

Trabalho 2

Aprendizado de Máquina
Professor Otávio Parraga

Arthur Kunzler (21103603), **Eduardo Ballico** (21102009), **Guilherme Dornelles** (21102248), **Giovane Milani**, **Vinicius Boff**

1. Introdução

O grupo escolheu desenvolver um modelo de aprendizado de máquina utilizando Redes Neurais profundas para Visão Computacional.

Hoje em dia, problemas de identificação de veículos para controle do trânsito, monitoramento e segurança, são cada vez mais relevantes. Isso se dá pois cada vez mais as cidades estão sendo modernizadas e equipadas com sistemas que necessitam de Visão Computacional registrando fotos e vídeos dos automóveis que trafegam pelas vias.

Dito isso, o grupo criou um modelo capaz de identificar qual o tipo de veículo presente em uma imagem, dentre as classes Carro, Caminhão, Ônibus e Motocicleta.

2. Referencial teórico

Fizemos o uso de Redes Neurais baseadas em Redes Residuais (ResNet). São um tipo de Rede Convolutiva, criada em 2015. Esta rede se destaca por trazer uma melhoria para as Redes Convolucionais usuais, que enfrentam o problema do “Vanishing gradient”, quando o gradiente dos pesos se torna tão pequeno com a passagem das camadas da rede, que acaba prejudicando o aprendizado da rede, em alguns casos até impedindo que ele ocorra.

Esse tipo de rede utiliza um mecanismo de blocos residuais entre blocos de convolução presentes entre camadas de ReLu. Faz com que os gradientes sejam diretamente propagados entre uma camada ReLu para a outra posterior, utilizando de “conexões residuais”, que fazem esse papel de levar o resíduo, como ilustrado na Imagem 1.

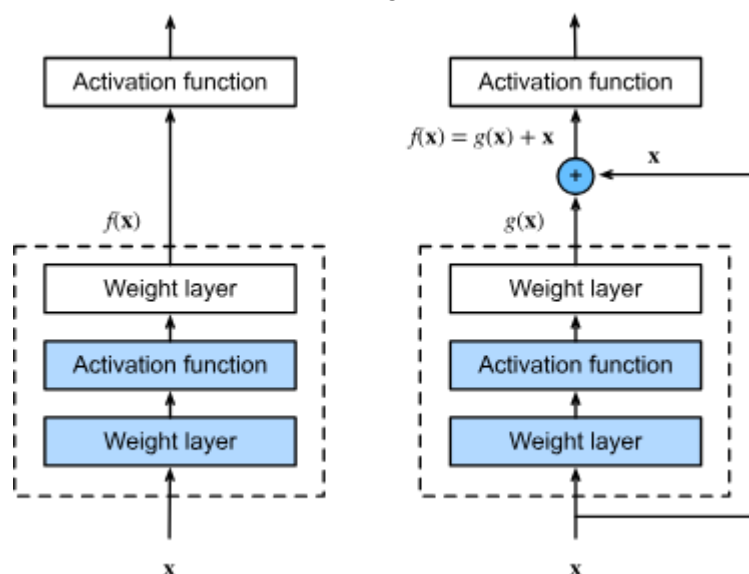


Imagem 1: Ilustração de um bloco normal (esquerda) com um bloco residual (direita).

Acima, na Imagem 1, vemos à direita um bloco residual, que traz o valor de x para a função de ativação, juntamente com o resultado do bloco $g(x)$. Isso facilita o aprendizado da função identidade $f(x)$.

Mais especificamente, utilizamos a arquitetura de Redes Residuais chamada 152, consiste de uma arquitetura com 152 camadas de convolução. Escolhemos essa arquitetura pois é mais robusta. Realizamos alguns testes de acurácia com outras arquiteturas de Rede Residual, como 50 e 34, porém performaram levemente abaixo ao que foi alcançado com a arquitetura 152.

3. Desenvolvimento

As funções de acurácia, matriz de confusão, validação e predição foram baseadas em códigos não próprios. Utilizamos os notebooks disponibilizados pelo professor, sites da internet e respostas do ChatGPT para construção de tais funções. No restante do código usamos referências das documentações oficiais das bibliotecas.

4. Coleta, preparação e avaliação dos dados

Para coleta dos dados, agrupamos dois datasets diferentes, ambos encontrados na internet. Um dos datasets possui cerca de 100 imagens de cada uma das quatro classes existentes, e o outro, cerca de 40. Utilizamos divisão aleatória de 80% para treinamento e 20% para teste, em um total de aproximadamente 140 imagens por classe.

Dividimos as imagens em pastas conforme as respectivas classes, e criamos uma pasta "Demo", separada contendo as imagens a serem usadas como demonstração durante a apresentação em aula.

Para preparação das imagens, criamos uma função de transformação, para realizar data augmentation no treinamento. Criamos um pipeline de transformação, manipulando características da imagem, como rotação em graus, inversão no eixo horizontal, alteração de perspectiva, coloração e saturação e zoom out. O pipeline funciona com base em probabilidade, para que randomicamente parte do dataset seja alterado. Isso ocorre em cada época de forma independente. Assim, aumenta-se a variabilidade dos dados de treinamento, contribuindo para a robustez do modelo.

Para o treinamento, utilizamos épocas como um valor variável, porém ao testar com mais de cerca de 12 épocas, o modelo perdeu acurácia. O valor da taxa de aprendizado foi encontrado manualmente, onde a relação época x taxa de aprendizado alcançasse acurácia satisfatória sem overfitting.

Para a função de treinamento, adicionamos um mecanismo de parada antecipada, caso a taxa de erro em validação não sofra melhora em até 0.01 comparada a taxa mínima local.

5. Resultados

O modelo desenvolvido obteve resultados superiores ao que o grupo esperava. Obtivemos como melhor performance 91.5% de acurácia. Essa métrica reflete o percentual de classificações corretas em relação ao total de amostras do conjunto de testes. Além disso, foi gerada a matriz de confusão conforme na Imagem 2.

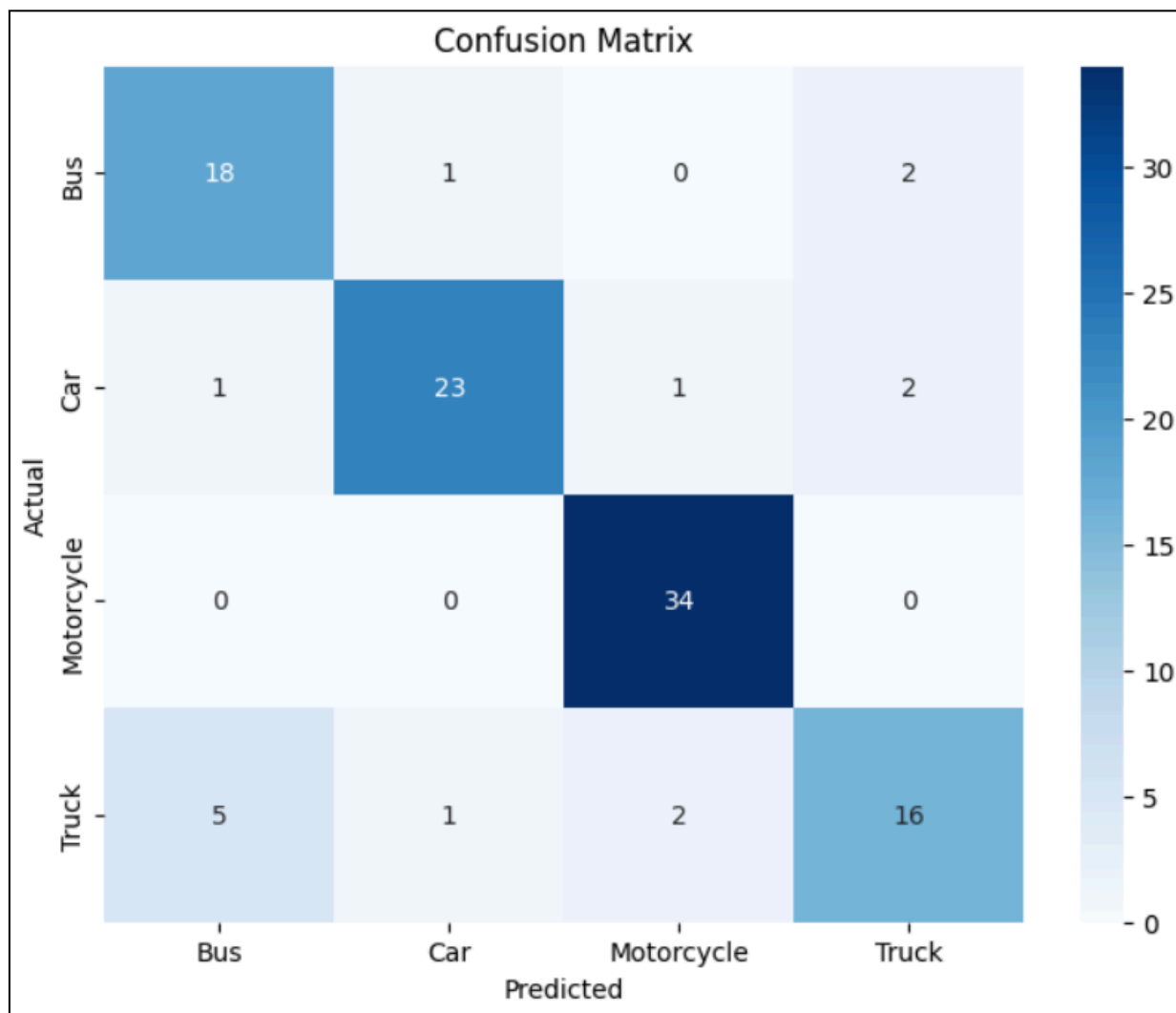



Imagem 2: Matriz de Confusão

Conforme podemos observar na Imagem 2, a matriz de confusão representa as classes previstas pelo modelo no eixo x e as classes reais das instâncias no eixo y. Obtivemos o maior grau de assertividade na classe Motocicleta, e o menor grau entre as classes Ônibus e Caminhão. Este é um resultado coerente pois são as classes com mais características visuais em comum. Porém, percebemos alguns casos “espúrios”, como o erro onde o modelo previu como Motocicleta o que na verdade era um Carro.

Observamos também, que o modelo tem certa variabilidade quanto aos erros de validação dependendo da divisão aleatória do dataset em treino e teste. Isso se dá pois, as imagens são bastante variadas, e algumas acabam tendo papel importante no aprendizado do modelo.

Nossa hipótese é que com maior base de dados para treinamento o modelo seria capaz de melhorar sua performance, se tornando mais assertivo em todas as classes.

Fizemos testes manuais com imagens encontradas na internet com diferentes modelos de veículos. Nesses testes, buscamos encontrar os limites do modelo. A Tabela 1, abaixo, retrata esses casos de teste que consideramos interessantes.

| Imagem | Descrição do veículo | Classe esperada | Clare prevista |
|---|--------------------------|-----------------|----------------|
|  | Moto moderna para viagem | Motocicleta | Motocicleta |

| | | | |
|---|-------------------------------|-------------|----------|
|  | Triciclo com passageiro extra | Motocicleta | Caminhão |
|  | Caminhão sem caçamba | Caminhão | Caminhão |
|  | Motorhome | Caminhão | Ônibus |
|  | Monster truck | Carro | Caminhão |
|  | Ônibus comum de Porto Alegre | Ônibus | Ônibus |
|  | Carro de Fórmula 1 | Carro | Carro |

Tabela 1: Testes com imagens selecionadas manualmente

Dados esses casos da Tabela 1, percebemos algumas limitações do modelo, como por exemplo a distinção do número de rodas para considerar algo como Motocicleta ou não, por exemplo, no caso do triciclo. Apesar da estrutura anormal quando comparada a uma moto, esse veículo aos olhos humanos se assemelha mais com uma do que com qualquer outra das três classes.

6. Conclusão

O modelo desenvolvido pelo grupo mostrou-se bem-sucedido em atingir seu principal objetivo: Um modelo de Aprendizado de Máquina baseado em redes neurais capaz de identificar veículos a partir de imagens.

Com uma acurácia de 91,5% e desempenho robusto em diversas situações distintas, o modelo se destacou pela satisfatória capacidade de generalização.

Destaca-se o uso de técnicas modernas de visão computacional, como a arquitetura ResNet-152, e aplicação de técnicas de treinamento visando mitigar problemas comuns de overfitting e baixo volume de dados.

Contudo, foram observadas algumas limitações do modelo, entre elas, a dificuldade em diferenciar classes semelhantes estruturalmente, além da sensibilidade a condições adversas nas imagens

Cenários como esses podem ser contornados parcialmente com

- Aumento do volume e densidade do dataset;

- Uso de técnicas avançadas de pré-processamento de imagem (Canny, Wavelet Transforms);

De modo geral, o trabalho evidenciou a aplicabilidade de redes neurais profundas para problemas práticos do cotidiano em visão computacional.

7. Referências

8.6. Residual Networks (ResNet) and ResNeXt — Dive into Deep Learning 1.0.3 documentation. Disponível em: <https://en.d2l.ai/chapter_convolutional-modern/resnet.html>. Acesso em: 25 nov. 2024.

SHARMA, T. Detailed Explanation of Residual Network(Resnet50) CNN Model. Disponível em: <<https://medium.com/@sharma.tanish096/detailed-explanation-of-residual-network-resnet50-cnn-model-106e0ab9fa9e>>.

torchvision.transforms — Torchvision master documentation. Disponível em: <<https://pytorch.org/vision/stable/transforms.html>>.

Wavelet Transforms. Disponível em: <<https://www.sciencedirect.com/topics/computer-science/wavelet-transforms>>. Acesso em 25 nov. 2024.

Datasets

Vehicle Type Recognition. Disponível em: <<https://www.kaggle.com/datasets/kaggleashwin/vehicle-type-recognition>>. Acesso em: 12 nov. 2024.

IBRAHIM, A. W. Cars Detection. Disponível em: <<https://www.kaggle.com/datasets/abdallahwagih/cars-detection>>. Acesso em: 20 nov. 2024.

Repositório

[Github](#)