

DeepFake Detection Using CNN Model

SATHVIKA^{*1}, SEEMA^{*2}, VIDWATH^{*3}

1,2 Student, 3 Assistant Professor, Department of Computer Science and Engineering Shree Devi Institute of Technology,
Mangaluru, Karnataka, India
Visvesvaraya Technological University

Abstract

Deepfake technology has made it possible to create images and videos that look very similar to genuine media. Because of this, it is not always easy to identify manipulated content just by looking at it. This project presents a web-based system called the DeepFake Forensic Intelligence System for the preliminary analysis of suspected images and videos. The system uses image preprocessing, face detection and a MobileNetV2-based Convolutional Neural Network (CNN) to classify media as REAL or DEEPFAKE. For videos, selected frames are analysed and the individual results are combined to obtain a video-level result. The system also generates SHA-256 information, an Evidence ID, analysis-history records and downloadable reports. On a local validation set of 1,000 images, the retrained model achieved 79.8% classification accuracy. The system is mainly intended for academic demonstration and preliminary screening and should not be treated as a final legal or forensic authentication tool.

Index Terms— DeepFake Detection, Convolutional Neural Network, MobileNetV2, Face Detection, Computer Vision, Video Analysis, SHA-256, Digital Forensics, Flask.

I. INTRODUCTION

Recent developments in generative technology have made it easier to create realistic-looking fake images and videos. In some cases, changes to a person's face, identity or expression are difficult to notice by normal visual inspection. The problem can become more difficult after an image or video is resized or compressed, as commonly happens when media is shared online.

Deepfake detection is therefore an important area in computer vision and digital-media forensics. Earlier research has looked at different kinds of facial manipulation and has used CNNs, face artifacts, temporal information and other visual features for detection [1]–[4]. These studies also show that detector performance can depend on the dataset, manipulation method, compression and conditions in which the detector is tested.

In this project, we developed a web application that can be used to examine suspected images and videos. After a file is uploaded, it is checked and prepared for analysis. Face detection is used to provide supporting information, while a trained MobileNetV2-based CNN performs the main real-or-deepfake classification. For video files, selected frames are analysed instead of sending every frame through the model.

The project also keeps basic information about the submitted evidence. Along with the prediction, the application records file information, generates a SHA-256 value and Evidence ID, maintains analysis history and provides a downloadable report. These features make it easier to document and repeat an analysis during academic testing. The prediction is still considered a preliminary result and not a final forensic conclusion.

II. LITERATURE REVIEW

FaceForensics++ introduced a large benchmark containing different types of manipulated facial images and became an important dataset for research on deepfake detection [1]. Li and Lyu studied artifacts caused by face warping and showed that these inconsistencies can be useful when detecting manipulated faces [2]. The DeepFake Detection Challenge dataset further demonstrated the importance of using diverse data when evaluating deepfake detectors [3]. CNNs are useful for this type of problem because the network can learn different visual patterns from the input image. Transfer learning can also reduce the amount of training required when developing a model for a specific task. MobileNetV2 was selected in this project because it is designed to provide useful image features with comparatively lower computational requirements [4].

Other researchers have proposed compact networks, recurrent models and methods for detecting different forms of facial manipulation [5]–[8]. These works show that a model trained on one type of data may behave differently when it receives new

manipulation techniques or media with different characteristics.

In a practical application, the classification result alone is not enough to describe the file that was analysed. For this reason, our system includes an application layer that records evidence information, analysis history and reports together with the CNN result.

III. PROBLEM STATEMENT

The increasing amount of synthetic visual content creates a need for simple tools that can help with the initial examination of suspicious images and videos. Checking a large number of files manually can take considerable time. On the other hand, a model-only system may provide a prediction without keeping useful information about the analysed file.

This project addresses these issues by developing a web application that accepts image and video evidence, performs CNN-based analysis, provides supporting face information, samples video frames and stores basic evidence details so that an analysis can be documented and repeated.

IV. OBJECTIVES

- To develop a web-based application for preliminary deepfake analysis.
- To analyse both image and video evidence.
- To detect face regions and provide them as supporting information.
- To preprocess the input and apply a MobileNetV2-based CNN for binary classification.
- To analyse selected video frames and combine their results.
- To generate SHA-256 based evidence information and an Evidence ID.
- To maintain analysis history and provide report generation.
- To handle invalid or unsupported files in a clear manner.

V. PROPOSED SYSTEM

A. System Architecture

The proposed system is divided into a number of stages. The user first uploads an image or video through the web interface. Flask receives the file and checks whether it is supported. The media is then processed, face detection is performed and the input is passed to the MobileNetV2-based CNN. After classification, the result is combined with evidence information such as the SHA-256 value and Evidence ID. The completed analysis can also be stored in history and included in a report.

System flow: User / Browser ☐ Flask Web Application ☐ File Validation and Media Processing ☐ MobileNetV2-based CNN ☐ REAL / DEEPFAKE Result ☐ SHA-256 + Evidence ID + Report + History

B. Image Analysis

When an image is uploaded, the application first checks the file format. The image is converted to RGB and resized to 224×224 pixels, which is the input size used by the trained CNN. Face detection is performed

separately so that the result page can show information about the detected faces. The saved Keras model then produces the binary output used by the application to display REAL or DEEPFAKE. The analysis record also contains the file name, face count, SHA-256 value and Evidence ID.

C. Video Analysis

The video module does not analyse every frame of the video. Instead, it reads the video information and selects frames at configured intervals. Each selected frame is processed in the same general way as an image. The frame-level predictions are then combined to obtain the final video classification. This reduces the amount of computation required for the prototype. However, if a manipulation appears only between two sampled frames, it may not affect the final result.

D. Evidence Integrity and Reporting

For each uploaded file, the system calculates a SHA-256 hash. This hash works as a file-level fingerprint because changing the file contents will normally produce a different hash value. An Evidence ID is also generated and stored with the analysis. Previous analyses can be viewed through the Evidence Vault, and the reporting feature creates a downloadable record containing the result and related evidence information.

VI. METHODOLOGY

A. Dataset Preparation and Training

The model was trained using labelled real and fake facial images arranged in separate training and validation folders. During training, augmentation techniques such as horizontal flipping, small rotations, zoom and contrast changes were used. These changes provide the model with slightly different versions of the training images and reduce dependence on one particular appearance of a face.

MobileNetV2 was used as the main feature-extraction network. Transfer learning was carried out in two stages. In the first stage, most of the MobileNetV2 layers were kept fixed while the classification layers were trained. In the second stage, some deeper layers were made trainable and a smaller learning rate was used for fine-tuning. Early stopping, checkpoints and learning-rate reduction were used during training to retain a suitable model state.

B. Preprocessing and CNN Inference

During prediction, the input image is converted to the required format and resized to 224×224 pixels. The processed image is then given to the saved Keras model. The binary model output is converted into the REAL or DEEPFAKE label shown by the application.

Face detection is currently used as additional information rather than as a separate voting mechanism. The CNN analyses the resized uploaded image, and the present implementation does not combine predictions from individually cropped faces.

C. Web Application and Validation

The server side of the application was developed using Python and Flask. OpenCV is used for image and video processing and face detection, while TensorFlow/Keras is used for loading the model and performing inference. HTML and CSS are used to build the web interface.

Testing was carried out for different parts of the application, including file upload, model loading, face detection, SHA-256 generation, report creation and history access. Unsupported file types are rejected before they reach the analysis stage.

VII. ALGORITHM

- Start the Flask application.

- Accept an image or video evidence file.
- Check whether the submitted file is valid.
- Detect face regions and collect supporting information.
- Resize and prepare the input for the CNN.
- Run the MobileNetV2-based CNN.
- Convert the model output into REAL or DEEPFAKE.
- For video, analyse selected frames and combine the frame results.
- Calculate SHA-256 and generate an Evidence ID.
- Store the analysis details and prepare the report.
- Display the final result to the user.

VIII. RESULTS AND DISCUSSION

The completed application brings media upload, CNN prediction and evidence recording together in one workflow. The retrained Keras model was tested directly on the project's local validation set, which contains 1,000 labelled images belonging to the real and fake classes.

The model achieved a validation accuracy of 79.8%. The average model output was 0.305 for the fake subset and 0.766 for the real subset. These results show that the model was able to separate the two classes to a useful extent on the local validation data. However, the result should not be assumed to represent performance on every type of deepfake or on media obtained from other sources.

For image analyses, the application stores the file name, detected face count, SHA-256 value, Evidence ID and other supporting information. For videos, information about the sampled frames is also retained. These features help place the CNN prediction within a simple evidence-management workflow.

TABLE I. MEASURED VALIDATION RESULT

Number of validation images: 1,000 Fake subset mean model output: 0.305 Real subset mean model output: 0.766 Classification accuracy: 79.8%

IX. LIMITATIONS

- The performance of the detector depends on the quality and variety of the training data. Therefore, the reported local accuracy should not be taken as proof that the model will work equally well on other datasets.
- The result may change for manipulation techniques that were not included during training, heavily compressed files, or inputs where the face is difficult to detect.
- Video analysis uses only selected frames. A short manipulation that appears between the sampled frames may not be included in the final result.
- Face detection currently provides supporting information, while the CNN works on the resized uploaded image rather than combining predictions from separate face crops.
- Audio deepfake detection, eye-reflection analysis and detailed micro-texture analysis are not included in the current validated implementation.

X. ADVANTAGES

- Provides one web-based workflow for image and video screening.
- Combines CNN classification with basic evidence identification and reporting.
- Uses SHA-256 information to help track submitted files.
- Maintains previous analyses through the Evidence Vault.
- Uses MobileNetV2, which is relatively lightweight for application-level inference.

XI. APPLICATIONS

- Preliminary screening of suspicious digital images and videos.
- Academic demonstrations of CNN-based computer vision.
- Documentation and organisation of analysed media.
- Prototype workflows for digital-media analysis and evidence reporting.

XII. FUTURE SCOPE

- Test the system with larger and more varied datasets containing different manipulation techniques.
- Use stronger temporal models for video-based deepfake detection.
- Improve the ability to handle manipulation methods that were not present in the training data.
- Add database-based case management and controlled evidence storage.
- Include more advanced forensic and media-provenance checks.
- Explore multimodal analysis, including audio and physiological features, after suitable training and validation.

XIII. CONCLUSION

This project presents a web-based system that uses a MobileNetV2-based CNN for the preliminary detection of deepfake images and videos. The system handles file upload, preprocessing, face detection, CNN inference and result display. For videos, selected frames are analysed and their predictions are combined to produce a video-level result.

The system also provides features for maintaining evidence information. Each analysis can be associated with a SHA-256 value, an Evidence ID, analysis history and a downloadable report. The retrained model achieved 79.8% accuracy on the local validation set of 1,000 images. The current system is intended mainly for academic demonstration and first-level screening. Future work can focus on larger datasets, improved video analysis and additional forensic features.

ACKNOWLEDGMENT

The authors would like to thank Shree Devi Institute of Technology, Mangaluru, for providing the academic environment and facilities required to complete this project. The authors also thank VIDWANTH, Assistant Professor, Department of Computer Science and Engineering, for providing guidance and support throughout the project.

REFERENCES

- [1] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies and M. Niethammer, "FaceForensics++: Learning to Detect Manipulated Facial Images," Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
- [2] Y. Li and S. Lyu, "Exposing DeepFake Videos By Detecting Face Warping Artifacts," arXiv preprint arXiv:1811.00656, 2018.
- [3] B. Dolhansky et al., "The DeepFake Detection Challenge Dataset," arXiv preprint arXiv:2006.07397, 2020.
- [4] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [5] D. Afchar, V. Nozick, J. Yamagishi and I. Echizen, "MesoNet: a Compact Facial Video Forgery Detection Network," IEEE International Workshop on Information Forensics and Security (WIFS), 2018.

- [6] D. Guera and E. J. Delp, “Deepfake Video Detection Using Recurrent Neural Networks,” 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2018.
- [7] M. Dang, F. Liu, J. Stehouwer, X. Liu and A. Jain, “On the Detection of Digital Face Manipulation,” Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
- [8] P. Korshunov and S. Marcel, “DeepFakes: a New Threat to Face Recognition? Assessment and Detection,” arXiv preprint arXiv:1812.08685, 2018.
- [9] TensorFlow, “TensorFlow Keras Documentation,” model loading, training and image preprocessing documentation.
- [10] OpenCV, “OpenCV Documentation,” image and video processing and computer-vision documentation.
- [11] Pallets Projects, “Flask Documentation,” web application framework documentation.

Copyright & License:

© Authors retain the copyright of this article. This work is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.