Reconhecimento de Modelos de Veículos

Fernando Magalhães

Laboratório de Sistemas Distribuídos Inteligentes (LSDi)
Universidade Federal do Maranhão (UFMA)
fbeneditovm@gmail.com

15/01/2018

Outline

Introdução

- É comum que pessoas não saibam reconhecer os carros que observam no dia a dia, incluindo alguns dos modelos mais comuns;
- Um método de reconhecimento de veículos além de ajudar os curiosos, pode ajudar agentes de segurança a identificar automóveis cuja a placa não está visível ou pode ter sido adulterada.

Objetivos

- Pré-processar uma base de dados para o treinamento de redes neurais;
- Treinar diferentes arquiteturas de redes neurais;
- Comparar o desempenho de cada uma delas;

Dataset

- 2120 imagens de 50 classes;
- 10 marcas cada uma com 5 modelos;
- Por volta de 37 imagens por classe;
- Posições, câmeras e iluminações variadas;
- Imagens não estavam cortadas para conter apenas a bounding box

Pré-processamento

Foram escritos vários scripts para pré-processar as imagens do dataset tornando possível o treinamento utilizando as redes que serão mencionadas a seguir na plataforma de machine learning do google cloud:

- Cortar as imagens
- Separá-las em pastas de acordo com a classe
- Redimensionar para 299, 299 em RGB
- Salvá-las em um arquivo .pickle que pudesse ser aberto no gcloud

Observação

Foi enviada uma mensagem para o email disponibilizado na página do dataset de modo a obter o modelo de veículo o qual cada número representava, mas nenhuma resposta foi obtida. Então foi necessário observar as imagens em busca dos modelos e por meio de pesquisas na internet foi possível obter o modelo que cada inteiro representava.
As imagens foram dividias em em 80% para treino e 20% para validação

Aumento de dados

Devido ao número pequeno de imagens por veículo, foi utilizada uma técnica de aumento de dados nas imagens reservadas para treino, foram utilizados de forma aleatória durante o treinamento espelhamentos horizontais, inclinações de até 30 graus e zooms.
As imagens foram dividias em em 80% para treino e 20% para validação

Configuração das Redes

Foram adotadas 4 arquiteturas de redes neurais: uma CNN simples, uma com um módulo inception, uma com 3 módulos inceptions e por fim a GoogLeNet pré-inicializada com os pesos da base ImageNet. Seguem as arquiteturas das redes construídas utilizando a bibioteca keras

SimpleConvNet

- Convolução(32 filtros de (3,3), relu)
- MaxPooling(32 filtros de (2,2))
- Convolução(32 filtros de (3,3), relu)
- MaxPooling(32 filtros de (2,2))
- Convolução(64 filtros de (3,3), relu)
- MaxPooling(64 filtros de (2,2))
- Flaten()
- FullyConnected(128 saídas, relu)
- Dropout(50%)
- FullyConnected(50 saídas, relu)
- Sigmoid()

Single Inception

- Convolução(32 filtros de (3,3), stride(2,2) relu)
- Convolução(32 filtros de (3,3), relu)
- Convolução(64 filtros de (3,3), relu)
- MaxPooling(64 filtros de (3,3), stride(2,2))
- Convolução(80 filtros de (1,1), relu)
- Convolução(192 filtros de (3,3), relu)
- MaxPooling(192 filtros de (3,3), stride(2,2))
- 1 Módulo inception
- GlobalAveragePooling()
- FullyConnected(50 saídas, relu)
- Sigmoid()

GoogLenet

Substitui o final da rede para retreinar de acordo com o novo dataset.
Toma a saída do último módulo inception e adiciona:

- GlobalAveragePooling()
- FullyConnected(1024 saídas, relu)
- FullyConnected(50 saídas, relu)
- Sigmoid()

Treinamento

O treinamento foi feito em 100 épocas para todas as configurações de CNN, exceto para a GoogLeNet, cujo treinamento foi dividido em duas etapas:

- 50 epocas treinando apenas a parte que foi adicionada ao final da rede.
- 50 epocas treinando também o último módulo inception da rede.

SimpleConvNet

A SimpleConvNet apresentou acurácia de 5% no dataset de treino e 6% no dataset de validação(figuras 1 e 2).

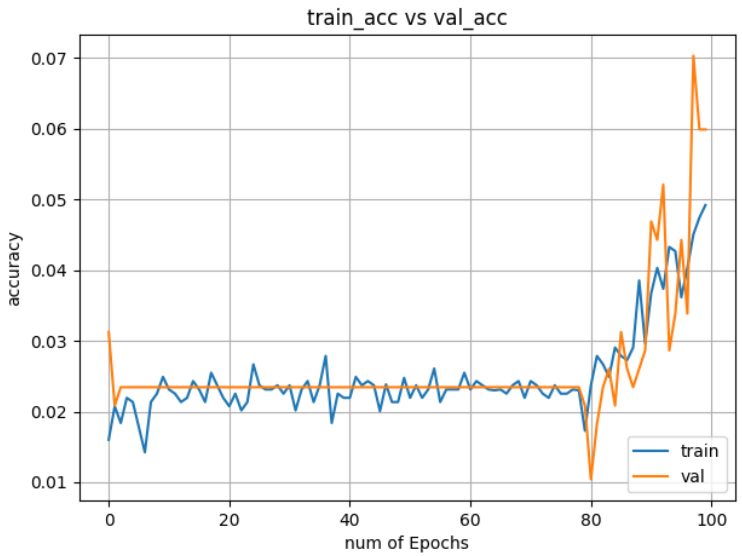

Figura 1: Acurácia SimpleConvNet.

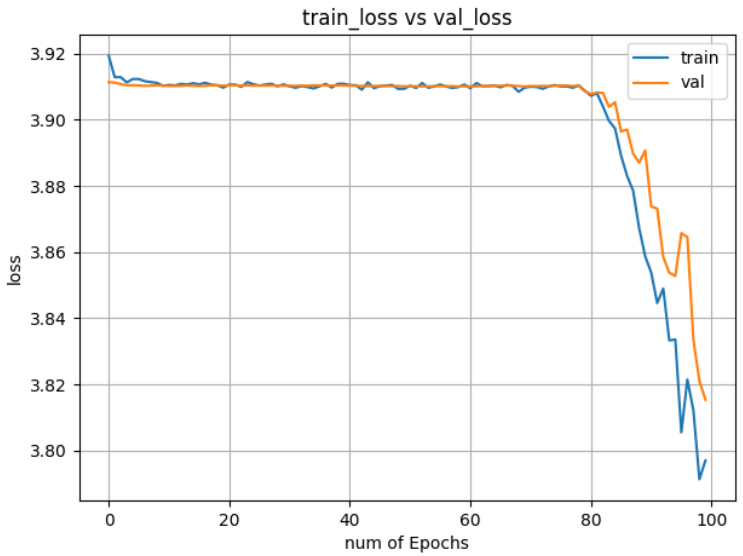

Figura 2: Loss SimpleConvNet.

Single Inception

A rede com 1 módulo inception apresentou uma acurácia de 93% no dataset de treino e 49% no dataset de validação(figuras 3 e 4).

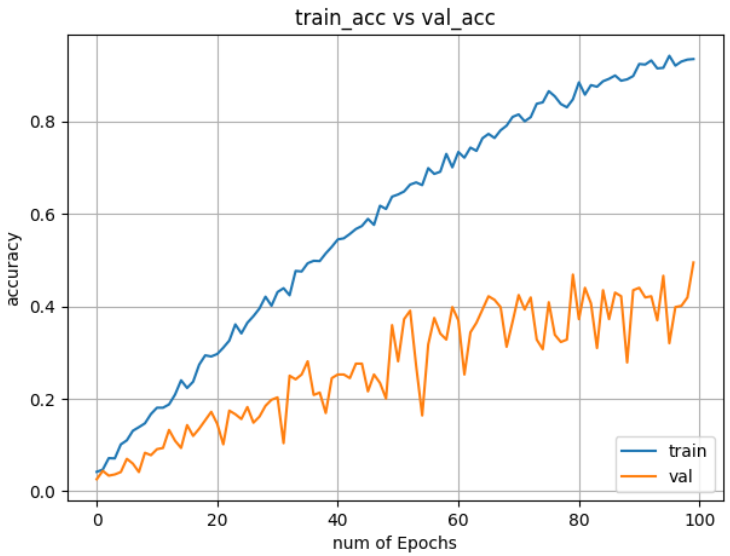

Figura 3: Acurácia Single Inception.

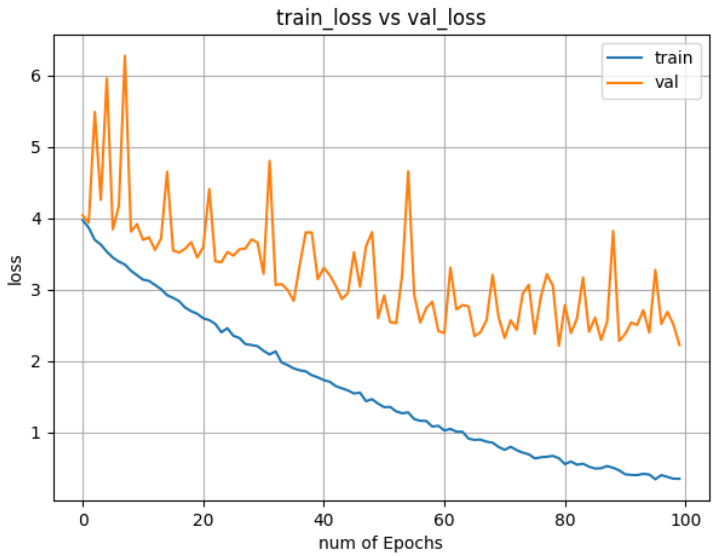

Figura 4: Loss Single Inception.

GoogLeNet etapa 1

Ao final das primeirs 50 épocas, a GoogLeNet apresentou uma acurácia de 63% no dataset de treino e 48% no dataset de validação (figuras 5 e 6).

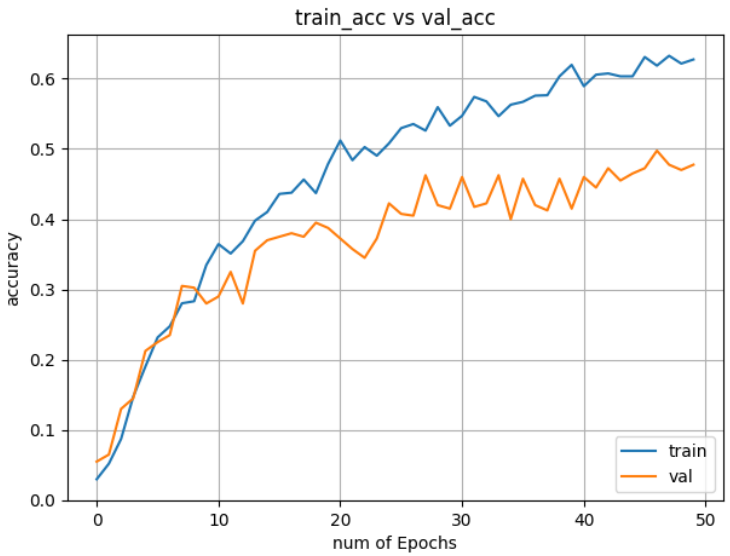

Figura 5: Acurácia GoogLeNet etapa 1.

Figura 6: Loss GoogLeNet etapa 1.

GoogLeNet etapa 2

Treinadas mais 50 épocas seguindo o descrito na seção anterior, a acurácia foi de 96% no dataset de treino e 67% no de validação(figuras 7 e 8).

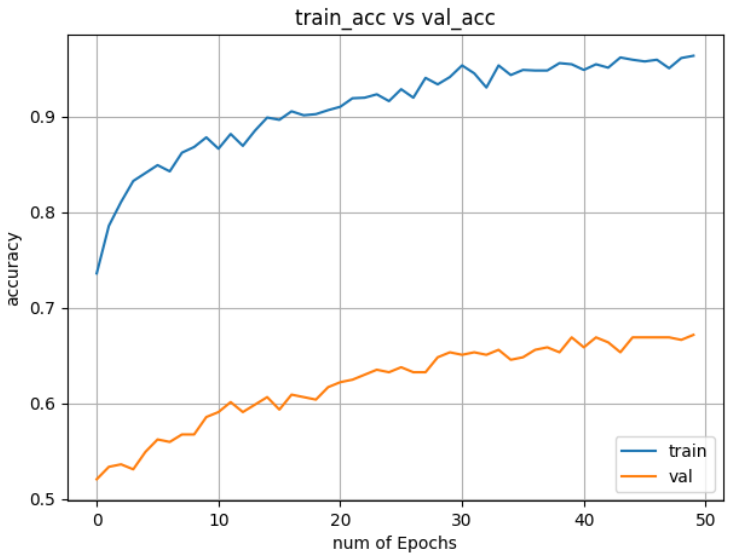

Figura 7: Acurácia GoogLeNet etapa 2.

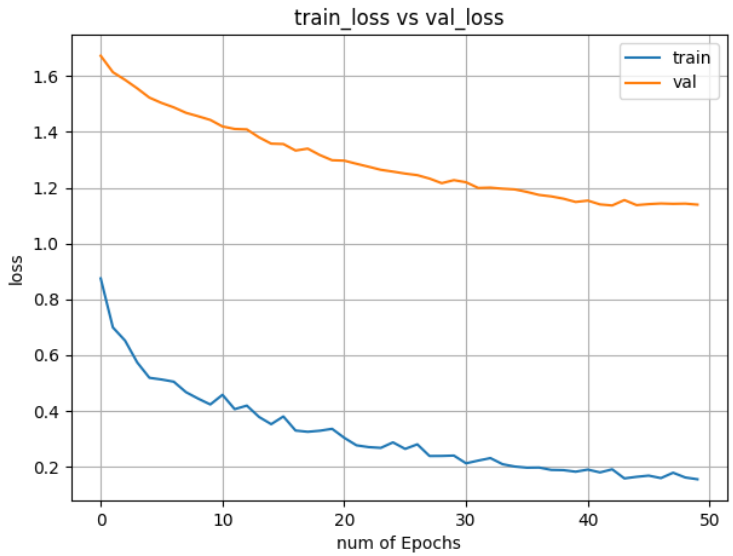

Figura 8: Loss GoogLeNet etapa 2.