CNN para Previsão de Rostos

Caio Franco Porto
Camilo Alves Mascarenhas de Almeida

30 de dezembro de 2024

1 Resumo

O trabalho teve como objetivo desenvolver e avaliar modelos de redes neurais convolucionais (CNNs) capazes de distinguir rostos masculinos e femininos utilizando imagens do banco de dados CUHK Face Sketch Database. Foram realizadas análises com imagens em cores, imagens convertidas para preto e branco e utilizando técnicas de aumento de dados (data augmentation). Métricas como Acurácia, F1-Score e AUC-ROC foram utilizadas para comparar os modelos. Os resultados mostraram principalmente que o data augmentation e a conversão do dataset para preto e branco melhoraram muito o desempenho do modelo.

2 Introdução

A classificação de imagens é um problema central em visão computacional, com aplicações que vão desde reconhecimento facial até diagnóstico médico. O objetivo geral dessa tarefa é categorizar imagens em classes predefinidas com base em padrões visuais. No entanto, devido à alta dimensionalidade das imagens e à complexidade das características visuais, métodos tradicionais de aprendizado de máquina frequentemente apresentam limitações em termos de precisão e escalabilidade.

Nesse contexto, as redes neurais convolucionais (CNNs) têm emergido como uma abordagem altamente eficaz para problemas de classificação de imagens. Inspiradas na organização do córtex visual humano, as CNNs são projetadas para identificar automaticamente padrões hierárquicos, desde bordas simples até estruturas mais complexas, como formas e texturas. Elas utilizam camadas convolucionais para extração de características, camadas de pooling para redução da dimensionalidade e camadas densas para a tomada de decisão, o que as torna particularmente adequadas para lidar com imagens em alta resolução.

Além disso, as CNNs têm a capacidade de generalizar bem em grandes volumes de dados e são robustas a variações nas imagens, como iluminação, ângulo e escala. Com o suporte de técnicas avançadas, como ajuste de hiperparâmetros e aumento de dados (data augmentation), essas redes podem alcançar desempenhos superiores em tarefas desafiadoras, como a diferenciação de características sutis entre classes.

O presente trabalho utiliza CNNs para realizar a classificação de imagens de rostos, com foco na distinção entre rostos masculinos e femininos. Essa tarefa não apenas testa a eficiência das redes em identificar padrões faciais, mas também explora o impacto de diferentes pré-processamentos de imagem e ajustes de hiperparâmetros no desempenho dos modelos.

3 Metodologia

Além disso, foi feita a normalização das imagens, padronizando todos os pixels em 0 e 1. Seguido disso, separamos o dataset em treino, teste e validação,

usando o parametro stratify para garantir que todos teriam proporcionalidade nas classes.

A arquitetura criada foi bem simples, com um camada de input seguindo de 4 camadas de convolução e pooling. Ao final da arquitetura adicionamos uma camada Flatten e uma camada Densa com função sigmoid. Compilamos usando o o otimizador Adam e a métrica de loss foi o binary-crossentropy.

Os hiperparâmetros foram escolhidos usando grid search com a biblioteca Keras Tuner. Os melhores parâmetros após 30 testes foram escolhidos para treinar as redes. Uma exeção foi o teste com imagens preto e branco, onde usamos os mesmos parâmetros dados para as imagens originais.

Adicionalmente, como usamos o early stopping como callback para evitar que o modelo treinasse mais que o necessário.

O processo de avaliação se deu primeiramente avaliando a acurássia e o loss. E, posteriormente, utilizamos F1-Score, curva ROC e área sobre a curva ROC para avaliar o modelo. Adjunto a isso analisamos os rostos que foram preditos erroneamente para procurar padrões nos erros.]A metodologia foi igual nos 3 testes que fizemos (imagens originais, preto e branco e com fata augmentation). Começamos criando a lista de rótulos que seria usada para classificar as imagens. Como cada arquivo começava com "m" ou "f" indicando o sexo, usamos esse parametro para criar os rótulos.

Além disso, foi feita a normalização das imagens, padronizando todos os pixels em 0 e 1. Seguido disso, separamos o dataset em treino, teste e validação, usando o parametro stratify para garantir que todos teriam proporcionalidade nas classes.

A arquitetura criada foi bem simples, com um camada de input seguindo de 4 camadas de convolução e pooling. Ao final da arquitetura adicionamos uma camada Flatten e uma camada Densa com função sigmoid. Compilamos usando o o otimizador Adam e a métrica de loss foi o binary-crossentropy.

Os hiperparâmetros foram escolhidos usando grid search com a biblioteca Keras Tuner. Os melhores parâmetros após 30 testes foram escolhidos para treinar as redes. Uma exeção foi o teste com imagens preto e branco, onde usamos os mesmos parâmetros dados para as imagens originais.

Adicionalmente, como usamos o early stopping como callback para evitar que o modelo treinasse mais que o necessário.

O processo de avaliação se deu primeiramente avaliando a acurássia e o loss. E, posteriormente, utilizamos F1-Score, curva ROC e área sobre a curva ROC para avaliar o modelo. Adjunto a isso analisamos os rostos que foram preditos erroneamente para procurar padrões nos erros.

4 Discussão

A F1 score indicou um bom equilíbrio nas classes nos testes com imagens preto e branco e Data Augmentation. Porém, com as imagens originais, mesmo com os melhores parametros encontrados no grid search, o resultado foi péssimo.

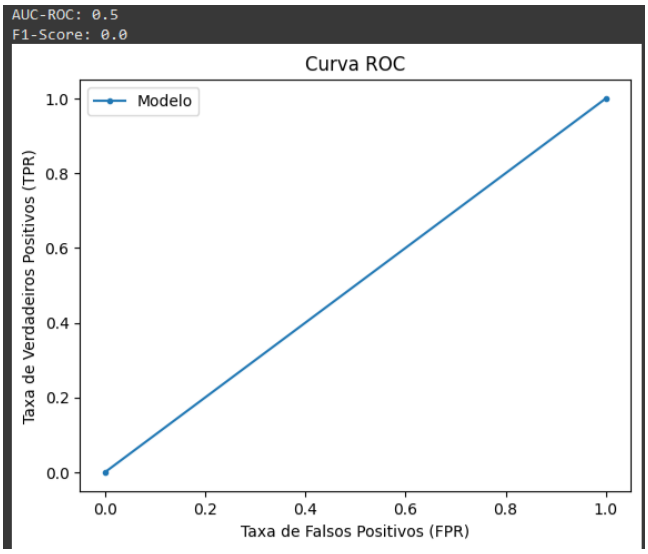

Figure 1: Imagens originais

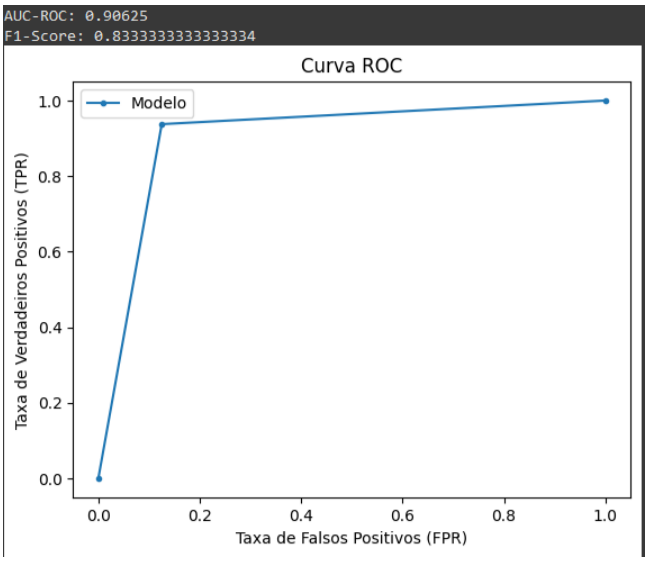

Figure 2: Imagens Preto e Branco

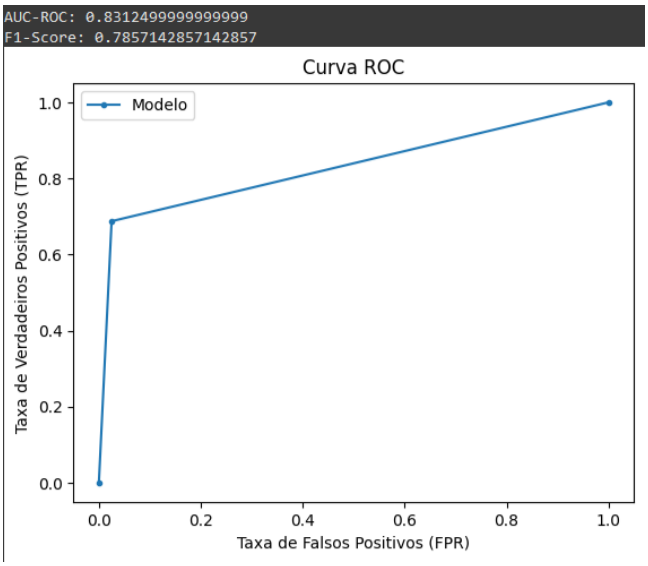

Figure 3: Imagens com Data Augmentation

Como podemos ver, os resultados com as imagens em preto e branco tiveram os melhores resultados. Particularmente isso foi contra o que acreditei, pois pensei que o Data Augmentation teria um desempenho muito melhor, mesmo em relação as imagens em preto e branco.

Algumas imagens tinha borrões, como por exemplo uma moça estava piscando o olho, ou desfoque na imagem. Essas imagens estavam nos erros, porém eram minoria no dataset. Na maioria das imagens classificadas erroneamentes víamos principalmente mulheres, provavelmente devido ao desbalanceamento entre as classes, com cabelos amarrados, que poderia dar impressão de cabelo curto masculino. Rostos mais quadrados também foram enquadrados como homens. Além disso, provalvelmente por sua minoria, crianças (garotas) foram classificadas errado. Acredito que melhorar a estrutura das camadas, talvez com um dropout ou batch normalization ajudaria a melhorar o desempenho, porém preferi manter a arquitetura simples e testar modos diferentes de utiliza-la.

Uma surpresa durante o desenvolvimento aconteceu quando fui adicionar o grid search, e na verdade ao contrário de melhorar todas as métricas, algumas pioraram, como as metricas dos dados com imagens originais. Foi um grande susto, mas como deu certo nos outros casos mantive no código.

Figure 4: Erros

5 Conclusão

O projeto demonstrou a eficácia de redes neurais convolucionais na classificação de gênero em imagens faciais, utilizando técnicas como conversão para escala de cinza, data augmentation e ajuste de hiperparâmetros. Os resultados, avaliados por métricas como F1-Score e AUC-ROC, indicaram que o modelo foi capaz de capturar padrões relevantes, com destaque para informações espaciais e texturais, enquanto a cor mostrou ser menos determinante para este problema. O uso de data augmentation também contribuiu para aumentar a robustez do modelo.

Para aprimorar o projeto, gostaria de usar de bases de dados maiores e mais diversificadas e a experimentação de arquiteturas mais complexas, como algo parecido com a ResNet ou EfficientNet. Adicionalmente, métodos de pré-processamento avançados, como equalização de histograma, e a aplicação de técnicas de interpretação, como Grad-CAM, podem ajudar a melhorar a performance do modelo e fornecer maior compreensão sobre os padrões utilizados na classificação.

6 Referência

PYTORCH TEAM. TorchVision: Models, Datasets, and Transforms. Disponível em: https://pytorch.org/vision/.

HUNTER, John D. Matplotlib: A 2D Graphics Environment. Computing in Science Engineering, v. 9, n. 3, p. 90-95, 2007.

TEAM, NumPy Developers. NumPy: The fundamental package for scientific computing in Python. Disponível em: https://numpy.org/.

ROSSUM, Guido van. Python Programming Language. Python Software Foundation. Disponível em: https://www.python.org/.

KAGGLE. CUHK Face Sketch Database (CUFS). Disponível em: https://www.kaggle.com/datasets/arbazkhan971/cuhk-face-sketch-database-cufs.

LUNDH, Fredrik. Python Imaging Library (PIL). 1999. Disponível em: https://pillow.readthedocs.io/.

CLARK, Steven. Matplotlib: Visualization with Python. 2003. Disponível em: https://matplotlib.org/.

KERAS TEAM. Keras Tuner. Disponível em: https://keras.io/keras$_{t}uner/$.

PEDREGOSA, Fabian et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, v. 12, p.

2825-2830, 2011. Disponível em: https://scikit-learn.org/.