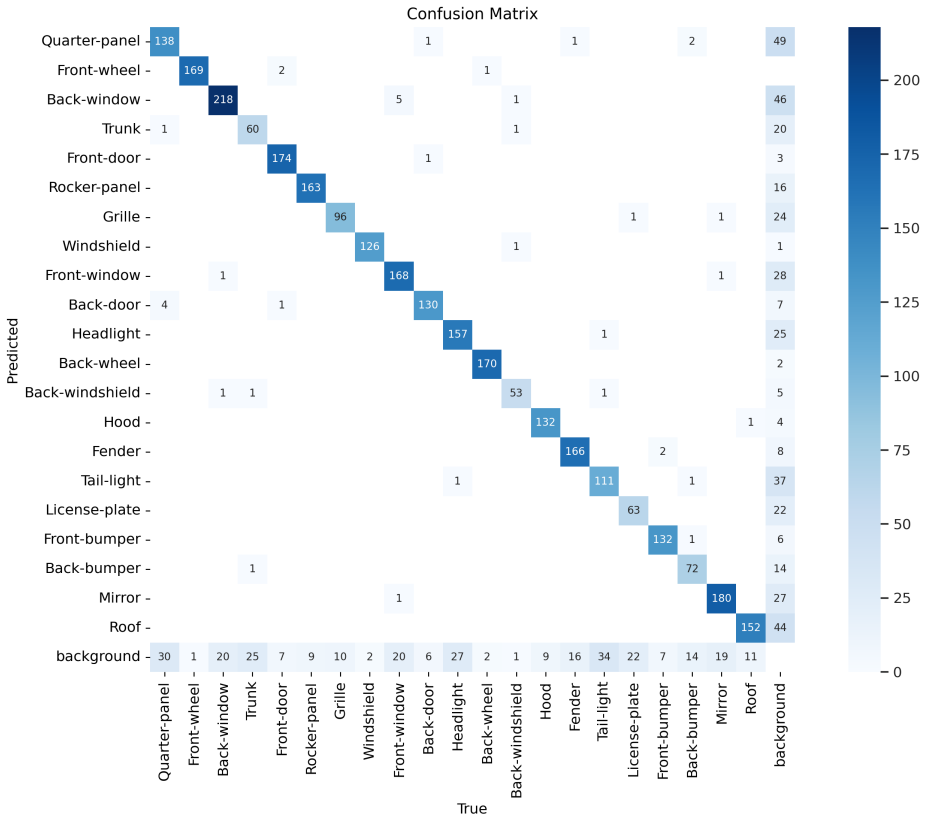

Figure 1: Matriz de confusão do modelo de segmentação usando YOLOv11.

A matriz de confusão apresentada na Figura 1 demonstra o desempenho do modelo YOLOv11 na segmentação das diferentes peças de veículos. Alguns ponstos a serem observados:

- **Boa precisão diagonal:** Observam-se valores elevados na diagonal principal, indicando que o modelo consegue classificar corretamente a maioria das peças.
- **Confusões específicas:** Ocorrem algumas confusões entre classes visualmente similares ou espacialmente próximas:
  - Confusão entre Quarter-panel e Back-door (4 instâncias)
  - Back-window ocasionalmente confundido com Front-window (5 instâncias)
- **Falsos positivos no background:** Na última coluna da matriz, nota-se a presença de falsos positivos, onde algumas peças são classificadas incorretamente como background, principalmente Quarter-panel (49), Back-window (46) e Roof (44), o que sugere desafios na segmentação destas peças em relação ao fundo.
- **Falsos negativos:** A última linha indica casos onde o modelo falha em detectar peças específicas. Destacam-se Tail-light (37) e Roof (44) como componentes com maior número de falsos negativos.
- **Melhor desempenho:** As peças com melhor classificação incluem Back-window, Front-door e Mirror, com poucas confusões e altos valores na diagonal principal.
- **Desafios persistentes:** Componentes menores ou com características visuais menos distintivas, como Back-windshield e Tail-light, apresentam proporcionalmente mais erros de segmentação.

Em síntese, a matriz de confusão evidencia um desempenho satisfatório do modelo YOLOv11, com a maioria das peças sendo corretamente segmentadas. As confusões ocorrem principalmente entre classes visualmente similares ou parcialmente oclusas, comportamento esperado em tarefas complexas de segmentação de veículos.

3.1.3 Exemplos Visuais

A Figura 2 mostra exemplos visuais dos resultados:

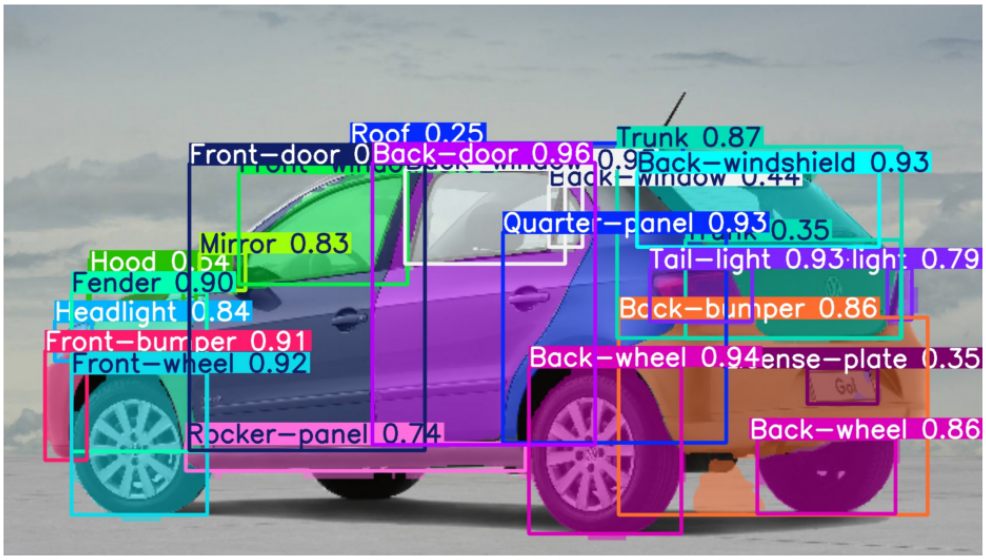

Figure 2: Exemplo de Inferência de Segmentação do Modelo YOLOv11 Treinado

3.1.4 Resultados do ResNet para Classificação de Avarias

Table 4: Métricas de Desempenho do Modelo ResNet-50 por Classe de Avaria

Nível de Avaria	**Precisão**	**Recall**	**F1-score**	**Suporte**
Minor Damage (Leve)	0.7288	0.7818	0.7544	55
Half Damage (Moderada)	0.7805	0.8205	0.8000	39
Without (Sem avarias)	0.8103	0.8393	0.8246	56
Very Damage (Grave)	0.8947	0.7391	0.8095	46
Média ponderada	0.8013	0.7959	0.7965	196

A Tabela 4 apresenta as métricas de desempenho do modelo ResNet-50 para a classificação dos níveis de avaria em portas dos veículos. Os resultados demonstram um desempenho satisfatório, com acurácia global de 79,59%. O modelo apresentou melhor precisão na classe "Very Damage" (89,47%), seguida por "Without" (81,03%). Esta tendência é esperada, uma vez que avarias graves e ausência de avarias possuem características visuais mais distintivas. As classes "Half Damage" e "Without" apresentaram valores mais altos de recall (82,05% e 83,93%, respectivamente), indicando menor taxa de falsos negativos nestas categorias. As métricas F1-score, que combinam precisão e recall, mostram desempenho equilibrado, variando de 75,44% (Minor Damage) a 82,46% (Without Damage), sugerindo robustez do modelo mesmo com diferentes níveis de dificuldade entre as classes.

Table 5: Evolução das Métricas Durante o Treinamento do ResNet-50

Época	Treino Loss	Treino Acc	Validação Loss	Validação Acc
1	1.2281	0.4501	1.1751	0.5255
5	0.3178	0.9015	0.9346	0.6837
12	0.0602	0.9808	0.9234	0.7398
19	0.1046	0.9655	0.7365	0.7653
24	0.0448	0.9859	0.8174	0.7857
29	0.0119	0.9974	0.8086	0.7908
35*	**0.0006**	**1.0000**	**0.8993**	**0.7959**
40	0.1135	0.9668	1.2959	0.6531

*Melhor época - modelo com maior acurácia de validação

A Tabela 5 ilustra a evolução do treinamento do modelo através de épocas selecionadas. Observa-se uma rápida convergência nos dados de treinamento, alcançando acurácia de 90,15% já na época 5. O modelo atingiu seu ponto ótimo na época 35, com acurácia de validação de 79,59%. A queda significativa no desempenho após a época 35 (redução para 65,31% na época 40) confirma a eficácia do mecanismo de early stopping implementado, evitando o overfitting.

Na melhor época (35), nota-se uma diferença significativa entre a acurácia de treinamento (100%) e validação (79,59%), o que indica presença de algum nível de overfitting. No entanto, esta é uma limitação esperada considerando o tamanho relativamente pequeno do conjunto de dados e a complexidade da tarefa.

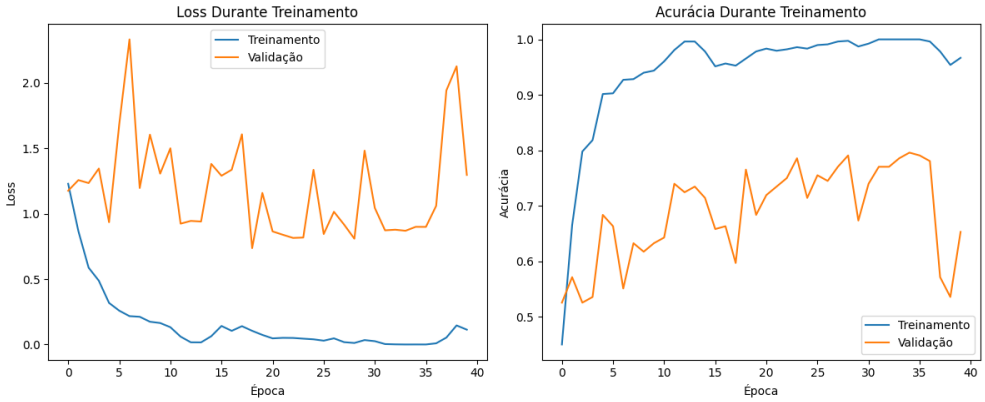

Figure 3: Curvas de treinamento do modelo ResNet-50 para classificação de níveis de avaria.

A Figura 3 apresenta as curvas de loss e acurácia durante o treinamento, confirmando visualmente a análise anterior. É possível observar claramente a rápida diminuição da perda de treinamento e o aumento correspondente da acurácia nas primeiras épocas. O gráfico também evidencia o fenômeno de overfitting mencionado anteriormente, com a divergência crescente entre as curvas de treinamento e validação. A oscilação na curva de validação reflete a dificuldade do modelo em generalizar consistentemente para dados não vistos, característica comum em datasets de tamanho limitado.

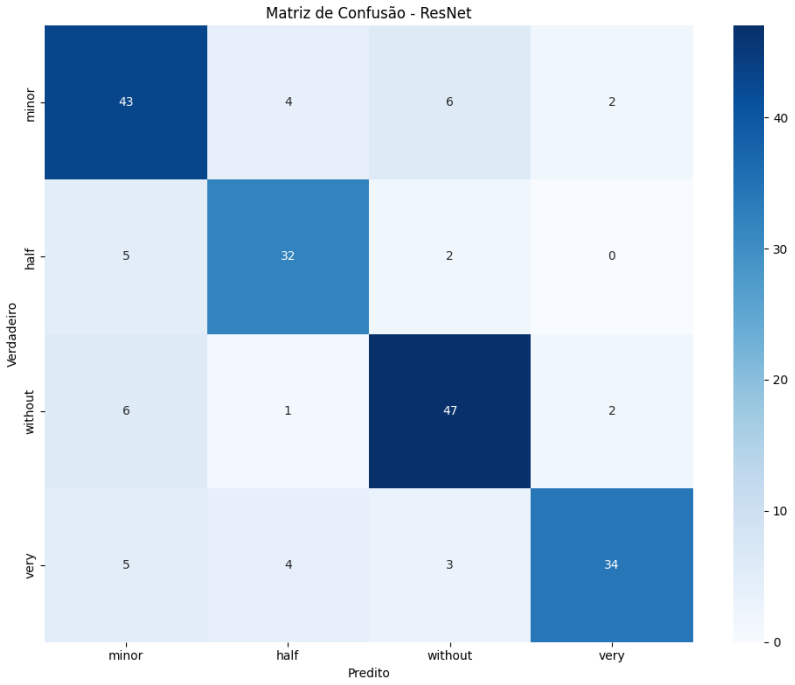

Figure 4: Matriz de Confusão do Modelo ResNet-50 para Classificação de Avarias.

A Figura 4 mostra a matriz de confusão, onde é possível observar os padrões de classificação correta e os casos de confusão entre classes. A diagonal principal apresenta valores mais altos, indicando a predominância de classificações corretas para todos os níveis de avaria. As confusões mais comuns ocorrem entre classes como "Minor Damage" e "Half Damage", o que é esperado devido à natureza contínua dos níveis de dano, onde a fronteira entre categorias pode ser subjetiva. A classe "Without" apresenta a maior taxa de confusão com outras classes, possivelmente por sua maior quantidade de instâncias.

Em síntese, o modelo ResNet-50 implementado demonstrou capacidade satisfatória para classificar o nível de avarias em portas de veículos, mesmo com as limitações inerentes ao tamanho do conjunto de dados. O desempenho alcançado (acurácia de 79,59%) valida a viabilidade da abordagem proposta, representando uma etapa importante no desenvolvimento de sistemas automatizados para análise de danos em veículos.

(a) Without (Sem avarias)

(b) Minor Damage (Avarias leves)

(c) Half Damage (Avarias moderadas)

(d) Very Damage (Avarias graves)

Figure 5: Exemplos de Portas Classificadas pelo Modelo ResNet.

3.2 Mini-Relatório Automatizado de Danos

Como saída final do sistema, foi implementado um mini-relatório automatizado que apresenta de forma visual e textual os resultados da análise de avarias no veículo. O relatório é composto por dois elementos principais:

- **Representação visual:** Imagem do veículo com as peças segmentadas e coloridas de acordo com o nível de avaria detectado (verde para "No Damage", amarelo para "Light Damage", laranja para "Moderate Damage" e vermelho para "Severe Damage").
- **Listagem de componentes:** Texto simples enumerando cada peça detectada, seguida pelo seu respectivo nível de avaria classificado pelo modelo.

A Figura 6 mostra um exemplo do mini-relatório gerado pelo sistema:

Figure 6: Exemplo de mini-relatório automatizado.

Este formato simplificado permite visualizar rapidamente quais peças do veículo apresentam danos e sua respectiva gravidade, fornecendo uma avaliação objetiva que pode auxiliar em contextos como inspeções de seguros e avaliações para revenda de veículos.

3.3 Discussão

Os resultados obtidos demonstram que a abordagem baseada em Deep Learning é eficaz para a análise automatizada de danos em veículos. O modelo YOLOv11 apresentou bom desempenho na segmentação de peças (mAP@0.5 de 0.91), enquanto o ResNet-50 alcançou resultados satisfatórios na classificação de avarias em portas (acurácia de 79.59%).

A metodologia em duas etapas permitiu dividir o problema complexo em tarefas mais gerenciáveis: primeiro identificar as peças do veículo e depois classificar o nível de dano em cada uma delas. Esta abordagem mostrou-se adequada, pois cada etapa requer análise de características visuais distintas.

Embora a ideia inicial de uso de aprendizado fracamente supervisionado não tenha sido implementada, a rotulagem manual das portas foi suficiente para demonstrar a viabilidade do conceito. As limitações observadas incluem a confusão entre classes intermediárias de dano (leve e moderado) e a restrição da classificação apenas às portas dos veículos, devido ao escopo do projeto.

A implementação do mini-relatório automatizado como resultado final do sistema demonstra o potencial prático da solução, oferecendo uma alternativa objetiva para processos tradicionalmente manuais e subjetivos de avaliação de danos.

4 Conclusão

Neste trabalho, foi desenvolvido um sistema de visão computacional capaz de analisar danos em veículos através da segmentação de peças e classificação de avarias. O sistema implementou com sucesso um pipeline completo, desde o processamento da imagem até a geração de um mini-relatório visual.

O modelo YOLOv11 demonstrou alta precisão na identificação das 21 classes de peças de veículos, enquanto o ResNet-50 apresentou desempenho satisfatório na classificação de 4 níveis de avaria em portas. Mesmo com a limitação de escopo, com classificação restrita às portas, os resultados validam a viabilidade da abordagem proposta.

As principais contribuições deste trabalho incluem a integração de técnicas de segmentação e classificação em um sistema unificado, a análise comparativa de desempenho dos modelos e a demonstração prática da aplicabilidade de Deep Learning para avaliação automatizada de danos em veículos.

Como trabalhos futuros, sugere-se a expansão do escopo para classificação de avarias em outros componentes além das portas, a exploração de técnicas de aprendizado semi-supervisionado para reduzir a necessidade de rotulagem manual e o desenvolvimento de estimativas de custo de reparo baseadas nas avarias detectadas.

5 Repositório de Código

O código deste projeto está disponível no seguinte repositório:

- `https://github.com/israelfontes/CarDamage`

References

[1] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[3] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully Convolutional Networks for Semantic Segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[4] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. *Proceedings of the European Conference on Computer Vision*, pages 801–818, 2018.

[5] Rahul Patil, Sudeep Kulkarni, and Yogesh Dhareshwar. A Comprehensive Survey on Vehicle Damage Detection Using Deep Learning. *International Journal of Engineering Research & Technology*, 6(6):510–514, 2017.

[6] Guan Wang, Yuchen Rao, Yuelong Li, and Jianru Xue. Car Damage Detection and Localization using Deep Learning. *IEEE Transactions on Intelligent Transportation Systems*, 23(2):879–888, 2022.

[7] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Weakly Supervised Learning for Object Detection and Segmentation. *International Journal of Computer Vision*, 127(6):745–761, 2019.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[9] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[10] Glenn Jocher. Ultralytics YOLO11. `https://github.com/ultralytics/ultralytics`, 2024.