DeepFake Detection using EfficientNetB4

# 🧠 Project Overview

DeepFakes are synthetic media in which a person in an existing image or video is replaced with someone else's likeness. This project aims to detect such manipulations using a pre-trained EfficientNetB4 model fine-tuned for binary image classification (Real vs Fake).

# 🗃️ Dataset

The dataset used contains two folders:  
- Real: Original, unaltered human faces.  
- Fake: AI-generated or manipulated faces.  
  
Images are structured in train, validation, and test directories and loaded using ImageDataGenerator with augmentation for generalization.  
  
If you're using a public dataset like DeepFake Detection Challenge (DFDC), mention it here: https://www.kaggle.com/c/deepfake-detection-challenge

# ⚙️ Model Architecture

The architecture is built using EfficientNetB4 from tensorflow.keras.applications, with the following:  
- Input size: (380, 380, 3)  
- Global Average Pooling  
- Dense layers with Dropout  
- Output layer: 1 neuron with sigmoid activation for binary classification  
  
The model is compiled using:  
- BinaryCrossentropy loss  
- Adam optimizer  
- Accuracy as the evaluation metric

# 🧪 Training

The model is trained using model.fit() on the training set with:  
- EarlyStopping and ModelCheckpoint callbacks  
- Data augmentation applied on training set  
- Real-time validation monitoring  
  
Training history is plotted and saved.

# 📊 Results

Sample Evaluation Metrics:  
- Accuracy: ~0.92  
- Loss curves plotted  
- Confusion matrix and classification report available  
  
Insert training and validation curves image here.  
Insert confusion matrix image here.

# 🔧 Requirements

tensorflow  
keras  
matplotlib  
numpy  
pandas  
opencv-python  
  
Install all dependencies with:  
pip install -r requirements.txt

# 📁 File Structure

Deepfake\_Detection/  
├── Deepfake\_Detection.ipynb  
├── dataset/  
│ ├── train/  
│ ├── val/  
│ └── test/  
├── model/  
│ └── efficientnetb4\_deepfake.h5  
├── plots/  
│ ├── accuracy\_loss.png  
│ └── confusion\_matrix.png  
└── README.md

# 🚀 How to Run

1. Clone this repo:  
git clone https://github.com/your-username/Deepfake-Detection-EfficientNetB4.git  
cd Deepfake-Detection-EfficientNetB4  
  
2. Place the dataset in the dataset/ folder.  
  
3. Run the notebook:  
jupyter notebook Deepfake\_Detection.ipynb

# 📌 Future Work

- Integrate video frame extraction to apply detection on videos.  
- Deploy the model with a web app using Flask or Streamlit.  
- Improve generalization using MixUp, CutMix, or adversarial training.

# 📣 Credits

- EfficientNet: https://arxiv.org/abs/1905.11946  
- DeepFake Dataset (Kaggle or custom)  
- Keras Applications  
- TensorFlow backend

# 📝 License

This project is licensed under the MIT License.