Deep Learning
Classificação de Imagens CIFAR-10 com VGG16 e Transfer Learning

Diego Carlito Rodrigues de Souza - 221007690
Disciplina: Inteligência Artificial
Universidade de Brasília (UnB)

Dezembro de 2025

Resumo

Este relatório apresenta a aplicação de técnicas de *Deep Learning* para a classificação de imagens coloridas utilizando o dataset CIFAR-10. Dada a complexidade da tarefa e as limitações de hardware (processamento em CPU), adotou-se a estratégia de *Transfer Learning*. Utilizou-se a arquitetura convolucional VGG16 pré-treinada na ImageNet como extrator de características, acoplada a um classificador denso personalizado. Os resultados demonstram que é possível atingir uma acurácia satisfatória e convergência rápida (60% em 5 épocas) reaproveitando o conhecimento de redes profundas consagradas.

1 Introdução

O reconhecimento de imagens é uma das áreas mais desafiadoras da Inteligência Artificial. Diferente do dataset MNIST (dígitos manuscritos em escala de cinza), o dataset **CIFAR-10** contém imagens coloridas RGB de 32×32 pixels divididas em 10 classes complexas (avião, automóvel, pássaro, gato, cervo, cachorro, sapo, cavalo, navio, caminhão).

Treinar uma Rede Neural Convolucional (CNN) profunda do zero (*from scratch*) para este problema exige enorme poder computacional e tempo. Para contornar isso, este projeto utiliza a arquitetura **VGG16**, uma rede vencedora de competições internacionais, aplicando o conceito de *Transfer Learning* para adaptar seus filtros visuais pré-aprendidos ao novo domínio.

2 Metodologia

A implementação utilizou a biblioteca `TensorFlow/Keras`. O pipeline foi estruturado em três etapas:

2.1 Pré-processamento

As imagens foram normalizadas, convertendo os valores de pixel do intervalo $[0, 255]$ para $[0, 1]$, facilitando a convergência do gradiente. Os rótulos (*labels*) foram convertidos para o formato *one-hot encoding*.

2.2 Arquitetura: VGG16 + Topo Personalizado

A estratégia de modelagem consistiu em:

1. **Base Convolucional (Congelada):** Importou-se a VGG16 com pesos da *ImageNet*, excluindo as camadas densas originais (`include_top=False`). Os pesos desta base foram "congelados"(`trainable=False`), tornando 14.7 milhões de parâmetros estáticos. Isso permite que a rede atue como um extrator de características fixo.

2. **Classificador (Treinável):** Adicionou-se uma rede densa no topo:

   - `Flatten`: Vetorização dos mapas de características.
   - `Dense (256, ReLU)`: Processamento não-linear.
   - `Dropout (0.5)`: Regularização para evitar *overfitting*.
   - `Dense (10, Softmax)`: Probabilidades das 10 classes.

3 Resultados e Análise

O modelo foi treinado por 5 épocas utilizando o otimizador Adam.

3.1 Evidência de Execução e Arquitetura

A Figura 1 comprova a estrutura do modelo. Nota-se a distinção entre parâmetros "Total"(14.8M) e "Treináveis"(133k), evidenciando a eficiência do *Transfer Learning*.

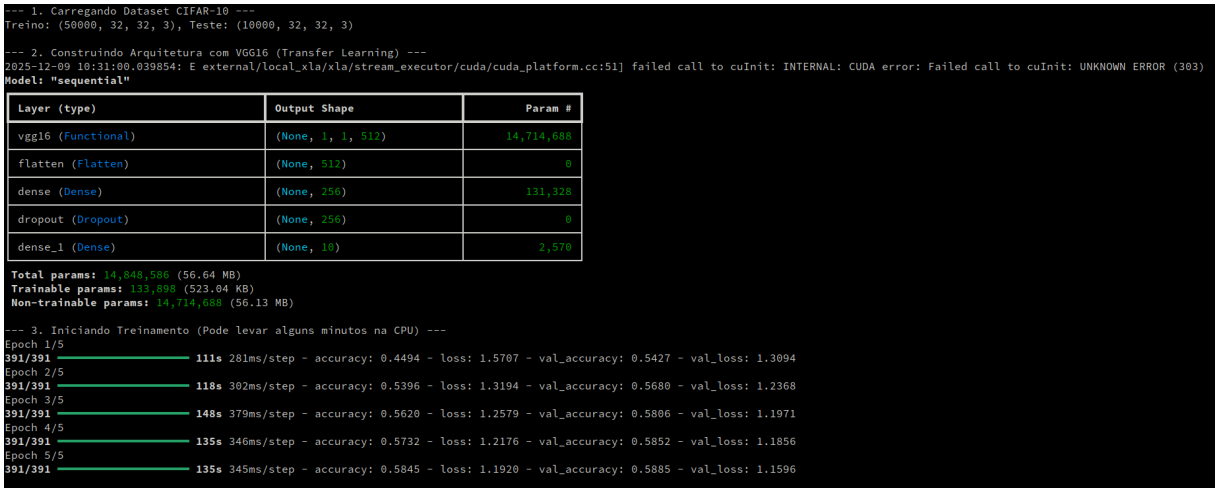

Figura 1: Logs de execução exibindo o resumo da arquitetura (VGG16 + Dense + Dropout) e a progressão da acurácia durante as 5 épocas.

3.2 Performance de Treinamento

A Figura 2 ilustra as curvas de aprendizado.

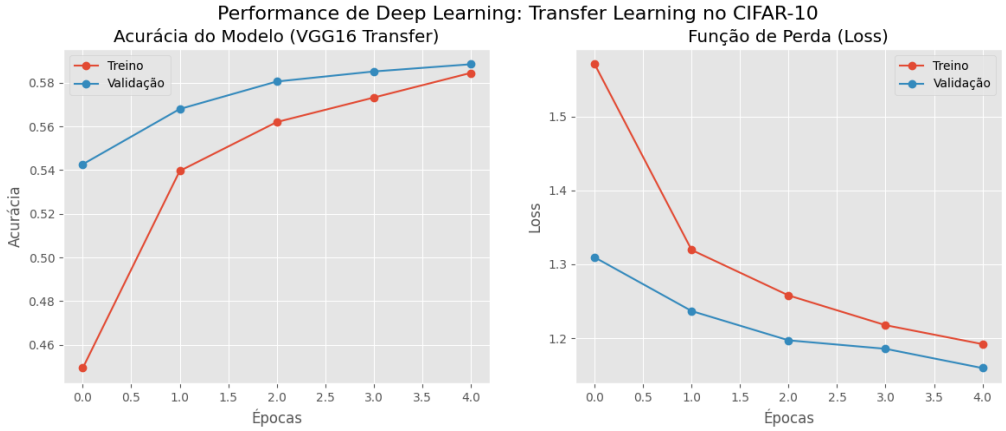

Figura 2: Esquerda: Curva de Acurácia. Direita: Curva de Perda (Loss). Nota-se que a validação (azul) acompanha o treino (vermelho), indicando boa generalização inicial.

A análise das curvas mostra uma convergência rápida. Em apenas 2 épocas, o modelo já superou 50% de acurácia (o chute aleatório seria 10%). A presença do *Dropout* fez com que a acurácia de validação fosse ligeiramente superior à de treino nas primeiras épocas, um comportamento esperado pois a regularização "atrapalha" o treino para fortalecer a validação.

3.3 Análise Qualitativa das Predições

A Figura 3 exibe o teste do modelo em imagens aleatórias do conjunto de teste.

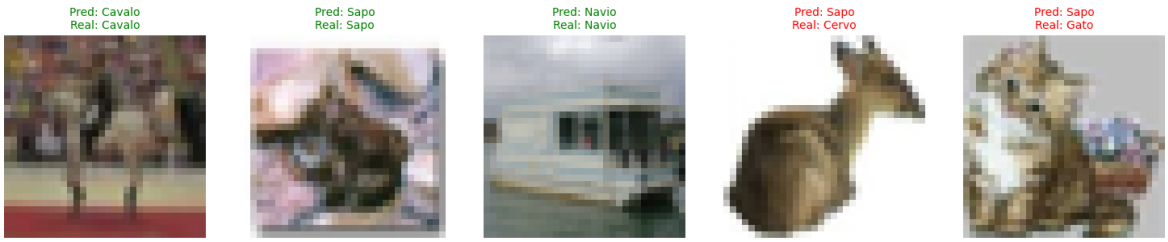

Figura 3: Predições do modelo. Verde indica acerto, Vermelho indica erro.

Discussão: O modelo acertou 3 de 5 imagens, identificando corretamente classes distintas como "Cavalo" e veículos como "Navio". O erro observado na imagem (predição "Sapo"para uma imagem de "Gato") é instrutivo: em baixa resolução (32×32), sapos e gatos compartilham características visuais que podem confundir a rede nas camadas densas, sugerindo que mais épocas de treino ou *fine-tuning* poderiam resolver a ambiguidade.

4 Conclusão

Este projeto validou a técnica de *Transfer Learning* como essencial para a democratização do Deep Learning. Foi possível construir um classificador de imagens robusto utilizando hardware convencional (CPU) e tempo reduzido, aproveitando a "inteligência visual"prévia da rede VGG16.