JCAI

Journal of Cognition and Artificial Intelligence
Vol. 01, No. 1, PP-13-17, JUNE 2025
https://doi.org/10.66512/jcai.1.1.2025.6

AI-Based Deep fake Detection

Muhammad Ajlal Haider[1], Uzair Izhar Khan[1], Muhammad Osama Masood[1], and Sarim Mashhood[1]

[1]*Sir Syed University of Engineering and Technology Karachi, Pakistan*

Correspondence Author: Muhammad Ajlal Haider (@ssuet.edu.pk)

Abstract

Deepfake technology raises questions regarding security and disinformation by making it possible to create incredibly lifelike fake photos and videos. In order to identify Deepfakes involving particular public figures, this paper suggests a real-time AI-based system that makes use of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. Following frame extraction and preprocessing (such as face detection and alignment), CNN-based spatial feature extraction and LSTM-based temporal analysis are carried out by our system. Included is an audio analysis module that looks at cloned voice patterns using CNN/RNN and spectrogram features. The entire pipeline is implemented as a web application, which uses adaptive preprocessing and data augmentation to support varying lighting conditions and video quality. We anticipate that the system will attain detection accuracy on par with the most advanced techniques found in the literature. The web interface lets you upload or stream stuff and get authenticity checks right away.

Index Terms: *Convolutional Neural Network, Deepfake Detection, Long Short-Term Memory, Multimodal Analysis, and Real-Time Video Forensics.*

I. INTRODUCTION

Right now, media's really important in shaping what people think and know. Because of that, it's tough to tell what's real anymore. Deepfakes—those fake videos and images made with AI that swap or change faces—are a threat to the truth and our trust [1]. Tech's getting better fast, especially with stuff like GANs, so Deepfakes are easier to fake and seem real. For example, Face Forensics++ shows that you can swap faces or copy expressions in real-time using stuff like Face Swap and Deepfakes [2]. They made fake frames from real videos and then checked them with Convolutional Neural Networks (CNNs) to see if they were fake. All this means we really need smart, automatic ways to spot Deepfakes, no matter how they're made or what the video is like [3].

Since Deepfakes are often used to mess with famous people, our project is all about creating a system that's good at spotting these kinds of attacks. Famous folks are easy targets for fake impersonations, which can change what people think, cause trouble, or ruin reputations [4]. To fix this, our system uses the latest in deep learning to look at both video and sound. CNNs check for weird stuff in the video, like strange-looking faces, bad skin texture, and messed-up eyes [5]. Then, Long Short-Term Memory (LSTMs) networks, watch for stuff that changes over time, like blinking weirdly [6], lips not matching the words [7], and faces moving strangely [8].

On top of all that, the system checks the audio for anything fishy. Fake voices, which often go with video Deepfakes, are found by looking at speech spectrograms [9]. CNNs and Recurrent Neural Networks (RNNs) work together to check these spectrograms for odd pitch, tone changes, and timing problems [10].

Our system is different because it looks at video and audio at the same time, instead of separately. Because of this, it can quickly figure out if a video is real or not, even if the video quality isn't great or the lighting is bad.

II. STATE OF THE ART

Deepfake detection research is exploring different ways to spot fakes, from using simple image-based CNNs to more complex models that look at both space and time. Researchers reviewed how deep learning is used, grouping methods by how they analyze the fake – by single images, over time, by sound, using both sound and video, or by looking at forensic clues [1]. They also point out problems like dealing with different sources, how well methods work when videos are compressed, and using them in real-time [3]. Some simpler methods, like those used by some researchers, only check individual frames and are good at spotting fakes in each image, but they don't look at how things change over time [11], and [12]. This means they can be fooled by really good fakes that are consistent from frame to frame. Better methods, like the CNN-RNN setup with PSO by some analysts [13] and the networks by another group [14] that watch for changes in space and time, do a better job at spotting these time-based problems, but they take more computing power. Most research focuses on what you can see, and not so much on combining sound and video to find fakes [9] and [10].

Our system takes those ideas and runs with them. It uses a CNN to grab spatial features, an LSTM to watch how things change over time, and a separate branch to analyze audio. By putting these together, it spots when things don't match up, like lips not syncing with speech [7]. Unlike those bulky 3D CNNs, our CNN→LSTM setup is both accurate and fast, great for using on the web in real-time.

We also make sure it works well in the real world by training it on different datasets and videos with things like compression, noise, and lighting changes [15]. To make it even faster, we use tricks like frame batching and quantization. This gives fact-checkers and media companies, something that actually works, not just something that looks good in a research paper.
So far we have discussed studies that have contributed to the Deepfake system [1], [10], [11], [13], and [14].

III. PROBLEM STATEMENT AND ITS PROPOSED SOLUTION

A. Problem Description

The increasing sophistication of Deepfake technology has introduced a serious threat to digital media authenticity particularly in the form of highly realistic but artificially generated images, videos, and audio Deepfakes are scary because they can fake public figures and mess with what people think. Tech like GANs has made its way too easy to create fake stuff that can fool almost anyone. This is bad news for people's reputations, trust in government, and even national security. The ways we have now to spot these fakes usually stink in real time. They only pay attention to how things look or sound. Plus, they can't handle bad video quality, bad lighting, or noisy backgrounds. So, we really need a better Deepfake detector. It should look at both video and sound, using fancy tech like CNN and LSTM networks to check video frames and spectrogram-based CNN-RNN models to analyze audio. It needs to work fast and be easy for anyone to use, so we can put it to work on websites and live streams.

B. Solution Framework

How the architecture of the Deepfake detector works is it mixes together some cool tech from different areas; computer vision, sound analysis, and some smart computer programs that can spot patterns. It's built to find fake stuff in videos, whether they're pre-recorded or live, and it's really good at spotting when someone's pretending to be a well-known person. For the video part, it uses a special kind of computer program called a CNN. This program is trained to spot weird stuff in faces. It's great at spotting small problems in images. For example, bad lighting, uneven faces, weird-looking skin, or misaligned facial features. These CNN programs process video frames using a bunch of filters to grab key details without overloading the system.

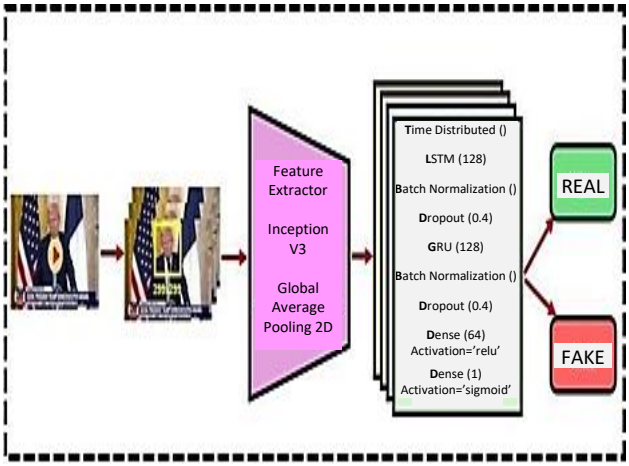

Figure 1: System Functional Diagram

The system uses Long Short-Term Memory (LSTM) networks to get a handle on time-based patterns and shifts in video frames. LSTMs are a good fit for sequential data and can spot weird stuff that happens over time, like strange blinking, shaky facial moves, and lips not matching what's being said. These kinds of time-based weirdness often show up when Deepfake creators use generative models. The system takes feature vectors from the CNN and runs them through LSTM layers. This lets it hold onto context from past frames when checking out stuff coming up. Thinking about the sequence helps the model get better.
To figure out if a live video is real or not, the system looks at both the video and the sound. It doesn't just watch the video; it checks the audio, too, for things like voice cloning or if someone messed with it. It turns the audio into spectrograms – kind of like visual sound maps that show how loud and high the sounds are as time passes. Then, a special network checks these maps for weird stuff in the voice, like tone, pitch, or timing problems. These problems usually show that the voice has been faked or cloned. By checking both video and audio, the system is better at spotting Deepfakes that try to trick you with both sights and sounds.
The model then uses a final layer to decide if the video is real or fake. It learns to make correct guesses by comparing its answers to the real ones and fixing its mistakes using a method called backpropagation. The Adam thingy helps to adjust during learning. We tried other gradient descent method at first to see which one performs better.
The system is built to work fast. It chops up video into small chunks and processes each one quickly using something like under two seconds on average. This makes it work in real-world web applications, media platforms, or streaming services where quick decisions are needed. People can upload videos or stream, and the system tells them right away if it thinks the video is real, even if the video quality isn't great or the audio is bad.

IV. PROPOSED METHODOLOGY

The Deepfake detection thing works like this: First, it grabs a bunch of videos. These videos star famous folks and come from places like YouTube. Some videos are real, and some are Deepfakes made with fancy tools that swap faces, copy expressions, and clone voices. This way, the system learns to spot both real and fake stuff.

Next, the system chops up each video into frames super quick. Then, it spots and lines up faces in each frame using tools like MTCNN or dlib. That way, it only focuses on the face part of the video. For the sound, it pulls out the speech parts.

We turned audio into visual representations like spectrograms or MFCCs. All images and audio were resized to fit standard scales. To make the system more resistant to problems and better at generalizing, it was trained with extra data, which comprised random changes to brightness, added noise, and fake video compression. This helps the system cope with differences in video quality, lighting, and resolution. A CNN was used to extract spatial features. It looks at each face image to spot small texture mistakes, mismatches, and face issues. We're going to try out architectures like Exception and ResNet. Transfer learning might be used to help things converge faster and perform better by grabbing pre-trained weights. The CNN is trained to figure out if each frame is real or fake just by looking at it. Deepfakes often screw up timing, something you can't see in just one frame. So, we run the CNN results through a LSTM network. This LSTM looks at sequences of frames and learns to spot weird motion, like unnatural blinking, face changes that happen too fast, or lips not matching the words. The audio side does the same thing: an LSTM checks audio sequences for cloned voices and weird timing. Then, the system puts the video and audio findings together to make a final call. This can be done by just combining the data or using another Neural Network layer to mix both types of info. The model then spits out a score or a simple real or fake label for the video. We train the system by feeding it real and fake videos that already have labels. To get it right, we use cross-entropy loss and Adam-like to cut down on mistakes. We might even try fancy tricks like Particle Swarm to make the model even better. Because it needs to work fast, we keep an eye on how long it takes to process stuff and try to speed things up. Things like model trimming or quantization can help reduce the load and make it faster on phones or websites. The whole setup has two parts: the deep learning part doing the heavy lifting and a website that people actually use. The backend, running in Python, takes the video, whether it's a file someone uploads or a live stream, grabs the frames, and sends them to the CNN-LSTM thingy. At the same time, the voice module checks the audio. Once it's done, the model returns its guess, along with how sure it is. The frontend is a website made with tools like Flask or Django. It's easy to use: people upload videos or paste a streaming link. The results pop up in real-time on a dashboard, maybe with boxes around Fake faces and graphs showing how sure the system is as it goes. To make sure it works well no matter the video quality, the system tweaks the video a bit first. It adjusts the brightness and contrast and resizes the frames to fit what the CNN expects. The model usually looks at images that are 224x224 pixels. It learns from videos that have things like different compression levels, resolutions, and lighting so it should work with both phone videos and pro recordings. For sound, we do a basic noise cleanup before pulling out key details, making things clearer and more accurate. The whole thing is made to be easily updated. The way it's built, we can easily add new improvements without having to rebuild everything from scratch. For, we could drop in extra classifiers or fancy setups like transformers to better understand how things change over time and catch long-term patterns. This design means the system can adapt as Deepfake tech and detection methods get better.

V. RESULTS AND DISCUSSION

We trained our models using both video and sound, labeling each clip as either real or fake. The video came from different places, and we turned the sound into something the computer could understand. We kept training the model and checking how well it did use accuracy – how often it got the right answer. We also used validation accuracy to see if it could handle video and sound, it hadn't seen before.

Below, we present the accuracy and validation accuracy values recorded over successive epochs during the training process:

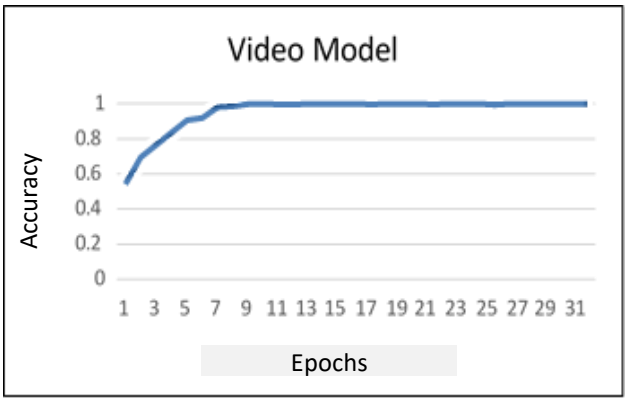

Figure 2: Video Model Training Accuracy

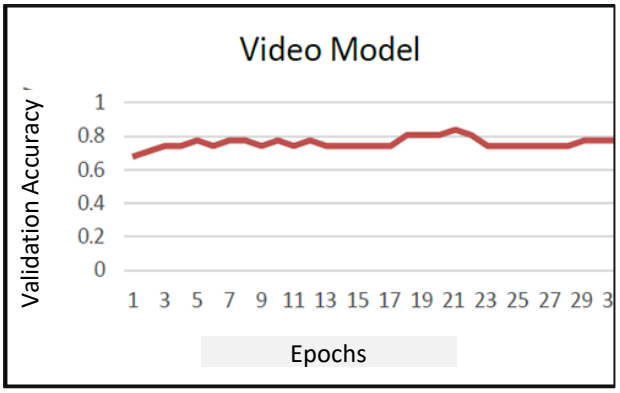

Figure 3: Video Model Validation Accuracy

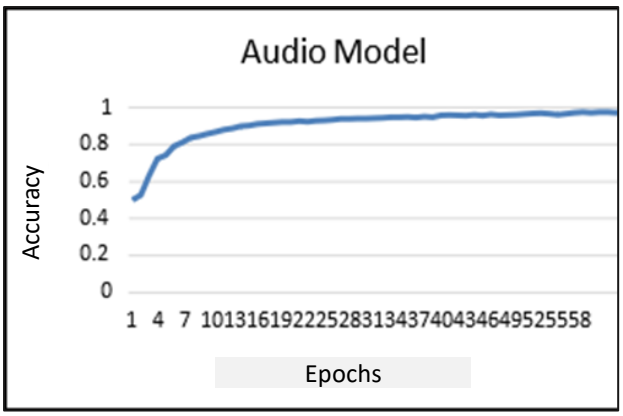

Figure 4: Audio Model Training Accuracy

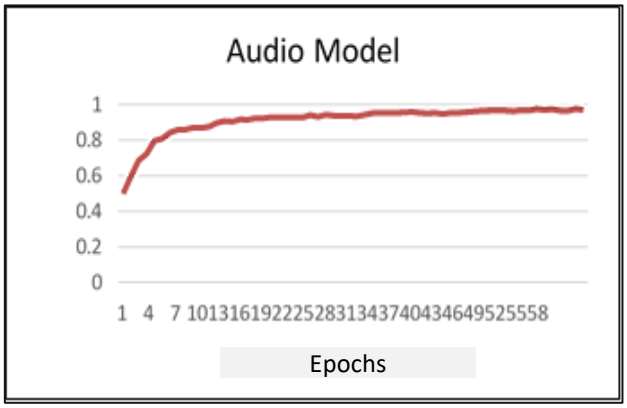

Figure 5: Audio Model Validation Accuracy

The training and validation accuracy keep getting better, which means the model is learning to tell real and fake stuff apart in different ways.

Here are the results (with labels) from the model (Figure 6- Figure 8):

Figure 6: Result----Video: REAL Audio: REAL

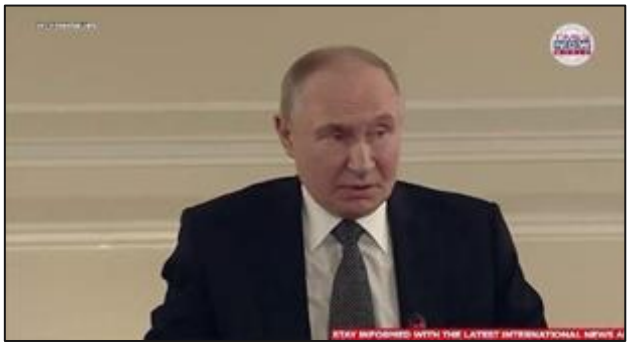

Figure 7: Result---- Video: FAKE Audio: REAL

Figure 8: Result---- Video: REAL Audio: FAKE

VI. CONCLUSION

Deepfakes are becoming more common, it's getting tough to tell what's real and what's fake, especially when they mess with both the video and sound. So, we built a system that spots Deepfakes of public figures by checking out the video and sound. We use CNNs to grab details of each v o frame and LSTMs to see if things stay consistent o time. This helps us catch fakes quickly and pretty a rately. Also, we look at the audio by using s trograms and MFCCs to find fake speech patterns. T s makes the whole system better at spotting Deepfakes. T nake it run fast, we improved how we prep the data and b ed up the backend with stuff like model pruning and q ntization. We've put together a straightforward website that lets people upload or stream videos and get them checked quickly. It should come in handy for reporters, fact-checkers, and social media folks. The design is also easy to build on, so we can add stuff like transformers or blockchain tracking down the road. Basically, this project gives us a useful and scalable way to catch deepfakes as they happen. This can assist people trust what they see on the internet.

Acknowledgment

The authors would like to thank the management of Sir Syed University of Engineering and Technology Karachi, Pakistan, for their support and assistance throughout this study.

Authors Contributions

All the authors equally contributed to this research study.

Conflict of Interest

The authors declare that there is no conflict of interest regarding the publication of this article.

Data Availability Statement

All data used in this study were collected from publicly available sources.

Funding

This research received no external funding.

References

[1] Heidari, A., Jafari Navimipour, N., Dag, H., & Unal, M. (2023). Deepfake Detection Using Deep Learning Methods: A Systematic and Comprehensive Review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*. https://doi.org/10.1002/widm.1520.

[2] Rössler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., & Nießner, M. (2019). FaceForensics++: Learning to Detect Manipulated Facial Images. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, South Korea, 1–11.

[3] Hosny, S. M. A. E., & Awad, A. I. (2020). Deepfake Detection Techniques: A Survey. *Proceedings of the International Conference on Computer Engineering and Systems (ICCES)*, Cairo, Egypt, 173–178.

[4] Hasan, H. R., & Salah, K. (2019). Combating Deepfake Videos Using Blockchain and Smart Contracts. *IEEE Access, 7*, 41596–41606.

[5] de Souza, G. B., da Silva Santos, D. F., Pires, R. G., Papa, J. P., & Marana, A. N. (2018). Efficient Width-Extended Convolutional Neural Network for Robust Face Spoofing Detection. *Proceedings of the 7th Brazilian Conference on Intelligent Systems (BRACIS)*, São Paulo, Brazil, 230–235.

[6] Li, Y., Chang, M. C., & Lyu, S. (2018). In Ictu Oculi: Exposing AI Created Fake Videos by Detecting Eye Blinking. *Proceedings of the IEEE International Workshop on Information Forensics and Security (WIFS)*, Hong Kong, 1–7.

[7] Agarwal, S., Farid, H., & Fried, O. (2020). Detecting Deep-Fake Videos from Phoneme- Viseme Mismatches. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Seattle, WA, USA, 660–661.

[8] Guera, D., & Delp, E. J. (2018). Deepfake Video Detection Using Recurrent Neural Networks. *Proceedings of the 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Auckland, New Zealand.

[9] Pham, H. T., Chambers, J. A., & Duc, N. M. (2022). End-to-End Audio Deepfake Detection. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Singapore, 8612–8616.

[10] Kosarkara, U., Sarkarkar, G., & Gedam, S. (2023). Revealing and Classification of Deepfakes Video's Images Using a Customized Convolution Neural Network Model. *Procedia Computer Science, 214*, 123–133. https://doi.org/10.1016/j.procs.2023.XX.

[11] Karandikar, A., Deshpande, V., Singh, S., Nagbhidkar, S., & Agrawal, S. (2020). Deepfake Video Detection Using Convolutional Neural Network. *International Journal of Advanced Trends in Computer Science and Engineering*, *9*(2), 1311–1315. https://doi.org/10.30534/ijatcse/2020/62922020.

[12] Korshunov, J., & Marcel, S. (2018). Deepfakes: A New Threat to Face Recognition? Assessment and Detection. ArXiv preprint, *arXiv*:1812.08685.

[13] Al-Adwan, A., Alazzam, H., Al-Anbaki, N., & Alduweib, E. (2024). Detection of Deepfake Media Using a Hybrid CNN– RNN Model and Particle Swarm Optimization (PSO) Algorithm. *Computers*, *13*(4), 99. https://doi.org/10.3390/computers13040099.

[14] de Lima, O., Franklin, S., Basu, S., Karwoski, B., & George, A. (2020). Deepfake Detection Using Spatiotemporal Convolutional Networks. *arXiv*. https://doi.org/10.48550/arXiv.2006.14749.

[15] Zhang, Z. (2019). *Detect Forgery Video by Performing Transfer Learning on Deep Neural Network* (PhD Dissertation). Department of Computer Science.