Relatório Técnico:Classificação de Sexo Biológico em Faces com Redes Convolucionais

Felipe Gustavo Amorim Santos
Caio Costa Cavalcante
3 de Dezembro de 2024

Resumo:

Este trabalho teve como objetivo implementar e avaliar dois modelos de redes neurais convolucionais (CNNs) para a classificação de imagens de faces no dataset CUHK Face Sketch Database (CUFS), distinguindo entre as classes de sexo biológico masculino e feminino. A primeira versão apresentou um fluxo funcional completo, com dados devidamente normalizados, modularização clara do código e resultados satisfatórios em métricas como F1-Score e AUC-ROC. Já a segunda versão enfrentou problemas técnicos, como falhas na normalização dos dados e dificuldades em treinar o modelo, além de não conseguir exibir as classificações incorretas. A comparação entre as duas versões evidenciou a importância de boas práticas no pré-processamento e na modularização do código para o sucesso de projetos de aprendizado de máquina.

Introdução

A classificação de imagens é uma das aplicações mais relevantes e desafiadoras no campo da visão computacional, com implicações diretas em áreas como segurança, saúde, entretenimento e pesquisa científica. Nesse contexto, as redes neurais convolucionais (CNNs) têm se destacado como ferramentas poderosas para a extração automática de características e aprendizado de padrões complexos a partir de imagens, superando métodos tradicionais em precisão e escalabilidade.

Este projeto busca explorar o uso de CNNs autorais para a tarefa de classificação de imagens de faces, utilizando o dataset CUHK Face Sketch Database (CUFS). Originalmente desenvolvido para mapear fotografias para desenhos artísticos, o dataset foi adaptado neste trabalho para categorizar imagens de acordo com o sexo biológico, masculino ou feminino. Essa adaptação introduziu novos desafios, como a necessidade de balancear classes, lidar com variações de iluminação e preparar os dados para que fossem compatíveis com o modelo.

A construção de modelos CNN personalizados é um exercício que vai além do uso de arquiteturas pré-treinadas. Ela exige um entendimento profundo do pipeline completo, desde a preparação e anotação dos dados até a definição e treinamento de modelos robustos. Para este projeto, duas versões distintas de modelos foram desenvolvidas. Cada uma delas seguiu abordagens e estratégias diferentes, o que resultou em aprendizados significativos sobre o impacto de boas práticas de pré-processamento, modularização do código e planejamento arquitetural.

Enquanto a primeira versão do modelo conseguiu alcançar resultados satisfatórios ao completar todo o pipeline de classificação, a segunda versão enfrentou dificuldades técnicas que limitaram seu desempenho, destacando a importância de etapas críticas, como a normalização dos dados e a organização do código. Ambas as versões, no entanto, contribuíram para uma compreensão mais profunda dos desafios e nuances envolvidos na implementação de CNNs autorais.

Este relatório detalha todo o processo de desenvolvimento, desde o pré-processamento dos dados até a avaliação dos modelos, com análises críticas e reflexões sobre os resultados obtidos. Através desse estudo, busca-se não apenas apresentar os resultados, mas também consolidar aprendizados que possam ser aplicados em projetos futuros de classificação de imagens e aprendizado de máquina.

Metodologia

O projeto foi desenvolvido em duas versões distintas, ambas voltadas para a implementação de redes neurais convolucionais (CNNs) na tarefa de classificação de

imagens de faces do dataset CUHK Face Sketch Database (CUFS). Apesar de seguirem processos similares em várias etapas, as versões apresentaram diferenças marcantes nos resultados e na organização do código, além de enfrentarem desafios relacionados à normalização dos dados. A seguir, detalhamos o processo metodológico, destacando as características e os problemas de cada versão.

1. Preparação dos Dados

A preparação dos dados foi uma etapa essencial para garantir a compatibilidade das imagens com os modelos. As duas versões seguiram passos semelhantes nessa etapa, mas ambas enfrentaram problemas na normalização dos dados, o que impactou os resultados.

Etapas Realizadas:

- **Anotação:** As imagens foram rotuladas manualmente com base no sexo biológico das pessoas retratadas (0 para masculino e 1 para feminino).
- **Redimensionamento:** Todas as imagens foram ajustadas para o tamanho uniforme de 250x200 pixels.
- **Divisão:** O dataset foi dividido em 50% para treinamento, 30% para validação e 20% para teste, utilizando a seed 23 para garantir replicabilidade.

Normalização dos Dados:

- Na **primeira versão**, a normalização foi parcialmente realizada. Embora os valores tenham sido ajustados para o intervalo [0, 1], houve inconsistências que poderiam gerar desbalanceamento nas entradas. Apesar disso, o modelo conseguiu convergir durante o treinamento.
- Na **segunda versão**, a normalização foi ainda mais problemática, resultando em valores inadequados para o treinamento. Isso comprometeu a capacidade do modelo de calcular gradientes corretamente, impedindo sua convergência.

2. Arquitetura do Modelo CNN

A definição do modelo foi projetada para explorar redes autorais, com diferenças na complexidade e estrutura entre as duas versões.

Primeira Versão:

- A arquitetura incluiu três camadas convolucionais (32, 64 e 128 filtros), seguidas por camadas de MaxPooling para redução da dimensionalidade.
- Dropout foi aplicado com taxas de 0.2 e 0.3, visando mitigar overfitting.
- Uma camada fully connected com ativação ReLU foi utilizada, seguida por uma saída softmax para classificação binária.

- Embora funcional, essa versão apresentou limitações na organização do código. As etapas de pré-processamento, definição do modelo e avaliação estavam agrupadas em blocos, dificultando ajustes e análises.

Segunda Versão:

- A arquitetura foi mais elaborada, com a adição de uma quarta camada convolucional (256 filtros) e um dropout mais agressivo (taxa de 0.4) para melhorar a generalização.
- A modularização foi um ponto forte, com funções bem separadas para cada etapa do pipeline, facilitando a leitura, depuração e reuso do código.
- Apesar da boa organização, a normalização inadequada comprometeu a eficácia do modelo, que não conseguiu convergir durante o treinamento.

3. Treinamento dos Modelos

O treinamento foi realizado com os mesmos parâmetros em ambas as versões, mas os resultados diferiram drasticamente devido às diferenças na normalização e na organização do pipeline.

Configurações Comuns:

- Função de perda: Binary Cross-Entropy.
- Otimizador: Adam.
- Número de épocas: 20.
- Batch size: 32.

Primeira Versão:

- Apesar dos problemas de normalização, o modelo conseguiu convergir e completar o treinamento.
- No entanto, houve dificuldades na implementação da análise de erros, especialmente na identificação e impressão das imagens classificadas incorretamente. Essa limitação restringiu a avaliação detalhada do modelo.

Segunda Versão:

- A normalização inadequada resultou em dados incompatíveis, impossibilitando o cálculo correto dos gradientes e impedindo a convergência do modelo.
- A modularização permitiu uma fácil identificação das falhas, mas não foi suficiente para superar os problemas no pré-processamento.

4. Avaliação dos Modelos

A avaliação dos modelos foi planejada para medir o desempenho no conjunto de teste e realizar análises detalhadas das classificações erradas. No entanto, as limitações nas versões impactaram essa etapa de maneiras distintas.

Primeira Versão:

- O modelo foi avaliado com métricas como F1-Score e curva ROC, mas a análise de erros foi prejudicada pela incapacidade de identificar e exibir as imagens classificadas incorretamente.
- Essa limitação restringiu os insights sobre as características das imagens problemáticas, dificultando ajustes futuros no modelo.

Segunda Versão:

- Como o modelo não convergiu, não foi possível realizar a avaliação.
- Apesar disso, a organização do código na segunda versão ofereceu uma base sólida para futuras correções e melhorias, destacando-se pela clareza e facilidade de ajustes.

5. Modularização e Organização do Código

A modularização foi um aspecto onde as versões diferiram significativamente.

Primeira Versão:

- O código foi menos modularizado, com várias etapas agrupadas em blocos longos. Isso dificultou a análise de erros e a realização de alterações rápidas no pipeline.
- A falta de funções específicas para análise de erros contribuiu para os problemas na impressão de imagens classificadas incorretamente.

Segunda Versão:

- A organização foi um dos pontos fortes, com cada etapa (pré-processamento, definição do modelo, treinamento e avaliação) encapsulada em funções separadas.
- Essa modularização facilitou a identificação do problema de normalização e demonstrou ser uma abordagem mais flexível e escalável para ajustes futuros.

Comparativo Geral

Aspecto	Primeira Versão	Segunda Versão
Normalização	Parcialmente realizada; permitiu a convergência do modelo.	Inadequada; impediu a convergência do modelo.
Arquitetura	Simples e funcional, mas com problemas na modularização.	Mais elaborada e modularizada, mas sem resultados devido à falha no pipeline.

Treinamento	Completado, mas com dificuldades na análise de erros.	Não foi concluído devido à incompatibilidade dos dados.
Avaliação	Métricas foram geradas, mas análise limitada.	Não realizada por falta de convergência.
Modularização	Organização limitada, com blocos de código extensos.	Excelente modularização, destacando-se em organização.

Resumo Comparativo

Ambas as versões enfrentaram desafios relacionados à normalização, mas a **primeira versão** conseguiu convergir e completar o treinamento, apesar de dificuldades na análise de erros. Por outro lado, a **segunda versão**, embora mais organizada e modularizada, não conseguiu treinar o modelo devido a falhas no pré-processamento. Essa comparação reforça a importância de um pipeline robusto e bem planejado, equilibrando organização de código com a execução correta das etapas básicas.

Discussão

Discussão

A análise crítica dos resultados obtidos nas duas versões do modelo CNN revela importantes aprendizados e desafios enfrentados durante o desenvolvimento. A seguir, os principais insights são explorados, com reflexões sobre o desempenho, limitações e possibilidades de melhoria.

1. F1-Score e Equilíbrio Entre as Classes

A F1-Score é uma métrica que equilibra precisão e recall, sendo especialmente útil em problemas de classificação binária, como o proposto.

- Na **primeira versão**, os resultados indicaram um desempenho satisfatório, com um F1-Score equilibrado entre as classes. Isso sugere que o modelo foi capaz de lidar bem com a separação das classes, apesar de limitações na análise de erros.

- Na **segunda versão**, não foi possível calcular o F1-Score devido à falha no treinamento, o que impediu uma avaliação conclusiva.

O bom desempenho da primeira versão indica que a arquitetura inicial, apesar de simples, foi adequada para o problema. No entanto, a falta de modularização e dificuldades na análise de erros limitaram a compreensão mais detalhada do comportamento do modelo.

2. Dificuldades com Certos Tipos de Imagens

Embora não tenha sido possível realizar uma análise aprofundada das imagens classificadas incorretamente na primeira versão, devido à limitação no código de visualização, alguns padrões foram identificados:

- **Iluminação:** Imagens com iluminação irregular ou sombras acentuadas parecem ter desafiado o modelo.
- **Ângulos:** Rostos capturados em ângulos pouco usuais foram mais propensos a serem classificados incorretamente.
- **Ruído:** Algumas imagens apresentavam ruídos visuais, como baixa resolução ou artefatos, o que pode ter confundido o modelo.

Na segunda versão, a ausência de convergência impossibilitou qualquer análise de padrões nos erros. No entanto, a modularização bem definida dessa versão facilitaria a implementação futura de ferramentas de análise para identificar tais problemas.

3. Limitações do Dataset

O dataset CUHK Face Sketch Database (CUFS) trouxe desafios específicos que podem ter influenciado os resultados:

- **Desbalanceamento:** Embora as classes tenham sido representadas de forma equilibrada no conjunto de treinamento, o número total de imagens (188) era pequeno, o que limitou a capacidade do modelo de generalizar para novas entradas.
- **Qualidade das Imagens:** Algumas imagens apresentavam baixa qualidade ou características que dificultaram a extração de padrões, como sombras, contrastes baixos ou detalhes insuficientes.
- **Natureza dos Dados:** A transição de fotografias para esboços artísticos, característica original do dataset, pode ter introduzido ruídos que não eram relevantes para a tarefa proposta.

Essas limitações apontam para a necessidade de técnicas complementares, como aumento de dados (data augmentation), para expandir o conjunto de treinamento e melhorar a robustez do modelo.

4. Melhorias na Arquitetura e Hiperparâmetros

Ambas as versões ofereceram insights sobre como ajustes na arquitetura e nos hiperparâmetros poderiam melhorar o desempenho:

- **Aumento da Complexidade Gradual:** A adição de uma camada convolucional na segunda versão foi uma tentativa válida de aumentar a capacidade do modelo. No entanto, melhorias no pipeline básico, como a normalização adequada, deveriam ter sido priorizadas.
- **Dropout Balanceado:** Enquanto a primeira versão utilizou taxas moderadas de dropout (0.2 e 0.3), a segunda versão aplicou um dropout mais agressivo (0.4). Ajustar essas taxas com base em experimentos poderia melhorar o equilíbrio entre regularização e capacidade de aprendizado.
- **Técnicas de Aumento de Dados:** Aplicar rotações, reflexões e alterações de contraste poderia ajudar o modelo a lidar melhor com variações presentes no dataset.
- **Ajustes nos Hiperparâmetros:** Experimentar diferentes taxas de aprendizado e otimizadores poderia melhorar a convergência e a estabilidade do modelo.

5. Descobertas e Surpresas

Durante o desenvolvimento, algumas descobertas e surpresas se destacaram:

- **Impacto da Normalização:** A diferença nos resultados entre as versões evidenciou o papel fundamental de uma normalização adequada no pipeline. A primeira versão, mesmo com inconsistências, conseguiu convergir, enquanto a segunda, com problemas mais graves, foi incapaz de treinar o modelo.
- **Importância da Modularização:** A segunda versão demonstrou que um código bem modularizado facilita ajustes e depuração, mesmo quando o pipeline encontra problemas. Isso reforça a importância de investir em boas práticas de organização desde o início.
- **Limitações Práticas:** A primeira versão conseguiu bons resultados iniciais, mas dificuldades na análise de erros limitaram os aprendizados adicionais. Isso ressaltou a importância de planejar ferramentas de visualização e análise desde o início do projeto.

Conclusão

Este projeto destacou tanto os acertos quanto as limitações na implementação de redes neurais convolucionais (CNNs) para a classificação de imagens de faces. A **primeira versão** foi capaz de convergir e apresentou resultados satisfatórios, mas sua normalização dos dados foi inconsistente, o que limitou a eficiência do treinamento. Além disso, a ausência de ferramentas para análise detalhada de erros restringiu o entendimento mais

profundo das limitações do modelo. Já a **segunda versão**, apesar de apresentar uma organização e modularização superiores no código, enfrentou falhas críticas na normalização, o que comprometeu completamente o treinamento e impediu qualquer avaliação concreta.

Para superar as limitações encontradas, seria necessário revisar e corrigir o processo de normalização, garantindo a escalonamento adequado dos valores dos dados em ambas as versões. Ampliar o conjunto de treinamento com técnicas de aumento de dados também seria uma medida essencial para melhorar a capacidade de generalização do modelo. Além disso, implementar ferramentas que facilitem a análise de erros, como a identificação e visualização de imagens classificadas incorretamente, ajudaria a diagnosticar e resolver problemas específicos do modelo. Ajustes na arquitetura, como explorar configurações alternativas de dropout e alterações nos hiperparâmetros, poderiam contribuir para melhorar o desempenho global.

Finalmente, uma maior automatização do pipeline, com a modularização de etapas e validações integradas, garantiria maior eficiência no desenvolvimento e na manutenção do projeto. Com essas ações, o modelo poderia superar as limitações atuais, proporcionando resultados mais robustos e insights mais significativos para aplicações futuras.

Referências

GÉRON, Aurélien. *Mãos à Obra: Aprendizado de Máquina com Scikit-learn, Keras e TensorFlow*. 2. ed. Rio de Janeiro: Alta Books, 2021.