Universidade Católica de Santa Catarina Unidade Joinville

Curso: Engenharia de Software

Disciplina: Inteligência Artificial

Alunos: Ícaro Caldeira Botelho, Gustavo Henrique Martins, Maruan Biasi El Achkar, Ricardo Falcão Schlieper

Relatório do Projeto: Imitador de Emojis

1.Título

Imitador de Emojis: Classificação de Expressões Faciais com Visão Computacional

2.Objetivo da Atividade

-Desenvolver uma aplicação interativa que utiliza visão computacional para reconhecer expressões faciais humanas.

-Permitir ao usuário treinar seu próprio classificador com imagens personalizadas.

-Proporcionar uma experiência gamificada de imitação de emojis.

-Aplicar técnicas de aprendizado profundo (CNN) com *TensorFlow* para classificação de expressões faciais.

3.Introdução

A visão computacional é uma área da inteligência artificial que busca capacitar máquinas a interpretar e compreender imagens e vídeo, simulando a percepção visual humana (SZELISKI, 2010). O projeto “imitador de Emoji” explora essa tecnologia ao empregar redes neurais convolucionais (CNNs) para treinar um modelo capaz de reconhecer expressões faciais personalizadas do usuário.

Segundo Gonzalez e Woods (2018),, o processamento de imagens é fundamental para a preparação e análise de dados visuais, e com o avanço do *deep learning*, técnicas como CNNs tornaram-se indispensáveis. Conforme Goodfellow et al. (2016), as CNNs automatizam a extração de características relevantes das imagens, superando métodos manuais.

Este relatório apresenta os fundamentos, estrutura, implementação e desempenho do projeto, evidenciando sua utilidade como ferramenta lúdica e tecnológica.

4.Descrição das Atividades

4.1 Ambiente e Ferramentas Utilizadas

-Linguagem: Python 3.9+

-Bibliotecas: OpenCV, TensorFlow, NumPY

-Ambiente de Execução: Google Colab/local

-Hardware: Computador com webcam integrada

4.2 Estrutura do Projeto

-emojis/: Imagens dos emojis (ex: happy.png, angry.png)

-dataset/train/<emoji>/: Imagens do usuário para cada expressão

-treino\_persobalizado.py: Captura de imagens e treinamento do modelo

-main\_game.py: Execução do jogo interativo

-model.h5 / best\_model.h5: Arquivos com pesos da CANN treinada

4.3 Modos de Funcionamento

a) Modo de Treinamento (treio\_personalizado.py)

-O usuário é guiado a capturar múltiplas imagens por emoji.

-Imagens são salvas organizadamente.

-Treinamento automático do modelo CNN com aumento de dados.

-Callback aplicados: EarlyStopping, ReducelROnPlateau e ModelCheckpoint

B) Modo Jogo (main\_game.py

-Utiliza model.h5 para prever expressões.

-Em 5 rodadas, o usuário tenta imitar emojis com tempo limitado.

Sistema de pontuação avalia a precisão das imitações.

5. Resultados Obtidos

-O sistema treinado com ~100 imagens por classe atingiram acurácia satisfatória (~90%) das partidas.

-O reconhecimento facial via Haar Cascade apresentou bom desempenho em ambientes iluminados.

-O tempo de treinamento médio para um dataset pequeno foi inferior a 5 minutos.

6.Discussão

A experiência demonstrou a viabilidade de construir um sistema de reconhecimento facial simples e eficaz com recursos acessíveis. A personalização do dataset permite ao modelo aprenda características específicas do usuário, aumentando a precisão.

Desafios enfrentados:

-Ambientes com baixa iluminação afetaram a detecção facial.

-Poucas imagens por classe geram *overfitting*.

Melhorias futuras:

-Uso de modelos pré-treinados com MobileNet(transfer learning)

-Coleta automática de dados em diferentes condições de luz e ângulo.

-Aplicação em dispositivos móveis com TensorFlow Lite.

7.Conclusão

O Projeto Imitador de Emojis alia conceitos de visão computacional, aprendizado profundo e interatividade para criar uma aplicação educacional e divertida. A personalização do modelo e o uso de ferramentas livres como OpenCV e TensorFlow tornaram possível a construção de um classificador eficiente, com potencial para aplicações mais amplas em jogos, educação e acessibilidade.

8.Referências Bibliográficas

SZELISKI, Richard. Computer Vision: Algorithms and Applications. Springer, 2010.  
- GONZALEZ, Rafael C.; WOODS, Richard E. Digital Image Processing. Pearson, 2018.  
- GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. Deep Learning. MIT Press, 2016.  
- KASSANI, Shahrokh et al. Real-time Quality Inspection of Fruits using Deep CNNs. Cleaner and Responsible Consumption, 2, 2021.