International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

AI Based Deepfake Detection Using Convolutional Neural Networks and Transfer Learning

Ms. Sakshi Dattatray Jagtap[1], Mr. Kaustubh Ramesh Jagdale[2], Mr. Nishant Shivaji Jadhav[3], Ms. Sayali Jaganath Patil[4], Dr. Nilesh Thorat[5]

[1,2,3,4]Student, Computer Science and Engineering, MIT ADT university

Abstract

The rapid development of artificial intelligence and generative technologies has made it easier to create manipulated images and videos that appear realistic. Such manipulated media, commonly known as deepfakes, can be misused for misinformation, identity impersonation and digital fraud. This paper presents Deep Shield AI, an AI-assisted system designed for preliminary screening of potentially manipulated images and videos. The system integrates a web-based interface, a Python-based backend and a convolutional neural network-based inference pipeline. The application uses image preprocessing and visual classification to analyse uploaded media and generate prediction labels and model scores. It also provides processing status, processing time and an inconclusive state for uncertain predictions. The preliminary validation results of the installed model checkpoint indicate an accuracy of 67.3125%, manipulated-class recall of 44.75% and ROC-AUC of 76.055%. These values represent validation performance and not independent test results. The complete evaluation on a licensed held-out deepfake dataset has not yet been performed. Therefore, the system is intended to support preliminary media screening and does not guarantee the authenticity of digital content.

Index Terms— Deepfake Detection, Convolutional Neural Networks, Transfer Learning, EfficientNet-B0, Image Processing, Computer Vision, Artificial Intelligence.

I. INTRODUCTION

The rapid advancement of artificial intelligence has significantly changed the way digital images and videos are created and modified. Deepfake technology uses artificial intelligence techniques to generate or manipulate visual content, often making it difficult to distinguish between authentic and manipulated media.

Deepfake images and videos can be misused for spreading misinformation, identity impersonation, online fraud and other harmful activities. Traditional manual verification methods may be time-consuming and may not always identify sophisticated manipulations..Deep learning techniques, especially Convolutional Neural Networks, can be used to analyse visual patterns and identify features associated with manipulated content. Transfer learning can also help adapt an existing visual feature extraction model to a specific classification task.

This project introduces Deep Shield AI, an AI-assisted deepfake image and video screening prototype. It combines a web-based interface, a Python backend and a CNN-based inference pipeline using EfficientNet-B0.

The system is intended to support preliminary screening of potentially manipulated media. It is not intended to replace professional forensic investigation or provide a guaranteed authenticity decision.

The proposed Deep Shield AI system focuses on providing an accessible platform for preliminary deepfake screening. It combines a web-based application with a deep learning inference pipeline to analyse uploaded images and selected video frames. The system presents prediction results in a user-friendly manner while acknowledging the limitations of automated deepfake detection.

II. PROBLEM STATEMENT

The increasing availability of AI-based image and video generation tools has made manipulated media more realistic and accessible. Identifying such content manually is difficult and time-consuming.

There is a need for an automated screening system that can analyse uploaded images and video frames, process visual information through a deep learning model and present understandable results to users.

The proposed system aims to integrate a CNN-based detection pipeline with a web application to provide an accessible platform for preliminary deepfake screening.

III. OBJECTIVES

The main objectives of the proposed system are:

1. To develop an AI-assisted system for screening potentially manipulated images and videos.
2. To implement a CNN-based visual classification pipeline using the selected deep learning architecture.
3. To apply image preprocessing techniques for consistent model input.
4. To integrate the machine learning inference pipeline with a Python-based backend.
5. To develop a web-based interface for media upload and result presentation.
6. To display prediction labels, model scores, processing status and processing time.
7. To provide an inconclusive status for uncertain predictions.
8. To evaluate the system using appropriate deepfake datasets and standard performance metrics.

IV. RELATED WORK

Research on deepfake detection has increased due to improvements in face manipulation and synthetic media generation.

Rössler et al. introduced Face Forensics++, a benchmark dataset containing different facial manipulation techniques. Their work provides an important foundation for evaluating automated facial manipulation detection methods [1].

Li et al. introduced Celeb-DF, a challenging dataset containing high-quality deepfake videos. The dataset highlights the importance of evaluating detection systems on realistic manipulated content [2].

Tolo sana et al. presented a survey of face manipulation techniques and detection methods. Their work discusses different manipulation approaches, datasets and challenges in deepfake detection [3].

Tan and Le introduced Efficient Net, a convolutional neural network architecture that uses compound scaling to balance network depth, width and resolution. Efficient Net provides the architectural foundation for the EfficientNet-B0 baseline used in this project [4].

Based on these research directions, the proposed Deep Shield AI prototype integrates an EfficientNet-B0-based inference pipeline with a web application for image and video screening.

V.PROPOSED SYSTEM

The proposed system is named Deep Shield AI. It provides a web-based interface through which users can submit supported image or video files for analysis.

The system consists of three main components:

A. Frontend Module

The frontend is developed using Next.js, React, TypeScript and Tailwind CSS. It provides the user interface for media submission and displays the analysis response.

B. Backend Module

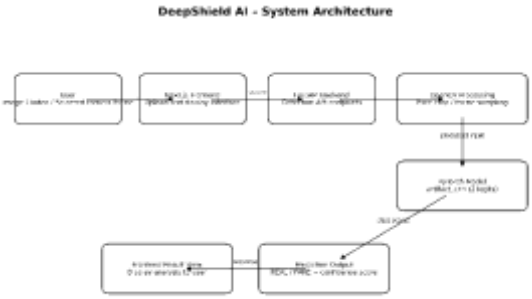
The backend is developed using Python and Fast API. Uvicorn is used to run the API service. The backend receives uploaded media, coordinates preprocessing and performs model inference.

C. Deep Learning Module

The inference pipeline uses Py Torch and EfficientNet-B0. The model processes visual input and generates class scores corresponding to authentic and manipulated categories.

OpenCV is used for image and video processing.

Deep Shield AI is designed as an integrated application in which the frontend communicates with the backend through API requests. The backend handles uploaded media, performs the required preprocessing and invokes the available deep learning model for inference. The results are returned to the frontend, where prediction labels, model scores and processing information are displayed. The system also supports an inconclusive result when the prediction does not meet the configured confidence thresholds.



VI. METHODOLOGY

The proposed system follows a sequence of media input, preprocessing, model inference and result presentation.

A. Media Input

The user uploads an image or supported video through the web interface. The backend receives the uploaded file and performs input handling.

B. Image Preprocessing

The input image is resized to 224 by 224 pixels. ImageNet-style preprocessing is applied before passing the image to the model. Consistent preprocessing is important for maintaining compatibility between model training and inference.

C. CNN-Based Feature Extraction

The system uses EfficientNet-B0 as its current CNN baseline. The architecture processes the visual input and extracts relevant image features for classification.

The exact transfer learning configuration, including pretrained weights, frozen layers, fine-tuned layers and classifier head, must be documented from the actual model training and checkpoint configuration.

D. Classification

The model produces class scores for the authentic and manipulated categories. The inference service interprets these scores according to the configured decision thresholds.

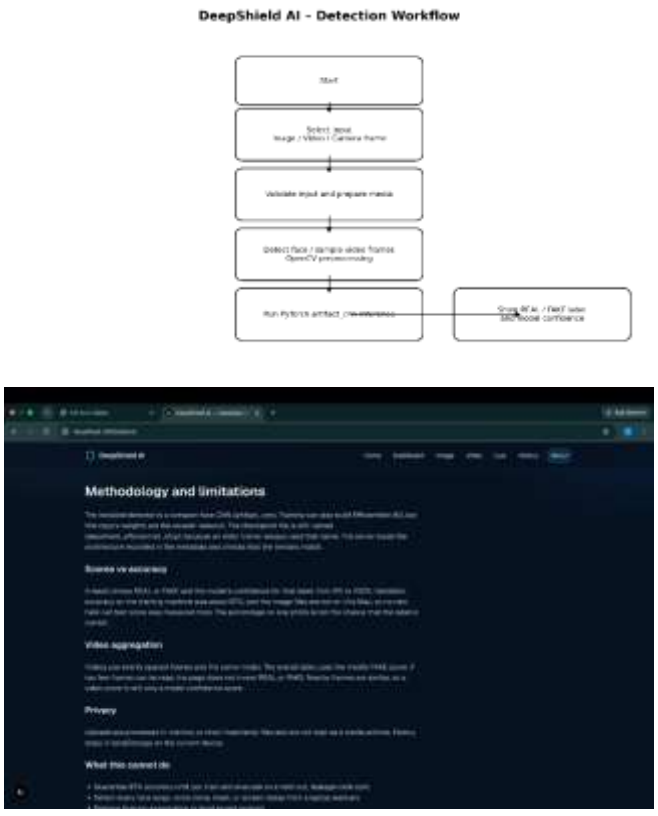
E. Uncertainty Handling

The system retains an inconclusive status for scores that fall between the configured decision thresholds. This prevents the application from forcing every uncertain input into a definitive category.

The thresholds must be calibrated and validated using a separate validation dataset.

F. Video Processing

For video input, selected frames are extracted and processed individually using the image-based inference pipeline. A reliable video-level prediction requires a separately validated frame sampling and aggregation strategy.



Methodology

VII. SYSTEM IMPLEMENTATION

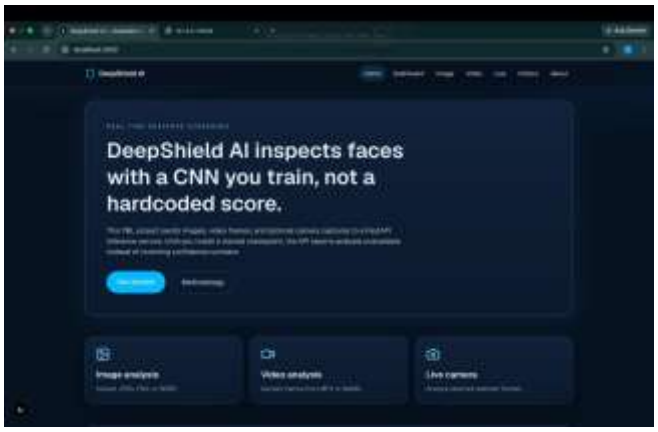
The project is implemented using the following technologies:

1. Frontend: Next.js, React, TypeScript and Tailwind CSS.
2. Backend: Python, Fast API and Uvicorn.
3. Deep Learning: Py Torch and EfficientNet-B0.

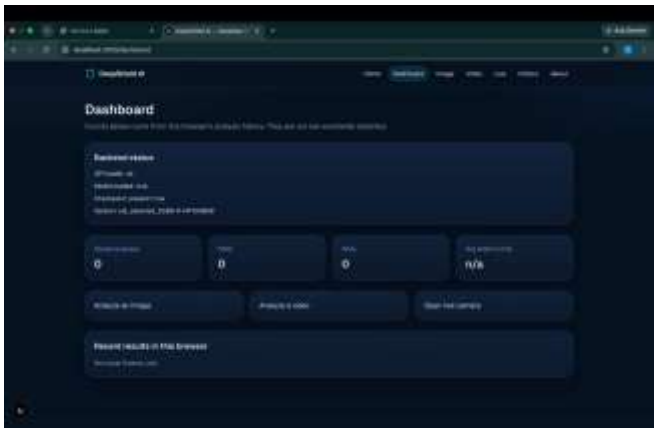
4. Media Processing: OpenCV.
5. Development Tools: Visual Studio Code, Cursor and Python virtual environment.

The frontend communicates with the Fast API backend for media analysis. The backend handles uploaded media and invokes the model inference workflow.

The application presents the prediction label, model score, processing status and processing time through the user interface.The frontend and backend communicate through API requests to provide an integrated media analysis workflow. The frontend manages media submission and result presentation, while the backend handles file processing and model inference. OpenCV is used for image and video processing, and the machine learning pipeline generates classification outputs using the available model checkpoint.



Home Page



Dashboard



Image upload page

VIII. RESULTS AND VERIFICATION

The current implementation of Deep Shield AI demonstrates the integration of the frontend application, backend API and machine learning inference pipeline.

The installed model checkpoint produced the following preliminary validation results:

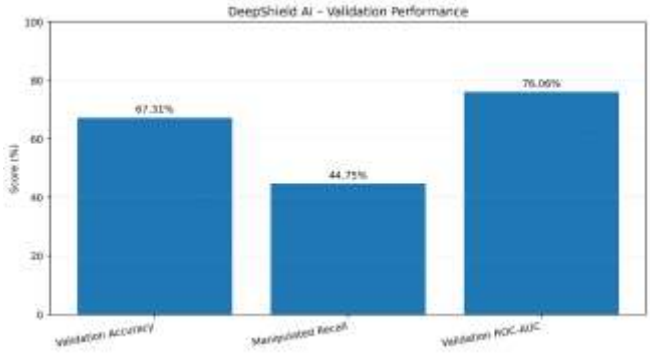
- Validation Accuracy: 67.3125%
- Manipulated-Class Recall: 44.75%
- ROC-AUC: 76.055%

These values represent validation performance and should not be interpreted as independent test results or guaranteed real-world detection accuracy.

The application supports image analysis, video frame processing and prediction result presentation. It displays prediction labels, model scores, processing status and processing time. An inconclusive status is also provided for uncertain predictions.

The software verification process includes backend testing, frontend testing, type checking, linting and production build verification.

The complete held-out test evaluation has not yet been performed because the original dataset image files were unavailable in the development environment. Therefore, further experimental evaluation using a complete licensed deepfake dataset is required to establish the model's generalization capability.



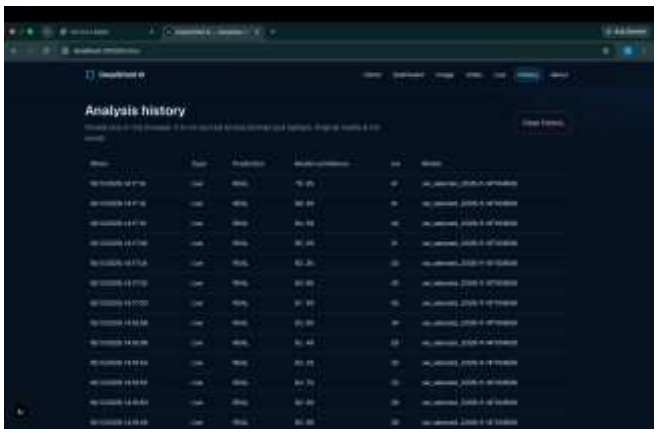


Image Analysis

IX. DATASET AND EVALUATION PROTOCOL

The project includes a dataset manifest containing 16,000 records, balanced between authentic and manipulated media samples.

To reduce data leakage during dataset preparation, source-based grouping is considered for training, validation and testing. Samples belonging to the same source video should be assigned to the same split using the source_id field. This approach helps prevent frames from the same source video from appearing in different dataset splits.

The project is designed to support licensed deepfake datasets such as Face Forensics++ and Celeb-DF. However, the original dataset image files were not available in the development environment for independently performing the complete training and testing process.

The installed model checkpoint has preliminary validation results, but the complete held-out test evaluation remains pending. Therefore, the reported validation metrics are presented separately from independent test performance.

Further evaluation using the complete dataset is necessary to measure the model's accuracy, precision, recall, F1-score and ROC-AUC under independent testing conditions.

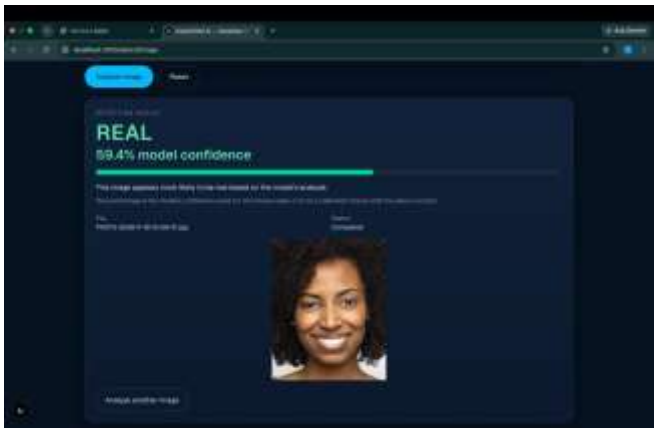


image Analysis Result – REAL

X. LIMITATIONS AND FUTURE WORK

Although DeepShield AI provides an integrated platform for preliminary deepfake screening, the current implementation has several limitations.

1. The installed model has limited preliminary validation performance and requires further improvement.
2. The model has not been fully evaluated on an independent held-out deepfake test dataset.
3. Face detection methods may fail to identify small, profile or partially visible faces.
4. Video-level predictions based on frame sampling and score aggregation require further validation.
5. Webcam analysis may be affected by lighting conditions, camera quality and replay attacks.
6. Model confidence scores are not calibrated probabilities of prediction correctness.
7. Detection performance may decrease when processing media generated using previously unseen manipulation techniques.

Future work will focus on training and evaluating the model using larger and more diverse licensed datasets. Additional improvements may include stronger face detection methods, confidence calibration, cross-dataset evaluation, explainable AI techniques and optimized model deployment.

The system processes user-provided images and videos for preliminary analysis. Therefore, secure file handling, restricted access to uploaded content and appropriate data-retention policies are important considerations for future development.

XI. CONCLUSION

This paper presents Deep Shield AI, an AI-assisted prototype system for preliminary deepfake image and video screening. The system integrates a web-based frontend, a Python-based Fast API backend, OpenCV media processing and a deep learning-based inference pipeline.

The implementation demonstrates an integrated workflow for media submission, preprocessing, model inference and result presentation. The application provides prediction labels, model scores, processing information and an inconclusive status for uncertain predictions.

The preliminary validation results indicate that further model improvement and comprehensive evaluation are necessary. The complete independent test evaluation has not yet been performed. Therefore, the current system does not claim guaranteed deepfake detection accuracy or replace professional digital forensic investigation.

Future work will focus on improving model performance, evaluating the system on independent licensed datasets, calibrating confidence scores and strengthening real-world detection capabilities**.**

XII. REFERENCES

1. A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies and M. Nießner, "Faceforensics++: Learning To Detect Manipulated Facial Images," In *Proceedings Of The Ieee/Cvf International Conference On Computer Vision (Iccv)*, 2019, Pp. 1–11, Doi: 10.1109/Iccv.2019.00009.
2. Y. Li, X. Yang, P. Sun, H. Qi And S. Lyu, "Celeb-Df: A Large-Scale Challenging Dataset For Deepfake Forensics," In *Proceedings Of The Ieee/Cvf Conference On Computer Vision And Pattern Recognition (Cvpr)*, 2020, Pp. 3204–3213, Doi: 10.1109/Cvpr42600.2020.00327.
3. R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales And J. Ortega-Garcia, "Deepfakes And Beyond: A Survey Of Face Manipulation And Fake Detection," *Information Fusion*, Vol. 64, Pp. 131–148, 2020, Doi: 10.1016/J.Inffus.2020.06.014.
4. M. Tan And Q. V. Le, "Efficientnet: Rethinking Model Scaling For Convolutional Neural Networks," In *Proceedings Of The 36th International Conference On Machine Learning*, Pmlr, Vol. 97, Pp. 6105–6114, 2019.
5. Fastapi Documentation. [Online]. Available: Https://Fastapi.Tiangolo.Com/
6. Py Torch Documentation. [Online]. Available: Https://Pytorch.Org/Docs/
7. Opencv Documentation. [Online]. Available: Https://Docs.Opencv.Org/
8. Next.Js Documentation. [Online]. Available: Https://Nextjs.Org/Docs