Universidade Federal do Rio Grande do Norte
Departamento de Engenharia Elétrica
ELE0606 - TOPICOS ESPECIAIS EM INTELIGENCIA ARTIFICIAL

Multi Layer Perceptron

Alunos: Pedro Artur F. Varela de Lira e Gutembergue Ferreira da Silva
Professor: José Alfredo Ferreira Costa

Dezembro
2023

Conteúdo

1 Introdução e Contexto

O Multi Layer Perceptron (MLP) é um dos modelos fundamentais no campo da inteligência artificial e do aprendizado de máquina. Trata-se de uma classe de redes neurais artificiais composta por múltiplas camadas de neurônios, organizadas em uma arquitetura com pelo menos uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída. Cada neurônio em uma camada está conectado a todos os neurônios na camada seguinte, e essas conexões são caracterizadas por pesos que são ajustados durante o processo de treinamento. O MLP é reconhecido por sua capacidade de aprender e modelar relações complexas entre variáveis de entrada e saída, sendo amplamente aplicado em tarefas de classificação, regressão e reconhecimento de padrões.

As Redes Neurais Convolucionais (CNNs) são um tipo especializado de redes neurais projetadas principalmente para processamento e análise de dados de grade, como imagens. Elas se destacam por suas camadas de convolução, que aplicam filtros para extrair características específicas da entrada, preservando a relação espacial entre os pixels. As CNNs são compostas por camadas de convolução, seguidas por camadas de pooling, que reduzem a dimensionalidade dos dados, e frequentemente incluem camadas totalmente conectadas para a classificação final. Devido à sua capacidade de aprender automaticamente características hierárquicas e invariantes a translações, as CNNs revolucionaram a área de visão computacional, sendo amplamente utilizadas em tarefas como reconhecimento de objetos, segmentação de imagens e detecção de padrões em diversas áreas, desde medicina até veículos autônomos.

Nesta atividade, irá ser realizada uma comparação entre um modelo de Multi Layer Perceptron padrão com duas camadas escondidas e um modelo de Convolutional Neural Network. Utilizaremos uma estrutura multitarefa em cascata profunda usando diferentes recursos de "submodelos" para cada um aumentar seus pontos fortes correlacionados, algoritmo impliementado na biblioteca MTCNN.

MTCNN é uma biblioteca python (pip) escrita pelo usuário ipacz do Github, que implementa o artigo Zhang, Kaipeng et al. "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks." IEEE Signal Processing Letters 23.10 (2016): 1499–1503. Crossref. Web.

A base de dados escolhida para este estudo possui característica única:

- Yale Face: é um conjunto de dados amplamente utilizado na área de visão computacional e reconhecimento facial. Este banco de dados é conhecido por fornecer imagens de faces sob diferentes condições de

iluminação, expressões faciais e fundos. Ele foi originalmente criado para fins de pesquisa e avaliação de algoritmos de reconhecimento facial. Essa base de dados tem sido bastante explorada em pesquisas acadêmicas e projetos industriais para avaliar e desenvolver algoritmos de reconhecimento facial. Esses conjuntos de dados desempenham um papel importante no treinamento e na avaliação de modelos de aprendizado de máquina para tarefas relacionadas a faces, como identificação facial e análise de expressões faciais.

Este relatório está organizado da seguinte forma: na seção seguinte, discutiremos brevemente a metodologia do trabalho, o pseudocódigo, os resultados obtidos e, por fim, serão apresentadas conclusões e discussões.

2 Metodologia

Utilizou-se a liguagem de progração Python, inserida em um ambiente de Jupyter Notebook disponibiizada pelo Kaggle. As principais etapas do código incluem:

- **Análise exploratória de dados**: essa parte consistiu na abordagem estatística e gráfica para examinar o conjunto de dados, identificar padrões, tendências, relações e anomalias, tendo como objetivo principal compreender a estrutura dos dados, resumir suas principais características e insights, e formular hipóteses ou direções para análises mais avançadas. O Yale Face Database é composto por 165 imagens em grayscale de 15 pessoas. São 11 imagens por pessoa com expressões diferentes.

- **Construção do Conjunto de Dados**: Pré-processar os dados, incluindo a remoção de valores ausentes e a codificação de variáveis categóricas, se necessário; dividir os dados em conjuntos de treinamento e teste para avaliar o modelo.

- **Setup dos Dados de Teste e Treinamento**: Preparação e organização dos dados para treinar, testar e avaliar o modelo. Nessa etapa foi feita toda a extração das faces.

- **Treinamento da Rede Neural Convolucional**: Aqui foram feitas algumas aplicações das operações de convolução para extrair características e padrões significativos das entradas, reduzindo a necessidade de pré-processamento manual. Essas redes aprendem automaticamente

hierarquias de características, tornando-as altamente eficazes em tarefas de visão computacional, como a classificação em questão.

- **Treinamento da MLP Padrão**: Aqui, o modelo será treinado com uma MLP padrão com duas camadas escondidas.
- **Avaliação dos Treinamentos**: É necessário avaliar o modelo com as métricas acurácia e matriz de confusão.

3 Pseudocódigo

O pseudocódigo abaixo reflete as principais etapas do código implementado no experimento, desde a importação das bibliotecas até a análise dos resultados de acurácia e mse.

- Importações
  - Importar bibliotecas necessárias
- Leitura da Imagem
  - Carregar imagem de um arquivo específico no Yale Face Database
  - Converter pixels da imagem para um formato RGB
  - Exibir a imagem original
- Detecção de Rostos
  - Criar um detector de rostos utilizando MTCNN
  - Detectar rostos na imagem utilizando o detector MTCNN
  - Exibir a imagem com caixas delimitadoras ao redor dos rostos detectados
- Extração de Rostos
  - Definir funções para extrair rostos de imagens e carregar o conjunto de dados
- Pré-processamento do Conjunto de Dados
  - Listar arquivos no diretório do conjunto de dados
  - Criar um DataFrame com informações sobre os arquivos
  - Dividir o conjunto de dados em conjuntos de treinamento e teste

- Carregamento e Divisão do Conjunto de Dados
  - Carregar imagens do conjunto de treinamento e teste
  - Definir parâmetros e diretórios para salvar as imagens do conjunto de treinamento e teste
- Definição do Modelo de Aprendizado Profundo
  - Definir um modelo sequencial do Keras com camadas convolucionais e densas
  - Compilar o modelo utilizando uma função de perda e otimizador específicos
- Treinamento do Modelo
  - Treinar o modelo usando o conjunto de treinamento e validar com o conjunto de teste (para os dois métodos - MLP e CNN)
  - Salvar o modelo treinado
- Avaliação do Modelo
  - Avaliar o desempenho do modelo utilizando métricas como precisão, recall e matriz de confusão
- Visualização dos Resultados
  - Gerar gráficos para visualizar a precisão e a perda ao longo das épocas de treinamento
  - Exibir a matriz de confusão para avaliação visual do desempenho do modelo
- Salvamento do Modelo
  - Salvar o modelo treinado em um arquivo
- Teste do Modelo
  - Escolher aleatoriamente uma imagem do conjunto de teste
  - Realizar a previsão do modelo na imagem escolhida
  - Exibir a imagem e o resultado da previsão (classe detectada e confiança)

É importante ressaltar que o pseudocódigo não foi implementado para funcionar efetivamente como solução do problema, mas sim como um meio de representar facilmente as etapas principais realizadas no processo. O código para o projeto completo e o resultados estão no link: `https://github.com/pedro-varela1/Arquivos_ELE-606/blob/main/MLP/yaleface-MLP-ele606.ipynb` (para o MLP) e `https://github.com/pedro-varela1/Arquivos_ELE-606/blob/main/MLP/yaleface-ele606.ipynb` (para o CNN).

4 Resultados

Para realizar a validação geral do modelo, foram realocados aproximadamente 70% da base de dados Yale Face para treinamento e aproximadamente 30% para teste e foram medidos a acurácia e erro médio quadrático.

Abaixo, vemos os resultados do treinamento, validação e teste para o modelo feito com a MLP de duas camadas escondidas de 512 neurônios cada.

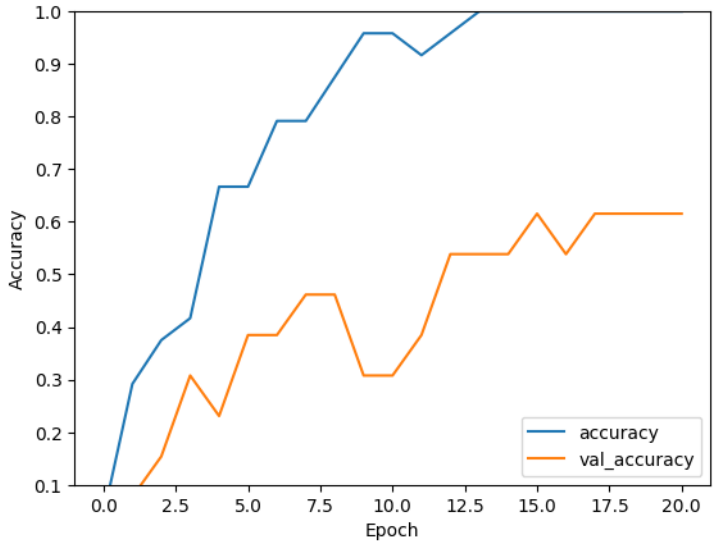

Figura 1: Evolução da Acurácia do modelo em relação ao tempo (MLP).

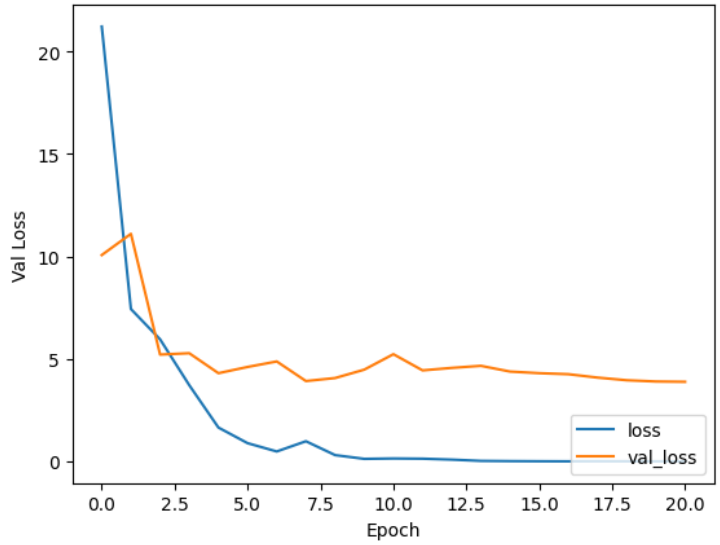

Figura 2: Evolução das perdas do modelo em relação ao tempo (MLP).

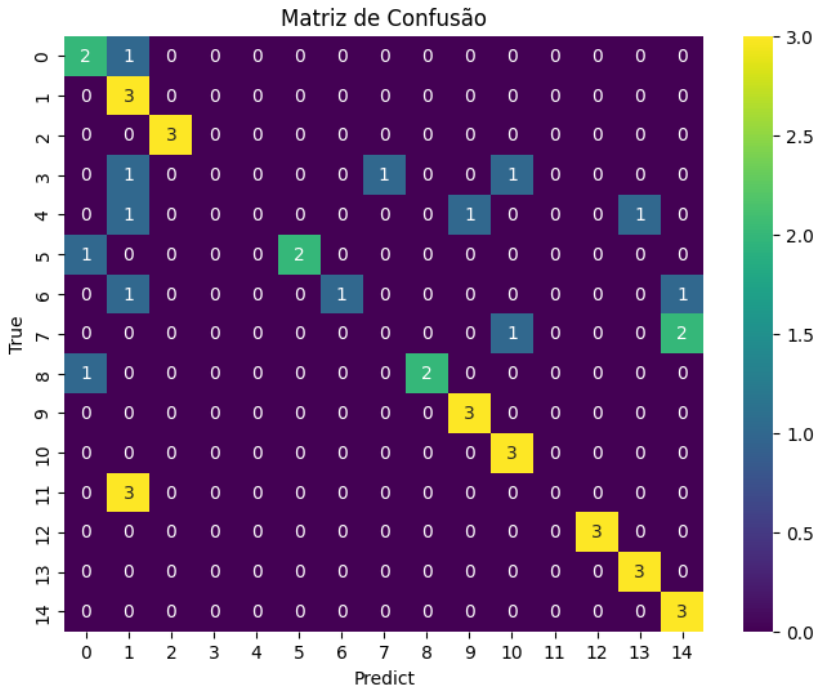

Figura 3: Matriz de Confusão (MLP).

Abaixo, vemos os resultados do treinamento, validação e teste para o modelo feito com a rede neural convolucional.

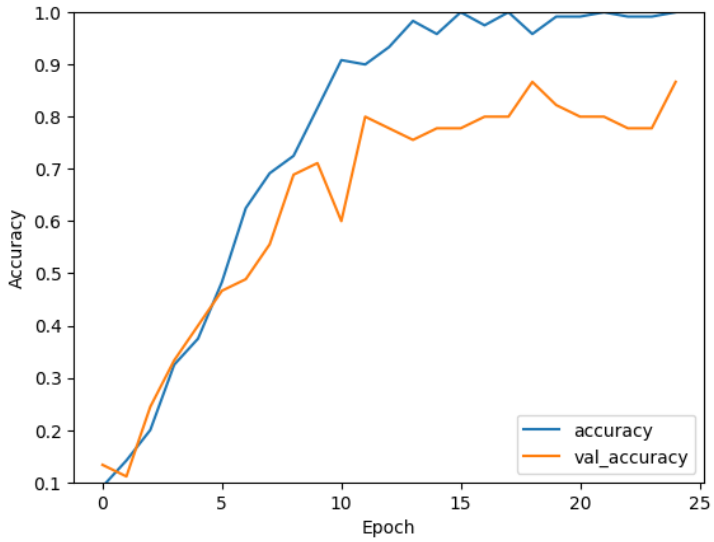

Figura 4: Evolução da Acurácia do modelo em relação ao tempo (CNN).

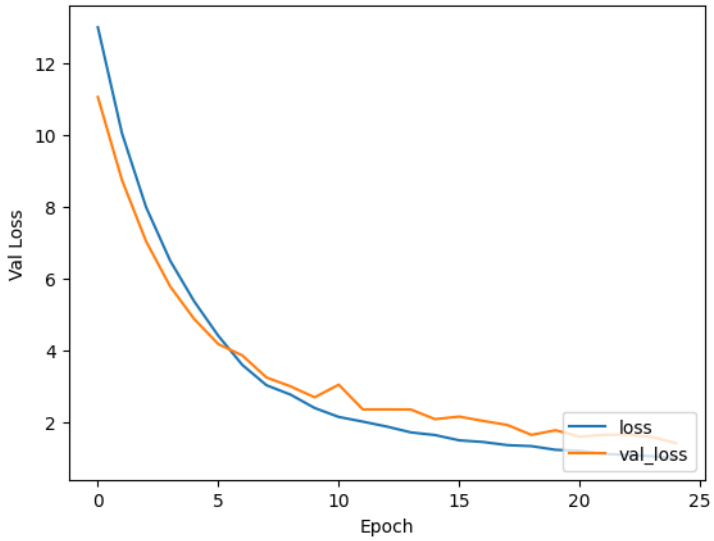

Figura 5: Evolução das perdas do modelo em relação ao tempo (CNN).

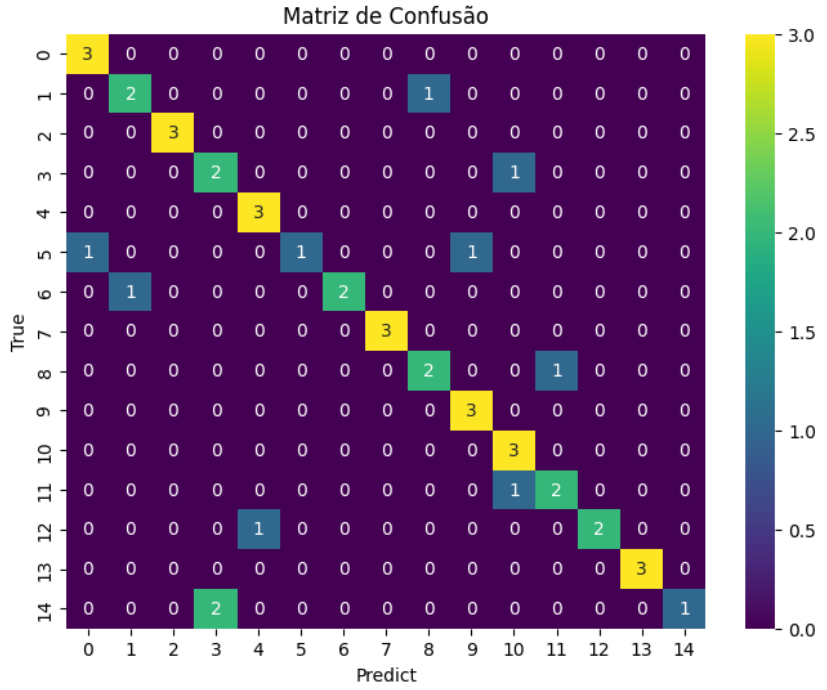

Figura 6: Matriz de Confusão (CNN).

5 Conclusão e Discussões

O modelo MLP padrão obteve acurácia de 100% no treinamento, mas apenas 62% nos dados de validação. Na matriz de confusão, percebe-se a ineficácia do modelo para algumas *labels* e a eficácia para outras, sendo muito instável. A soma desses fatores nos mostra um claro *overfitting* desse modelo, não sendo capaz de generalizar o treinamento em dados não vistos. Esse comportamento era esperado para uma rede neural MLP padrão, já que não consegue captar os *features* necessários de uma imagem sem as operações de convolução.

Já a rede neural convolucional obteve os mesmos 100% de acurácia do treinamento e 78% de acurácia nos dados de validação, além dos erros na validação também serem mais otimizados do que na MLP padrão. Além disso, na matriz de acurácia, vemos uma não dependência do modelo no tipo de *label* utilizada, já que o CNN obteve menor confusão.

Os resultados obtidos entre o modelo MLP padrão e a rede neural convolucional demonstram claramente a diferença na capacidade de generalização entre essas abordagens. Enquanto o MLP apresentou um desempenho perfeito no treinamento, seu rendimento nos dados de validação foi significati-

vamente menor, revelando uma falha crucial na capacidade de extrapolar o conhecimento adquirido. A instabilidade evidenciada na matriz de confusão indica uma ineficácia para certas categorias, ressaltando a falta de generalização do modelo para dados não vistos. Esse comportamento, esperado para um MLP padrão, reforça a importância das operações de convolução na extração de características necessárias para reconhecimento de padrões em imagens.

Por outro lado, a rede neural convolucional apresentou não apenas uma acurácia de treinamento elevada, mas também um desempenho mais robusto nos dados de validação, evidenciando uma capacidade superior de generalização. A redução significativa dos erros na validação, em comparação com o MLP padrão, reflete a eficácia do CNN na aprendizagem de padrões mais complexos e na capacidade de reconhecer diferentes características nas imagens. Além disso, a menor confusão observada na matriz de acurácia destaca a independência do modelo em relação aos diferentes tipos de rótulos utilizados, reforçando sua capacidade de lidar com uma variedade de classes sem comprometer seu desempenho. Esses resultados ressaltam a superioridade do CNN na extração de características relevantes para a tarefa de classificação de imagens.

Bibliografia

Carolina Bento (Towards Data Science). Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis. Disponível em: `https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-co` Acesso em: 06/12/2023.

IBM. What are convolutional neural networks?. Disponível em: `https://www.ibm.com/topics/convolutional-neural-networks`. Acesso em: 06/12/2023.

RAYEN BEN FATHALLAH (Kaggle Notebook). yale_face_database using cnn. Disponível em: `https://www.kaggle.com/code/rayenbenfathallah/yale-face-database-using-cnn`. Acesso em: 06/12/2023.