Detection of DeepFake Images Using PySpark and Convolutional Neural Networks

# Abstract

This study presents an approach for detecting DeepFake images using a combination of PySpark for data processing and a Convolutional Neural Network (CNN) for image classification. The methodology includes data acquisition, preprocessing, model training, evaluation, and the use of various performance metrics to validate the model's effectiveness.

# 1. Introduction

DeepFake technology has become increasingly prevalent, posing significant challenges in terms of misinformation and digital security. This research aims to develop an efficient and scalable method for detecting DeepFake images using machine learning techniques.

# 2. Data Acquisition and Preprocessing

## 2.1 Data Source

The dataset used in this study includes both real and DeepFake images, sourced from a publicly available dataset on Kaggle.

## 2.2 Data Import

The data was imported and extracted using Python's standard libraries for handling ZIP files and setting up the environment to mimic Kaggle's input structure.

## 2.3 Data Preprocessing

Preprocessing steps involved resizing images, normalizing pixel values, and splitting the dataset into training, validation, and test sets.

# 3. Model Development

## 3.1 Convolutional Neural Network (CNN) Architecture

The CNN model consists of several convolutional layers followed by max-pooling layers, dropout for regularization, and dense layers for classification.

## 3.2 Model Training

The model was trained using the training dataset, with early stopping implemented to prevent overfitting.

# 4. Model Evaluation

## 4.1 Accuracy and Loss Curves

The training process was monitored using accuracy and loss curves for both training and validation datasets.

## 4.2 Confusion Matrix

The confusion matrix was used to evaluate the model's performance on the test set.

# 5. Conclusion

This study demonstrates the effectiveness of using a CNN for the detection of DeepFake images. Future work will focus on enhancing the model's accuracy and exploring more advanced architectures and preprocessing techniques.

# References

- TensorFlow: [https://www.tensorflow.org/](https://www.tensorflow.org/)  
- Kaggle Dataset: [https://www.kaggle.com/](https://www.kaggle.com/)