Detection of Deepfake Videos and Audios on Social Media Platforms

Charles Fross
cwf016@shsu.edu
M.S. Digital Forensics
Sam Houston State University

Qingzhong (Frank) Liu, Professor
liu@shsu.edu
Department of Computer Science
Sam Houston State University

Abstract— Deepfake technology, a rapidly evolving application of artificial intelligence, has enabled the creation of highly realistic yet synthetic multimedia content. While this innovation offers potential benefits in areas such as entertainment and education, its misuse has raised significant ethical and security concerns, including misinformation and financial fraud. This study evaluates the effectiveness of current deepfake detection methods, focusing on the Xception model for video detection and the LCNN model for audio detection, using a dataset composed of real-life and deepfake content. The dataset includes deepfakes generated by tools such as the Deepfake Offensive Toolkit and Haotian AI, a cutting-edge provider known for its high-quality outputs. Our findings reveal that the Xception model, while achieving 89.1% accuracy on control datasets, struggled to detect Haotian AI-generated deepfakes, misclassifying nearly all samples as authentic. This performance gap highlights the need for more diverse training datasets and advanced detection frameworks capable of addressing the nuances of emerging deepfake tools. Additionally, metadata changes caused by uploading and downloading content on social media platforms were found to have minimal impact on detection accuracy, challenging the feasibility of metadata-based detection approaches. This research underscores the limitations of current deepfake detection models and emphasizes the necessity for multimodal approaches and broader datasets to enhance robustness. The study's implications call for continued advancements in detection methods to keep pace with the growing sophistication of deepfake technologies.

Keywords—deepfake, Deepfake Offensive Toolkit, Faceswap, DeepFaceLab, Voicemod, Real-Time Voice Cloning, metadata analysis, social media, deepfake detection, deep learning, digital forensics

I. INTRODUCTION

A. *Context*

Artificial intelligence has been enhanced in recent years to include an application to multimedia known as "deepfakes". A deepfake is some form of media input that has been artificially transformed to mimic some training media to produce an output like the training media. Realtime video and/or audio as well as recorded video, audio, and images can all be used for the generation of deepfakes. Improvements have been made to the specific artificial intelligence algorithms used for generations of deepfakes to more closely resemble the training data used. With the advent of this application of artificial intelligence has come malicious use of deepfakes. Deepfake technology has been used to both make humorous content for social media popularity, and for scams [1]. Deepfakes can become harmful on social media sites when used for scams or misinformation. It has been reported that deepfakes have impacted elderly American citizens through fraud using deepfake technology to gain trust of the victim, resulting in a total loss of 3.4 billion USD in 2023 [43]. While metadata has been suggested as a tool for identifying deepfake content, this approach faces significant limitations. Alterations, encryption, or removal of metadata from deepfake media can render this method ineffective. Moreover, many social media platforms further complicate detection by removing or encrypting metadata, often as a privacy measure, making it challenging or even impossible to rely on metadata for identifying deepfakes. [37][38][39][40]. While efforts have been successful before to detect the use of deepfake-related scams, such as asking an attacker to make a gesture with their hand or face to reveal artifacts or errors of their software, recent developments in such scams have prompted legislation to be made to regulate such use of deepfakes, and further research to be conducted to improve existing detection methods [3][4][20].

This paper will evaluate the ability of current deepfake detection methods to detect deepfake media generated by known deepfake algorithms, and some samples from deepfakes generated by Haotian AI, a company whose deepfake software is reported to offer capabilities comparable to or exceeding those of DeepFaceLab and its live version [2]. We also aim to explore the impact of uploading both original videos and audios to social media sites on metadata for both real life content and deepfake content. The experiment will be conducted as follows. First, samples of video and audio will be utilized as input

for different deepfake algorithms. Second, the generated deepfake content is uploaded to different social media platforms and then downloaded for analysis of the changes to the metadata of the original videos and audio and the deepfake versions. Finally, a deepfake detection model will be trained and tested on the original and deepfake content.

B. *Technical Background*

B.1. *Deep Learning Fundamentals*. Deep learning, a subset of machine learning, relies on neural networks with multiple layers to extract complex patterns from data. Convolutional Neural Networks (CNNs), a specialized type of deep neural network, excel in processing spatial data like images or audio. By using convolutional and pooling layers, CNNs reduce dimensionality while amplifying significant features, such as edges, anomalies, and distortions, making them vital for both the generation and detection of deepfakes [21][22][27][53].

Mathematical Functions in CNNs:

1. Convolution Operation

$$(f * g)(x, y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f(i, j) \cdot g(x + i, y + j)$$

1. This operation extracts local features from input data by sliding a filter (a.k.a. a kernel) over the input matrix (image or audio).

2. Pooling:

$$P_{ij} = \max_{(m,n) \epsilon R_{ij}} (I_{m,n})$$

2. This function reduces spatial dimensions while preserving the most significant features in the region R_{ij}.

B.2. *Deepfake Generation Techniques*. Deepfake generation leverages neural network architectures, primarily autoencoders and GANs. B.2.a. *Autoencoders*. These models compress input data into a latent representation, then reconstruct it. When two decoders are trained on distinct inputs, they can generate outputs blending shared features of both, enabling realistic deepfake synthesis [23][28].

B.2.b. *Generative Adversarial Networks (GANs).* GANs consist of a generator, which produces synthetic outputs, and a discriminator, which evaluates their authenticity. The iterative adversarial process between the two improves the quality of outputs, often rendering them indistinguishable from genuine media [30].

B.3. *Deepfake Detection Methods*. Effective detection of deepfakes requires identifying artifacts or inconsistencies introduced during synthesis. B.3.a. *Artifacts and Behavioral Anomalies.* Research highlights common issues such as unnatural facial movements, warping, and audio-video desynchronization [30][31][32][33]. B.3.b. *CNN-Based Techniques.* CNNs excel at detecting such irregularities in visual data, leveraging their feature extraction capabilities [35].

Loss Function for CNN Models:

3. Both audio and video deepfake detection models used in this paper utilized categorical cross-entropy loss for classification tasks. This is defined as:

$$L = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

Where C is the number of classes, y_i is the true label (one-hot encoded), and $\hat{y}_i$ is the predicted probability for class i.

4. This loss function measures the divergence between the predicted class probabilities and the true labels, guiding the model to improve its predictions during training.

B.3.c. *GAN-Based Techniques.* While CNNs excel at detecting visual irregularities, GANs are commonly used in creating highly realistic fake data. Understanding the mathematical basis of these methods provides insight into their capabilities and limitations. GANs, like those used in Simswap, are central to the generation of deepfakes. The adversarial training process involves two neural networks: a generator and a discriminator [62].

1. Discriminator Loss:

$$L_D = -\mathbb{E}_{x \sim p_{data}}[\log D(x)] - \mathbb{E}_{z \sim p_z}\left[\log(1 - D(G(z)))\right]$$

5. The discriminator learns to distinguish between real samples (x) from the true data distribution (p_{data}) and fake samples ($G(z)$) generated by the generator.
2. Generator Loss:

$$L_G = -\mathbb{E}_{z \sim p_z}[\log D(G(z))]$$

6. The generator aims to produce fake samples that the discriminator classifies as real, effectively "tricking" it.

These loss functions represent the adversarial training process, which underpins the creation of realistic deepfakes in Simswap. Understanding these mathematical foundations

aids in designing detection models capable of identifying subtle artifacts generated during GAN training.

B.3.d. *Emerging Challenges*:

Advances in neural network algorithms reduce errors and anomalies, thereby complicating detection and necessitating innovative approaches, such as multimodal or transformer-based methods [30]

II. METHODOLOGY

A. *Dataset collection*

The beginning of the project required collecting a dataset to use for generating the deepfake content. The dataset chosen for the video deepfakes originated from a dataset on Kaggle labeled, "Deep Fake Detection (DFD) Entire Original Dataset," but was discovered later to have come from Google & JigSaw based on information on the FaceForensics++ Github repository [5][18][12]. This dataset contains videos of several different people all performing similar actions and in similar scenarios, which allowed for easy comparison of the effects of the deepfake tools used in this project. The primary reason why this dataset was selected was due to its sheer size, allowing for the dynamic selection of the quantity of videos required for this project. 200 video samples were chosen at random from this dataset. Some samples from the Haotian AI Telegram channel were selected for this dataset due to the cutting-edge algorithms that their deepfake tools have been reported to utilize and the capacity for their tools to produce outputs comparable to or exceeding the quality of industry-standard tools like DeepFaceLab. For the samples from Haotian AI, 28 video files showing samples of their deepfake tools were downloaded from their Telegram channel [6]. This inclusion allowed the project to evaluate deepfake detection methods against some of the most advanced and sophisticated deepfake technologies currently available, ensuring that the results are relevant to emerging real-world challenges in detecting high-quality deepfakes. For the audio deepfake dataset, 200 audio samples were chosen at random from two different datasets of male and female voice samples from Kaggle [7][8]. These audio samples were later preprocessed using FFmpeg and Librosa to ensure the normalization and consistency of the quality of each audio file. The preprocessing of the source audio files used for the generation of the deepfakes involved using Audacity to remove sections of audio heavily polluted with other individuals' voices (e.g., interviews, public speeches, music, etc., …), applause, or other noise. Audacity was also utilized to ensure the normalization of the source audio. Some audio source files were rejected due to insufficient media found of the voices of certain individuals or some source audio presented issues with finding enough substantial audio of the desired voice without background noise. All data collected was found from publicly available sources, ensuring compliance with ethical standards regarding various individuals desire for privacy or protection of intellectual property. An essential directory structure of "real-life-videos," "real-life-audios," "deepfake-videos," and "deepfake-audios" was created, with each folder containing subfolders for different social media sites. This structure was crucial for managing storage and tracking metadata changes for original and deepfake files. Ultimately, these datasets were fair enough to not only be used for this project but may be used in future projects.

B. *Deepfake Generation*

B.1. *Tools Used*. Multiple tools for video and audio deepfakes were considered and tested. The ethical, computational, and financial constraints informed tool selection. B.1.*a*. *Video Deepfakes*. For video deepfakes, tools like DeepFaceLab_DirectX12 and the Deepfake Offensive Toolkit were explored. The Deepfake Offensive Toolkit was chosen for its efficiency and quality, allowing batch processing of 200 deepfake videos. DOT utilized the GAN-based model Simswap. DeepFaceLab_DirectX12 is a deepfake generation software which allows the utilization of the CPU or acceleration of the deepfake generation process with the GPU [19]. This software proved to be very user-friendly with available YouTube tutorials to assist in a simple video deepfake process [17].

B.2.*b*. *Audio Deepfakes*. For audio deepfakes, tools included "Voicemod," "PlayHT," and "Real-Time Voice Cloning." Real-Time Voice Cloning generated 200 audio files using a pretrained model. Voicemod and PlayHT are two different kinds of commercial off-the-shelf tools which provide unique features to select for audio deepfake generation for free or subscription-based [14][15]. Voicemod allows users to produce alterations of the user's own voice with a selection of free filters to transform their voice. PlayHT's paid options allowed for the cloning of a provided sample of a source voice. "Real-Time Voice Cloning" was the primary tool used, generating deepfakes in a timely manner using a pretrained model [13]. B.2.b.i. *Mathematical Basis of the RNN Model in Real-Time Voice Cloning*. RTVC relies on recurrent neural networks (RNNs), the same neural

network which are useful in video deepfake detection when using their advanced variants like long short-term memory (LSTM) or gated recurrent units (GRU). RNNs are designed to process sequential data, such as audio waveforms or text, by maintaining a hidden state that captures temporal dependencies [53][54].

1. Input Representation
    - RTVC takes a Mel-Spectrogram of the audio as input:

$$S(f,t) = \text{Log-Mel}(f,t)$$

Where f represents frequency bands and t represents time steps.

2. RNN Hidden State Update
    - For each time step t, the RNN computes the hidden state h_t using the input x_t (e.g., Mel-Spectrogram features at time t) and the previous hidden state h_{t-1}:

$$h_t = \sigma(W_{hx}x_t + W_{hh}h_{t-1} + b_h)$$

        - W_{hx}: Weight matrix for the input
        - W_{hh}: Weight matrix for the hidden state.
        - b_h: Bias term.
        - σ: Activation function (e.g., tanh or ReLU).
3. Output Generation
    - The RNN produces an output y_t at each time step:

$$y_t = W_{hy}h_t + b_y$$

        - W_{hy}: Weight matrix for the output layer.
        - b_y: Bias term.
4. Enhanced RNN Variants
    - RTVC often employs advanced RNN variants to address vanishing gradient problems:
        - Long Short-Term Memory (LSTM):

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$$

$$f_t = \sigma\big(W_{xf}x_t + W_{hf}h_{t-1} + b_f\big)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$

$$h_t = o_t \odot \tanh(c_t)$$

            - i_t, f_t, o_t: Input, forget, and output gates, respectively.
            - c_t: Cell state that captures long-term dependencies.
    - Gated Recurrent Units (GRUs):

$$z_t = \sigma(W_{xz}x_t + W_{hz}h_{t-1} + b_z)$$

$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + b_r)$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tanh(W_{xh}x_t + r_t \odot W_{hh}h_{t-1} + b_h)$$

            - z_t, r_t: Update and reset gates, respectively.
5. Application to Voice Cloning
    - Encoder-Decoder Framework:
        - The encoder extracts a fixed-dimensional embedding $z_{speaker}$ from the input voice, capturing speaker-specific features.
        - The decoder generates the Mel-Spectrogram of the cloned voice using the RNN or its variants (LSTM/GRU) to maintain temporal coherence.
6. Output Processing
    - The generated Mel-Spectrogram is converted back to audio using a vocoder, such as WaveNet or HiFi-GAN, which synthesizes high-quality waveforms.

The RNN model in RTVC captures temporal dependencies in audio, ensuring that the generated voice matches the natural rhythm and intonation of the target speaker. While the pretrained RTVC model achieves fast cloning, its reliance on simpler embeddings and models results in lower quality compared to tools like PlayHT, which may use more advanced architectures or higher-quality training datasets. Consequently, the use of the pretrained model in RTVC led to a lower quality audio deepfake than what was produced with PlayHT. Additional preprocessing involved tools like FFmpeg, Whisper, and Librosa for extracting mel-spectrograms, and transcribing the audio. To briefly explain FFmpeg is used to standardize audio properties such as channels, sample rate, and bit depth to ensure consistency across datasets. This process is crucial for extracting reliable features for deepfake generation and detection. Below is a mathematical explanation of

how FFmpeg was used to preprocess the audio dataset. B.2.b.ii. *Audio Preprocessing with FFmpeg*. FFmpeg applies transformations to the audio signal as seen below [57][58].

1. Resampling to f_s =22,050 Hz:

$$y_{resampled}[n] = \sum_{k=-\infty}^{\infty} y[k] \cdot h(n-k)$$

Where $y[k]$ is the original audio signal and h is the interpolation filter.

2. Channel Conversion (e.g., stereo to mono):

$$y_{mono}[n] = \frac{1}{2} \cdot (y_{left}[n] + y_{right}[n])$$

3. Bit Depth Adjustment to 16-bit: Quantizes the signal $y[n]$ into 2^{16} discrete levels.

B.2.b.iii. *Mel-Spectrogram Extraction (with Librosa)*. Mel-Spectrograms represent the frequency content of an audio signal on the perceptually scaled Mel frequency axis [59][60].

1. Compute the Short-Time Fourier Transform (STFT):

$$X(f,t) = \int_{-\infty}^{\infty} x(\tau) \cdot w(t-\tau) \cdot e^{-j2\pi f\tau} d\tau$$

where w(t) is a window function.

2. Map frequencies to the Mel scale using triangular filters:

$$M(f_m,t) = \sum_f |X(f,t)|^2 \cdot W(f_m,f)$$

$W(f_m,f)$ is the Mel filter bank.

3. Apply the logarithm:

$$\text{Log-Mel}(f_m,t) = \log(M(f_m,t) + \epsilon)$$

Mel-Spectrograms emphasize frequency bands important for human perception, enhancing the ability of machine learning models to detect subtle manipulations in deepfakes. B.2.b.iv. *MFCCs: Mathematical Basis.* Mel-Frequency Cepstral Coefficients (MFCCs) provided a compact representation of the spectral envelope, derived from the Mel-Spectrogram [61]:

1. Compute the logarithm of the Mel-Spectrogram:

$$\log(M(f,t))$$

2. Apply the Discrete Cosine Transform (DCT) to reduce dimensionality:

$$MFCC_k = \sum_{n=1}^{N} \log(M_n) \cdot \cos\left[k\left(n-\frac{1}{2}\right)\frac{\pi}{N}\right]$$

- M_n: Mel-Spectrogram coefficients.
- N: Number of Mel bands.
- k: Index of the MFCC.

By focusing on the lower-order coefficients, MFCCs capture the broad spectral shape, essential for identifying voice characteristics used in Real-Time Voice Cloning and audio deepfake generation. Mel-Spectrograms and MFCCs serve as input features for machine learning models. They transform raw audio into a format that emphasizes relevant characteristics, improving the quality and efficiency of deepfake generation and detection. B.2.b.iv. *Transcription with Whisper*. Whisper transcribes audio into text, aiding in evaluating coherence and alignment between synthesized speech and its intended content. Whisper uses a transformer model trained on spectrograms [55][56].

- Input spectrogram:

$$S(t,f) = \text{Log-Mel}(f,t)$$

- Self-attention mechanism:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where Q, K, V are query, key, and value matrices, and d_k is the dimension of the key.

- Output: Decoded text using beam search or similar algorithms.

The transcription of the audio ensures the generated audio matches the expected linguistic structure, a critical aspect of high-quality deepfake synthesis.

B.2.c. *Tools Not Used*. Tools like Tacotron2, DeepFaceLab_DirectX12, and Faceswap faced computational, compatibility, or ethical constraints. Faceswap, for instance, required excessive processing time and encountered technical issues. DeepFaceLab_DirectX12 was not selected as the primary video deepfake generation tool due to technical constraints which prevented automation of the batch scripts. Deciding to not use PlayHT as the primary audio deepfake generation tool was focused on the inability to automate the process of generating all audio deepfakes, which was unhelpful for time constraints. B.3. *Challenges Faced*. Computational limitations, such as insufficient GPU power and

system disconnections, were mitigated by using a university GPU workstation and alternative software. These adjustments improved processing efficiency and quality. When Faceswap was allowed to successfully complete the generation of one deepfake video, the quality was easy to detect due to the hardcoded face mask used for detecting the face of the target subject. This confirmed the suspicion of the software's lack of trustworthiness for a high quality deepfake generation tool. B.4. *Deepfake Creation Process*. Video deepfake generation involved extracting source images, training models, and batch processing. Audio deepfakes followed a multi-step process, including transcription, spectrogram generation, and model training.

C. *Data Dissemination and Metadata Analysis*

The third step of the project required uploading the deepfake and real-life or original (RLO) content to different social media platforms. A variety of social media platforms were considered, but some required to be avoided or were not capable of uploading or downloading all the content from the project. The metadata extracted from the control and uploaded content showed substantial changes which reflect the privacy policies and efficiency of the file transfer algorithms selected by different social media platforms based on their level of popularity. The metadata analysis also showed a substantial change from the RLO content to the deepfake content, confirming what some sources have said about the feasibility of detection based on its metadata. The analysis also showed a significant difference in the scale of alterations of the metadata of specifically deepfake content uploaded to social media. This may suggest that social media platforms have implemented anti-deepfake protection. However, this would only confirm an earlier hypothesis of the infeasibility of detection based on metadata due to the altered state post-upload.

C.1. *Uploading to Social Media Platforms.* The social media sites attempted were the sites which resulted in the largest successful uploads. Uploading RLO video files in batch was successful in different numbers depending on the site. Of all social media sites which were attempted to upload content to per information found about their capacity for uploading videos or audios, the most consistent social media sites which allowed the most successful amount of content uploaded and downloaded content were Facebook, Telegram, and SoundCloud. Other social media sites, which may have allowed some content or almost the entire amount of RLO and deepfake content, were compartmentalized due to issues relating to file size, file media type allowable, ease of use, upload frequency standards, and batch upload or download capacity. Instagram, like Facebook, allowed batch uploading up to ten or more videos at a time and to upload videos up to a certain file size, but was not feasible for downloading all content due to reasons unknown. As with all social media sites attempted except for Telegram and Facebook, attempts to upload a large majority of the deepfake video content to Instagram were unsuccessful due to exceeding the file size limit. The deepfake videos which were successfully uploaded to, and downloaded from, social media sites other than Facebook and Telegram were either produced by the Faceswap software after insufficient iterations were allowed or were from the samples collected from the Haotian AI Telegram channel. Some sites, such as Truth-Social, Tumblr, Twitter, and Threads, could only allow upload frequencies either one or up to four videos at a time. Other social media sites that were either not feasible for uploading and downloading much of the content or could not allow for the upload or download video or audio media types, are illustrated (see Fig. 1). Facebook, while not possible to upload audio, allowed for the successful upload and download of both RLO and deepfake videos thanks to the batch download feature in the Account Center. Telegram appeared to have no total file size or frequency limit granted enough internet speed to upload all requested content.

C. 2. *Metadata Extraction and Analysis.* The initial analysis of metadata showed that uploading the content to almost all social media sites had substantial impacts on the metadata of each file. However, some more interesting meaning arrived from creating different scripts to search for any trends in the metadata files. The metadata viewer software, "EXIFTOOL," was used to create reports of the metadata of both the RLO and deepfake files. After creating these reports of the metadata of each file, RLO or deepfake and uploaded or not, Python scripts were written to make comparisons of different classes of the content. Comparisons were made of the non-uploaded content against the uploaded content, RLO videos to deepfake videos, and RLO audios to deepfake audios. Metrics which were used to compare content with others were primarily changes to existing attributes, addition of attributes, and removal of attributes. This graph (see Fig. 3) revealed how the most changes in metadata from uploading to

social media came from deepfake videos uploaded to social media sites. This change may be due to the file size and unique dimensions of deepfake videos as well as the quality. After these comparisons were made, alternative comparisons were made to observe other trends in changes of metadata. An additional Python script was written to observe the objective changes in metadata of all metadata reports and comparison records. The resulting output of this script showed that the most common and considerable changes in metadata of all files (RLO to deepfake, original to uploaded) were changes in the file name ("FileName"), file size ("MediaDataSize"), and average bitrate ("AvgBitrate"). These changes are illustrated in the appendix (see Fig. 2). This trend is consistent with the noticeable differences in the files downloaded from social media platforms as most downloaded content had a different name (e.g., Facebook is known to encrypt user content for privacy standards), and previous studies have shown that some metadata will be reduced or removed to enhance efficiency of uploading and storage of the files. One final script which was written revealed the relative changes of uploading to social media platforms to account for discrepancies in the uploaded or downloaded file count (see Fig. 4). This graph helped improve the neutrality of the data gathered, thereby revealing social media sites which had the most substantial impact on the metadata of files. The sites which had such were Truth Social, Facebook, Instagram, and then Tumblr and Twitter tied in that order. This trend discovered is consistent with the review made by [41] about Truth Social how the site does encrypt the data uploaded to the site when at rest. It is expected for most social media platforms to encrypt and remove most metadata uploaded to the site to improve the efficiency of the upload [42].

D. *Deepfake Detection Models*

D.1. *Video Deepfake Detection*. The video deepfake detection process called for the use of the Xception model. D.1.*a*. *Introduction to Xception*. This model is based on depth-wise separable convolutions as opposed to the usual standard convolutional layers used in convolutional neural networks (CNNs) [46]. To demonstrate the computational efficiency of the Xception model, consider the following:

1. Normal Convolution in CNNs
In a standard convolutional layer [53]:

- A kernel (filter) of size $k \times k$ operates on all input channels simultaneously to produce each output channel.
- Mathematically:

$$O(i,j,m) = \sum_{n=1}^{C} \sum_{p=1}^{k} \sum_{q=1}^{k} K(p,q,n,m) \cdot I(i+p, j+q, n)$$

  - $O(i,j,m)$: Output value at position (i,j) in the m-th output channel.
  - $K(p,q,n,m)$: Kernel weight for input channel n and output channel m.
  - $I(i+p, j+q, n)$: Input value at position $(i+p, j+q)$ in the n-th input channel.
  - C: Number of output channels.

This requires $(k \cdot k \cdot C \cdot M)$ multiplications, where M is the number of output channels.

2. Depthwise Separable Convolution

Depthwise separable convolution breaks this process into two smaller operations:

1. Depthwise Convolution:
   - Applies a single filter per input channel independently.
   - Mathematically:

$$O_d(i,j,n) = \sum_{p=1}^{k} \sum_{q=1}^{k} K_d(p,q,n) \cdot I(\, i+p, j+q, n)$$

     - $K_d(p,q,n)$: Depthwise kernel for the n-th channel.
- Reduces the number of multiplications to $(k \cdot k \cdot C)$.

2. Pointwise Convolution:
   - Applies a 1×1 kernel across all channels to combine Depthwise outputs:

$$O_p(i,j,m) = \sum_{n=1}^{C} K_p(n,m) \cdot O_d(i,j,n)$$

     - $K_p(n,m)$: Pointwise kernel weight for combining input channel n to output channel m.
   - Requires $(C \cdot M)$ multiplications.

Total Computation in Depthwise Separable Convolution:

$$\text{Total Operations} = (k \cdot k \cdot C) + (C \cdot M)$$

Compared to normal convolution, this is significantly smaller, especially for large k, C, and M.

3. Application to Xception

The Xception model replaces standard convolutions with depthwise separable convolutions in its convolutional blocks [46]. This approach:

1. Reduces the number of parameters.
2. Increases efficiency by focusing on spatial filtering (depthwise convolution) and channel mixing (pointwise convolution) as separate tasks.

Aspect	Normal Convolution	Depthwise Separable Convolution
Multiplications	$k \cdot k \cdot C \cdot M$	$k \cdot k \cdot C + C \cdot M$
Parameters	$k \cdot k \cdot C \cdot M$	$k \cdot k \cdot C + C \cdot M$
Efficiency	High Computational Cost	More efficient
Usage in Xception	Not used	Used in all major blocks

Table 1. Comparison: Normal Convolution vs. Depthwise Separable Convolution.

How Xception Processes Data:

1. Input Layer: Processes input data into a standard size (e.g., 299x299x3 for images).
2. Depthwise Separable Convolutions: Apply Depthwise filtering and channel mixing separately.
3. Global Average Pooling: Aggregates spatial features into a single vector per channel:

$$GAP(c) = \frac{1}{H \cdot W} \sum_{i=1}^{H} \sum_{j=1}^{W} O(i,j,c)$$

4. Fully Connected Layers: Use dense layers for classification:

$$y = \text{softmax}(Wx + b)$$

Advantages of Depthwise Separable Convolutions

1. Efficiency: Reduces the computational load significantly.
2. Flexibility: Allows better use of model parameters for larger networks.
3. Scalability: Well-suited for large-scale datasets and high-resolution images.

By leveraging depthwise separable convolutions, Xception achieves better performance with fewer parameters compared to traditional CNNs. This makes it particularly effective for tasks like deepfake detection, where computational efficiency and high accuracy are crucial. This has shown that it thus requires less computational resources and parameters. Studies have shown that the model outperforms the MobileNet V3, VGG-16, ResNet-50, ResNet-152, and Inception V3 models in terms of accuracy of deepfake detection and required fewer gradient descent steps and thus fewer parameters to approach negligible loss than its parent, Inception V3 [44][47][49]. This model can recognize subtle artifacts in images so well that it has shown to have applications in recognizing diseases in leaves and peaches, and quickly detect signs of COVID-19 from Xray images [47][48]. D.1.*b*. *Evaluation on Control Dataset*. The Xception model was trained and evaluated on the control dataset, achieving 89.1% accuracy. The architecture of the implementation for this model is illustrated later (see Fig. 16). However, detection performance dropped significantly for social media content, suggesting platform-induced changes in file attributes. A Python script was written to train and evaluate the Xception model on the Control dataset. The evaluation metrics showed that this model was good at recognizing deepfakes with a score of 89.1%. This value is to be expected when some of the content in the deepfake video dataset contained frames of real people due to both the Haotian AI samples and the quality of the deepfakes generated. The confusion matrix showing the overall distribution of this evaluation of the model is illustrated in Figure 5 (see Fig. 5). D.1.c. *Evaluation on Social Media Datasets*. An extension of the application of this deepfake detection model was considered and applied to evaluate the performance on detecting videos from the dataset of social media content. A 25 percent sample size was used for each dataset, including the control dataset, due to time constraints. The results of this experiment showed the model was unsuccessful at recognizing the selected deepfake video content in each selection, nearly completely misclassifying 100 percent of the deepfake video sample of each dataset. This suggests that the model either predicted the videos collected from the Haotian AI to be authentic (not deepfake) or changes made in the uploading process of the videos to social media sites. However, most confusion matrices in the appendix show that the latter is not possible due to a near successful RLO video prediction rate of the content uploaded to social media. The results of each evaluation of the Xception deepfake detection model on the 25% sample size of

each dataset of each social media site collected can be seen in the appendix of this paper. Figures 10, 12, and 15 show the most noticeable results of misclassification due to the dataset used was predominantly Haotian AI videos, as originally generated deepfake videos were too big of file size to upload to such social media platforms. This difference can be noticed by observing Figures 11, and 14 as the most normal results, with some misclassification expected due to discrepancies in metadata after uploading to social media. Figure 9 shows the control evaluation of the Xception model.

D.2. *Audio Deepfake Detection*. The audio deepfake detection process executed the use of the Light Convolutional Neural Network (LCNN) model. D.2.*a*. *Introduction to the LCNN Model*. The audio deepfake detection process utilized the Light Convolutional Neural Network (LCNN) model, which has demonstrated a strong ability to extract robust features provided by Mel-Spectrograms and MFCCs, aiding in effective deepfake detection. D.2.a.i. *Relevance of Feature Extraction*. By mapping the audio signal onto a perceptually relevant frequency scale, Mel-Spectrograms enable the LCNN to identify subtle frequency-domain artifacts introduced during deepfake synthesis. MFCCs compactly represent the spectral envelope, allowing the LCNN to focus on time-domain and frequency-domain irregularities. Unlike standard models that employ the Rectified Linear Unit (ReLU) activation function, the LCNN processes the extracted features using a series of 1D convolutional layers and employs a Max-Feature-Map (MFM) activation function.

$$MFM(x) = \max(w_1 \cdot x + b_1, w_2 \cdot x + b_2)$$

This activation function selects the most relevant features, suppressing noise and irrelevant activations, making the LCNN highly effective for detecting audio manipulations [50][51][63]. Research has highlighted the LCNN model's efficiency in recognizing both time-domain and frequency-domain artifacts commonly found in synthetic audio. Beyond deepfake detection, this model has also found applications in tasks such as speaker verification and detecting anomalies in live audio streams. In terms of computational efficiency, the LCNN model is highly optimized, achieving notable accuracy with minimal training time. For instance, it completed training on the control dataset in under five minutes, showcasing its practical application potential for real-time analysis scenarios [50]. D.2.*b*. *Evaluation on Control Dataset.* The evaluation metrics for the LCNN on the control dataset indicated a test accuracy of 100%, underscoring its reliability in detecting audio deepfakes. The architecture of the model implementation is illustrated later (see Fig. 17). This level of performance is attributed to its robust architecture and the quality of the curated dataset used during training. The confusion matrix summarizing the results of the model's evaluation on the control dataset is provided in Figure 6 (see Fig. 6). D.2.*c*. *Evaluation on Social Media Platforms*. Additionally, an extension of this model was applied to assess its performance on a broader dataset comprising various audio samples, including social media content. The results of this experiment illustrated the model's adaptability to diverse audio sources, further solidifying its efficacy in deepfake detection tasks [50][52]. The results of this extension show the LCNN model was just as capable of detecting deepfake content from the content uploaded to SoundCloud and Telegram as it was with the control (see Fig. 7) (see Fig. 8).

III. EXPERIMENTS AND RESULTS

A. *Experimental Setup*

The experiments involved generating deepfake content, uploading and downloading files from social media platforms, and analyzing metadata changes. Detection models were trained on the control dataset and tested on social media datasets.

B. *Model Training and Evaluation*

B.1. *Xception Model for Video Detection*.

- *Control Dataset Results*: Achieved 89.1% accuracy, with a confusion matrix showing a near-perfect classification of real-life and deepfake videos. Further evaluation metrics of original test of Xception model on Control dataset are illustrated in Figure 19 (see Appendix).
- *Social Media Dataset Results*: Nearly all deepfake videos were misclassified, likely due to metadata alterations.

B.2. *LCNN Model for Audio Detection*.

- *Control Dataset Results*: Achieved 100% accuracy, as shown in Figure 6. Further information on evaluation metrics of LCNN

model test on Control dataset are illustrated in Figure 18 (see Appendix).

- *Social Media Dataset Results*: Maintained high accuracy for platforms like Telegram and SoundCloud, as shown in Figures 7 and 8.

C. *Metadata Analysis Results*

Python scripts revealed consistent metadata changes in file name, file size, and average bitrate, with deepfake files showing more significant alterations. Social media platforms like Facebook, Instagram, and Truth Social exhibited the most substantial metadata modifications.

IV. DISCUSSION

A. *Impact of Haotian AI Samples on Xception Model Performance*

The inclusion of samples generated by Haotian AI's deepfake tools revealed significant challenges in detection accuracy. The Xception model, which performed effectively on the control dataset with an accuracy of 89.1%, demonstrated a marked inability to recognize deepfake content from Haotian AI's tools. Specifically, nearly 100% of the Haotian AI-generated samples were misclassified as authentic. This suggests that the advanced algorithms employed by Haotian AI produce artifacts and features that are either indistinguishable by the Xception model or sufficiently realistic to bypass its detection capabilities. One possible explanation for this discrepancy is that the dataset of deepfake videos used to train the Xception model may have been primarily generated using the Deepfake Offensive Toolkit. While this toolkit is known for its efficiency and ability to produce convincing deepfakes, it does not necessarily represent the full spectrum of artifacts and features introduced by other advanced tools like Haotian AI. The reliance on a single generation tool for training could lead to overfitting to its specific patterns and artifacts, reducing the model's ability to generalize to other tools. The Deepfake Offensive Toolkit is notable for its ability to generate high-quality deepfakes efficiently. It uses pre-trained models optimized for real-time deepfake generation, allowing for faster production compared to tools like Faceswap or DeepFaceLab. However, this efficiency can come at the cost of diversity in the types of artifacts and distortions produced. This limitation may have contributed to the Xception model's inability to detect Haotian AI's deepfakes, as the training data did not sufficiently encompass the nuanced features of Haotian AI's outputs.

B. *Metadata Changes and Their Limited Impact on Detection*

The results also demonstrated that changes in metadata had minimal impact on the Xception model's ability to recognize deepfakes. While uploading content to social media platforms caused significant alterations in metadata, such as changes in file name, size, and average bitrate, these did not appear to affect the detection process. The model's reliance on visual artifacts rather than metadata attributes accounts for this finding. This observation aligns with the critique of online suggestions advocating for metadata analysis as a reliable deepfake detection method. As discussed in the introduction, metadata can be easily removed, encrypted, or altered by social media platforms, rendering it an unreliable standalone approach. The project's results reinforce the idea that metadata-based detection is infeasible for robust deepfake identification, particularly when dealing with advanced generation tools or altered files.

C. *Future Implications*

The findings suggest a need for advanced detection techniques that go beyond traditional artifact analysis. Multimodal approaches, which integrate visual, audio, and metadata cues, could enhance the ability to detect sophisticated deepfakes. Furthermore, training datasets must incorporate a wider range of tools, including Haotian AI and other emerging technologies, to ensure the models can generalize effectively. Expanding the dataset to include variations in generation quality and algorithms would mitigate the limitations of relying on any single deepfake tool. Moreover, further research is required to evaluate the role of anti-deepfake strategies implemented by social media platforms, as these may introduce subtle changes that impact detection models. Exploring alternative detection frameworks, such as transformer-based or multimodal methods, could address the evolving landscape of deepfake technologies.

V. CONCLUSION

This study evaluated the Xception, and LCNN model for detecting deepfakes in video and audio datasets. While the LCNN model demonstrated consistent accuracy across control and social media datasets, the

Xception model struggled with detecting high-quality deepfakes generated by Haotian AI. The significant misclassification of Haotian AI samples underscores the limitations of current models in addressing advanced algorithms. Additionally, the possible reliance on deepfake samples generated by the Deepfake Offensive Toolkit during model training may have contributed to the model's reduced generalization. While the toolkit produces efficient and convincing deepfakes, its artifacts may not fully capture the nuances of other tools like Haotian AI. This highlights the importance of diverse datasets in model training. Furthermore, the study revealed that metadata changes resulting from social media uploads had minimal impact on detection accuracy, challenging the effectiveness of metadata-based detection strategies. These findings emphasize the importance of training detection models with diverse datasets and adopting multimodal approaches to address the growing sophistication of deepfake technologies. Future research should prioritize the inclusion of samples from emerging deepfake tools and explore innovative detection frameworks to ensure the reliability and applicability of detection methods in real-world scenarios.

REFERENCES

[1] "27 Arrested In $360 Million DeepFake Scam Syndicate," *Frank on Fraud*, Oct. 14, 2024. https://frankonfraud.com/fraud-trends/27-arrested-in-46-million-deepfake-scam-syndicate/ (accessed Nov. 29, 2024).

[2] "Haotian AI : Providing Deepfake AI For Scam Bosses," *Frank on Fraud*, Oct. 10, 2024. https://frankonfraud.com/fraud-trends/haotian-ai-providing-deepfake-ai-for-scam-bosses/ (accessed Nov. 29, 2024).

[3] Anis Trabelsi, M. M. Pic, and Jean-Luc Dugelay, "Improving Deepfake Detection by Mixing Top Solutions of the DFDC," *2021 29th European Signal Processing Conference (EUSIPCO)*, vol. abs 1710 10196, pp. 643–647, Aug. 2022, doi: https://doi.org/10.23919/eusipco55093.2022.9909905.

[4] A. Kaushal, S. Singh, S. Negi, and S. Chhaukar, "A Comparative Study on Deepfake Detection Algorithms," *IEEE Xplore*, Dec. 01, 2022. https://ieeexplore.ieee.org/document/10074593

[5] Sanika Tiwarekar, "Deep Fake Detection (DFD) Entire Original Dataset," *Kaggle.com*, 2024. https://www.kaggle.com/datasets/sanikatiwarekar/deep-fake-detection-dfd-entire-original-dataset (accessed Nov. 29, 2024).

[6] Haotian AI, "昊天AI换脸 官方频道-AI换脸-实时换脸," *Telegram*, 2024. https://t.me/haotianai (accessed Nov. 29, 2024).

[7] S. Kumar, "1000 male voice samples," *Kaggle.com*, 2022. https://www.kaggle.com/datasets/sandeep16064/1000-male-voice-samples (accessed Nov. 29, 2024).

[8] S. Kumar, "1000 female voice samples," *Kaggle.com*, 2022. https://www.kaggle.com/datasets/sandeep16064/2000-voice-samples (accessed Nov. 29, 2024).

[9] iperov, "iperov/DeepFaceLab," *GitHub*, Apr. 08, 2020. https://github.com/iperov/DeepFaceLab

[10] waywardspooky, "Reddit - Dive into anything," *Reddit.com*, 2024. https://www.reddit.com/r/DeepFaceLab/comments/1eyvlwo/deepfacelab_shutdown_on_github/ (accessed Nov. 29, 2024).

[11] G. Chaabouni, A. Indulkar, G. Patrini, and V. Papadopoulos, "the Deepfake Offensive Toolkit," *GitHub*, Dec. 11, 2022. https://github.com/sensity-ai/dot

[12] ondyari, "FaceForensics++: Learning to Detect Manipulated Facial Images," *GitHub*, Dec. 02, 2022. https://github.com/ondyari/FaceForensics

[13] C. Jemine, "CorentinJ/Real-Time-Voice-Cloning," *GitHub*, Jun. 05, 2020. https://github.com/CorentinJ/Real-Time-Voice-Cloning

[14] "AI Voice Generator & Realistic Text to Speech Online," *Play.ht*, 2016. https://play.ht

[15] "Free Real Time Voice Changer & Modulator - Voicemod," *www.voicemod.net*. https://www.voicemod.net

[16] OpenAI, "Whisper," *GitHub*, Oct. 09, 2022. https://github.com/openai/whisper

[17] Deepfakery, "Easy Deepfake Tutorial: DeepFaceLab 2.0 Quick96," *YouTube*, Jul. 27, 2020. https://www.youtube.com/watch?v=lSM-9RBk3HQ (accessed Nov. 29, 2024).

[18] N. Dufour and A. Gully, "Contributing Data to Deepfake Detection Research," *research.google*, Sep. 24, 2019. https://research.google/blog/contributing-data-to-deepfake-detection-research/

[19] "Download DIRECTX 12 to add amazing graphics effects | NVIDIA," *www.nvidia.com*. https://www.nvidia.com/en-us/geforce/technologies/dx12/

[20] Michelle M. Graham, "Deepfakes: Federal and state regulation aims to curb a growing threat," *Thomson Reuters Institute*, Jun. 26, 2024. https://www.thomsonreuters.com/en-us/posts/government/deepfakes-federal-state-regulation/

[21] GeeksforGeeks, "Types of Convolution Kernels," *GeeksforGeeks*, Jul. 22, 2024. https://www.geeksforgeeks.org/types-of-convolution-kernels/

[22] GeeksforGeeks, "Introduction to Convolution Neural Network," *GeeksforGeeks*, Mar. 14, 2024. https://www.geeksforgeeks.org/introduction-convolution-neural-network/

[23] A. Zucconi, "Understanding the Technology Behind DeepFakes - Alan Zucconi," *Alan Zucconi*, Mar. 14, 2018. https://www.alanzucconi.com/2018/03/14/understanding-the-technology-behind-deepfakes/

[24] Q. Luo and K. Vinayagam Sivasundari, "Whisper+AASIST for DeepFake Audio Detection," *Lecture Notes in Computer Science*, pp. 121–133, 2024, doi: https://doi.org/10.1007/978-3-031-61382-1_8.

[25] T.-P. Doan, L. Nguyen-Vu, S. Jung, and K. Hong, "BTS-E: Audio Deepfake Detection Using Breathing-Talking-Silence Encoder," *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Jun. 2023, doi: https://doi.org/10.1109/icassp49357.2023.10095927.

[26] Q. Luo and K. Sivasundari, "Whisper+AASIST for DeepFake Audio Detection."

[27] GeeksforGeeks, "CNN | Introduction to Pooling Layer," *GeeksforGeeks*, Aug. 05, 2019. https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/

[28] "What are DeepFakes and How Dangerous are They?," *GeeksforGeeks*, Mar. 12, 2020. https://www.geeksforgeeks.org/what-are-deepfakes-and-how-dangerous-are-they/

[29] "Welcome," *Faceswap*. https://faceswap.dev

[30] H. Lee *et al.*, "The Tug-of-War Between Deepfake Generation and Detection," *Arxiv.org*, 2020. https://arxiv.org/html/2407.06174v4?utm_source=chatgpt.com (accessed Nov. 30, 2024).

[31] A. Kaur, Azadeh Noori Hoshyar, Vidya Saikrishna, S. Firmin, and F. Xia, "Deepfake video detection: challenges and opportunities," *Artificial intelligence review*, vol. 57, no. 6, May 2024, doi: https://doi.org/10.1007/s10462-024-10810-6.

[32] G. Gupta, K. Raja, M. Gupta, T. Jan, Scott Thompson Whiteside, and M. Prasad, "A Comprehensive Review of DeepFake Detection Using Advanced Machine Learning and Fusion Methods," *Electronics*, vol. 13, no. 1, pp. 95–95, Dec. 2023, doi: https://doi.org/10.3390/electronics13010095.

[33] A. Naitali, M. Ridouani, F. Salahdine, and N. Kaabouch, "Deepfake Attacks: Generation, Detection, Datasets, Challenges, and Research Directions," *Computers*, vol. 12, no. 10, p. 216, Oct. 2023, doi: https://doi.org/10.3390/computers12100216.

[34] E.-G. Lee, I. Lee, and S.-B. Yoo, "ClueCatcher: Catching Domain-Wise Independent Clues for Deepfake Detection," *Mathematics*, vol. 11, no. 18, p. 3952, Jan. 2023, doi: https://doi.org/10.3390/math11183952.

[35] Y. Li and S. Lyu, "Exposing DeepFake Videos By Detecting Face Warping Artifacts," *arXiv:1811.00656 [cs]*, May 2019, Available: https://arxiv.org/abs/1811.00656

[36] "The Deepfake Offensive Toolkit," *Changelog*, 2022. https://changelog.com/news/the-deepfake-offensive-toolkit-zD4V (accessed Dec. 01, 2024).

[37] K. O'Neal-Kenny, "Can Metadata Help Identify Deepfakes? — INM348 Article 2 Extras," *www.linkedin.com*, Nov. 11, 2021. https://www.linkedin.com/pulse/can-metadata-help-identify-deepfakes-inm348-article-2-kaitlyn-o-neal/

[38] M. Edirisooriya, "Defeating Deepfakes with Metadata," *Medium*, Aug. 11, 2023. https://medium.com/@maninda/defeating-deepfakes-with-metadata-2e1de42d24c6

[39] O. Laurent, "Study exposes social media sites that delete photographs' metadata - 1854 Photography," *www.1854.photography*. https://www.1854.photography/2013/03/study-exposes-social-media-sites-that-delete-photographs-metadata/

[40] "Social Media Metadata on Mobile Devices: Gathering Valuable Crumbs," *Digital Mountain*. https://digitalmountain.com/newsletter/social-media-metadata-on-mobile-devices-gathering-valuable-crumbs/

[41] "Common Sense Privacy Standard Privacy Report for Truth Social," *Commonsense.org*, 2023. https://privacy.commonsense.org/privacy-report/Truth-Social (accessed Dec. 02, 2024).

[42] Scott, "Scott Wyden Kivowitz," *Scott Wyden Kivowitz*, Jun. 03, 2010. https://scottwyden.com/facebooks-photo-upload-compression/ (accessed Dec. 02, 2024).

[43] "How to Spot Deepfake Scams," *Ncoa.org*, Oct. 30, 2024. https://www.ncoa.org/article/understanding-deepfakes-what-older-adults-need-to-know/

[44] Ashok V and Preetha Theresa Joy, "Deepfake Detection Using XceptionNet," *2023 IEEE International Conference on Recent Advances in Systems Science and Engineering (RASSE)*, Nov. 2023, doi: https://doi.org/10.1109/rasse60029.2023.10363477.

[45] D. Ahmed Alkurdi, M. Cevik, and A. Akgundogdu, “Advancing Deepfake Detection Using Xception Architecture: A Robust Approach for Safeguarding against Fabricated News on Social Media,” *Techscience.com*, 2024. http://www.techscience.com/cmc/online/detail/22022 (accessed Dec. 05, 2024).

[46] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” *arXiv.org*, 2016. https://arxiv.org/abs/1610.02357

[47] G. Boesch, “Xception Model: Analyzing Depthwise Separable Convolutions - viso.ai,” *viso.ai*, May 16, 2024. https://viso.ai/deep-learning/xception-model/?utm_source=chatgpt.com (accessed Dec. 05, 2024).

[48] A. H. Panahi, A. Rafiei, and A. Rezaee, “FCOD: Fast COVID-19 Detector based on deep learning techniques,” *Informatics in Medicine Unlocked*, vol. 22, p. 100506, 2021, doi: https://doi.org/10.1016/j.imu.2020.100506.

[49] N. Yao *et al.*, “L2MXception: an improved Xception network for classification of peach diseases,” *Plant Methods*, vol. 17, no. 1, Apr. 2021, doi: https://doi.org/10.1186/s13007-021-00736-3.

[50] Z. Wu, R. K. Das, J. Yang, and H. Li, “Light Convolutional Neural Network with Feature Genuinization for Detection of Synthetic Speech Attacks,” *arXiv.org*, 2020. https://arxiv.org/abs/2009.09637 (accessed Dec. 06, 2024).

[51] Z. Yang, M. Jian, B. Bao, and L. Wu, “Max-Feature-Map Based Light Convolutional Embedding Networks for Face Verification,” *Lecture notes in computer science*, pp. 58–65, Jan. 2017, doi: https://doi.org/10.1007/978-3-319-69923-3_7.

[52] S. Joshi and M. Dua, “Noise Robust Audio Spoof Detection Using Hybrid Feature Extraction and LCNN,” *SN Computer Science*, vol. 5, no. 4, Apr. 2024, doi: https://doi.org/10.1007/s42979-024-02774-9.

[53] I. Goodfellow, Y. Bengio, and A. Courville, “Deep Learning,” *www.deeplearningbook.org*, 2016. https://www.deeplearningbook.org

[54] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: https://doi.org/10.1162/neco.1997.9.8.1735.

[55] A. Vaswani *et al.*, “Attention Is All You Need,” *arXiv.org*, Dec. 05, 2017. https://arxiv.org/abs/1706.03762

[56] OpenAI, “Whisper,” *GitHub*, Oct. 09, 2022. https://github.com/openai/whisper

[57] “Documentation,” *ffmpeg.org*. https://ffmpeg.org/documentation.html

[58] E. Tatulli and T. Hueber, “Feature extraction using multimodal convolutional neural networks for visual speech recognition,” *IEEE Xplore*, Mar. 01, 2017. https://ieeexplore.ieee.org/abstract/document/7952701 (accessed Jul. 09, 2020).

[59] S. S. Stevens, “A Scale for the Measurement of the Psychological Magnitude Pitch,” *Acoustical Society of America Journal*, vol. 8, no. 3, p. 185, 1937, doi: https://doi.org/10.1121/1.1915893.

[60] B. Mcfee *et al.*, “librosa: Audio and Music Signal Analysis in Python,” *PROC. OF THE 14th PYTHON IN SCIENCE CONF. (SCIPY*, vol. 2015, p. 1.

[61] S. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4, pp. 357–366, Aug. 1980, doi: https://doi.org/10.1109/tassp.1980.1163420.

[62] I. J. Goodfellow *et al.*, “Generative Adversarial Networks,” *arXiv (Cornell University)*, vol. 1, Jun. 2014, doi: https://doi.org/10.48550/arxiv.1406.2661.

[63] X. Wu, R. He, Z. Sun, and T. Tan, “A Light CNN for Deep Face Representation With Noisy Labels,” *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2884–2896, Nov. 2018, doi: https://doi.org/10.1109/tifs.2018.2833032.

APPENDIX

The following are materials which are referenced in this paper to aid either the discussion of the data distribution on social media and metadata analysis, or detection results.

Fig. 1 Table showing what content was successful to upload to which social media sites, including information relating to content type, size, amount, and capabilities of batch upload and download to and from which social media sites.

	Video Files	*Sound Files*	*Real Life Videos*	*Real Life Audio*	*Deepfake Video*	*Deepfake Audio*	*Batch upload*	*Batch download*
DeviantArt	Yes	Yes	No	No	Yes	No	No	No
Discord	Yes	No	Yes	Yes	No (Too big file size)	Yes	Only up to 10	No
Facebook	Yes	No	Yes	No	Only to a certain file size	No	10+	Yes
Instagram	Yes	No	Yes	No	Only to a certain file size	No	10+	Yes, but not comprehensive
Pinterest	No	No	N/A	N/A	N/A	N/A	N/A	N/A
Reddit	Yes	No	Yes	No	Only to a certain file size	No	No	No
Snapchat	Yes	No	Yes	No	No (Not feasible)	No	No	No
SoundCloud	No	Yes	No	Yes	No	Yes	Yes	No
Telegram	Yes	Yes	Yes	Yes	Yes	Yes	Yes, as long as the internet could handle the upload demand	No
TikTok	Yes	No	Yes	No	Only to a certain file size	No	Yes, but this would clump the files into one file	No
Truth-Social	Yes	No	Yes	No	Only to a certain file size	No	No (implicitly flagged for spam)	No
Tumblr	Yes	No (Failed)	Yes	N/A	Only to a certain file size	N/A	No (implicitly flagged for spam)	No
Twitter	Yes	No	Yes	No	Only to a certain file size	No	Only to four at a time, but could make a thread	No
Vimeo	Yes	No	Yes	No	Only to a certain file size	No	No	No
WhatsApp	Yes	Yes	Yes	Yes	Only to a certain file size	Yes	No	No
YouTube	Yes	No	Yes	No	Only to a certain file size	No	Only up to 10	No
Threads	Yes	No	Yes	No	Only to a certain file size	No	No (was flagged for spam)	N/A
Steam	No	No	N/A	N/A	N/A	N/A	N/A	N/A

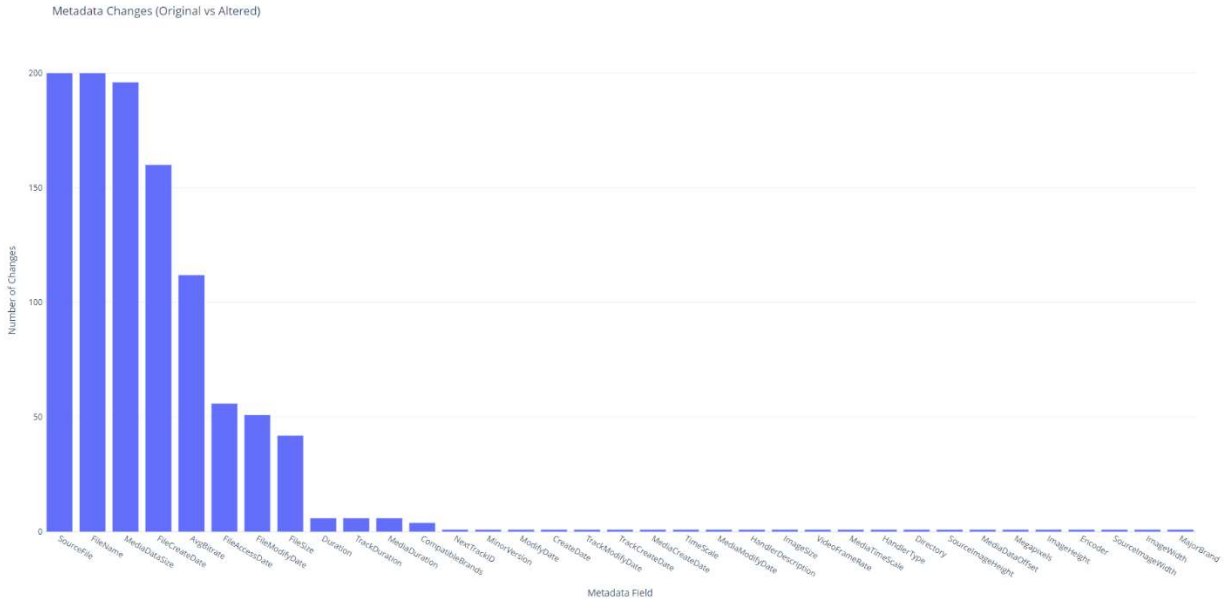

Fig. 2 Graph showing organization of metadata changes, showing attributes with number of changes from largest to smallest.

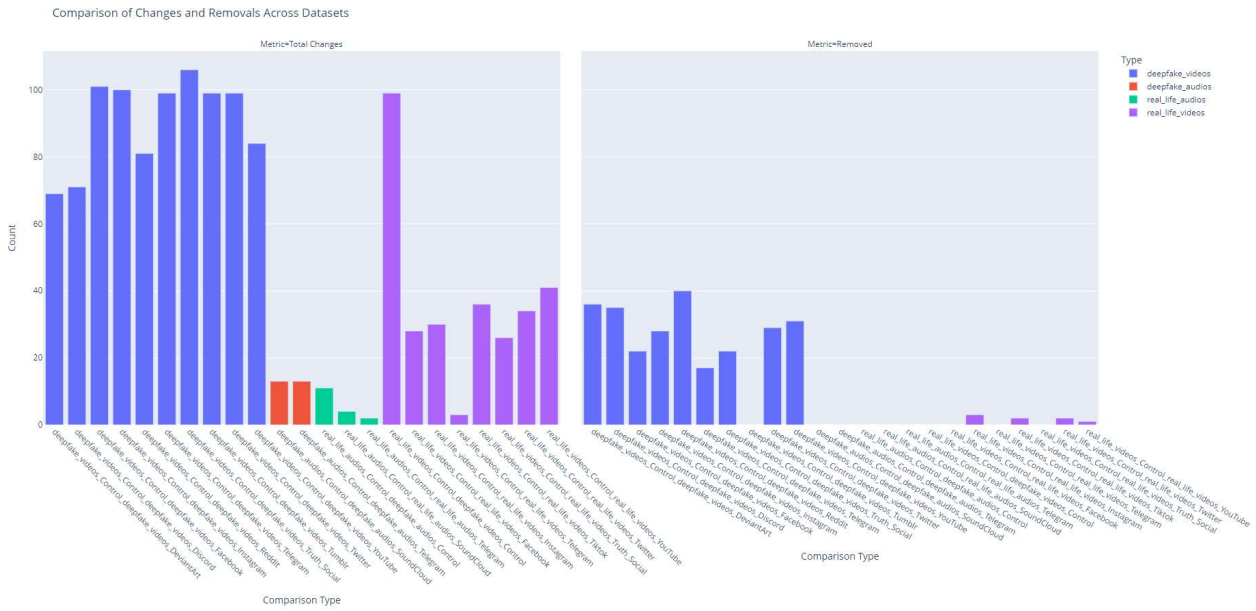

Fig. 3 Graph showing changes either addition or alteration of attributes (left) and removal of attributes (right).

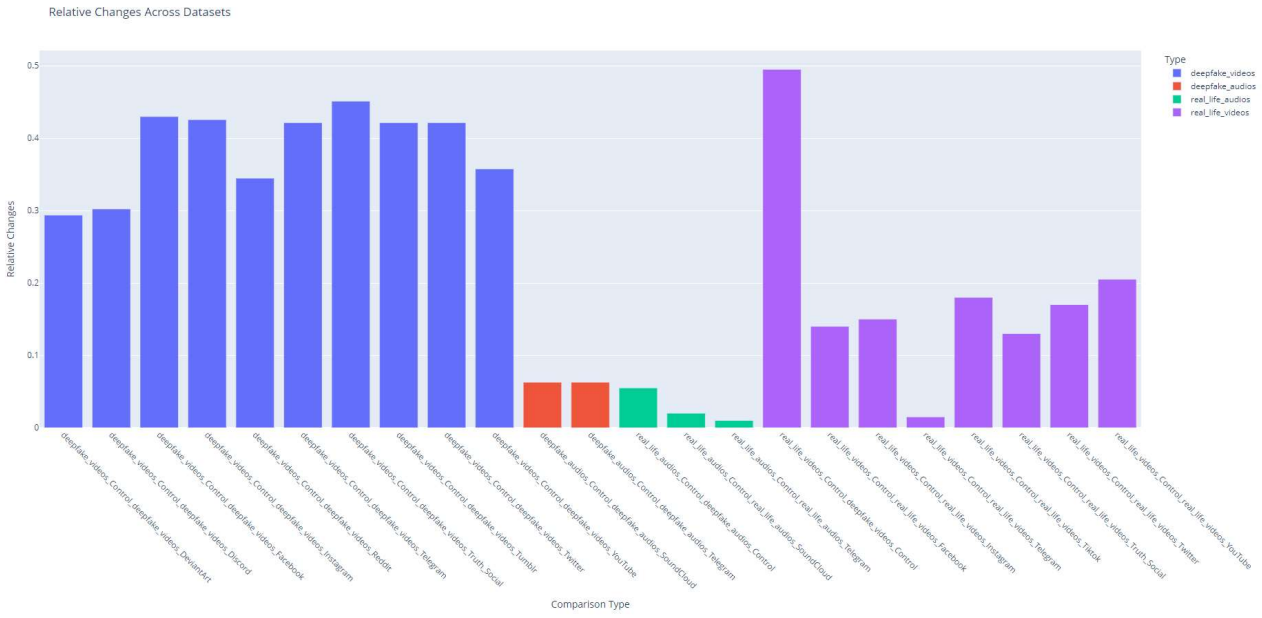

Fig. 4 Graph showing relative changes across datasets, with deepfake content uploaded to social media sites on the left, audio files in the middle (red and green), real-life content to deepfake shown as the purple bar leftmost with all other real-life content uploaded to social media to the right in purple.

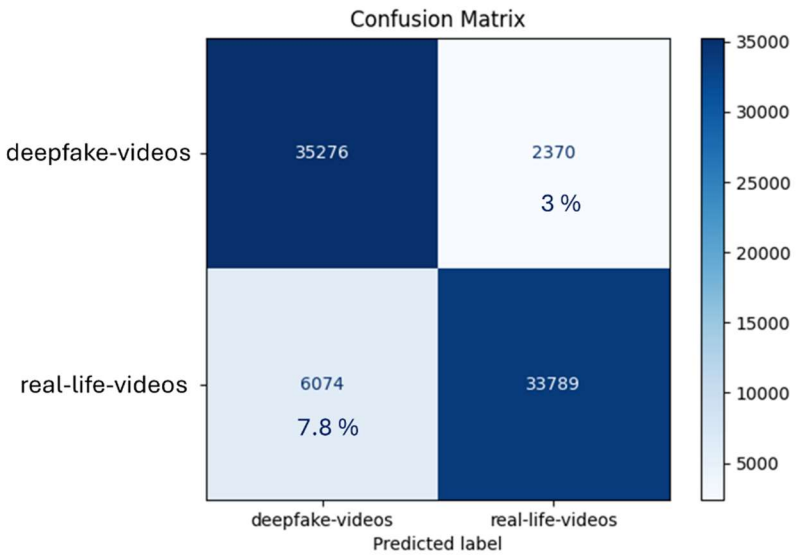

Fig. 5 Confusion Matrix shows a distribution of correct classifications and misclassifications of RLO and deepfake control videos.

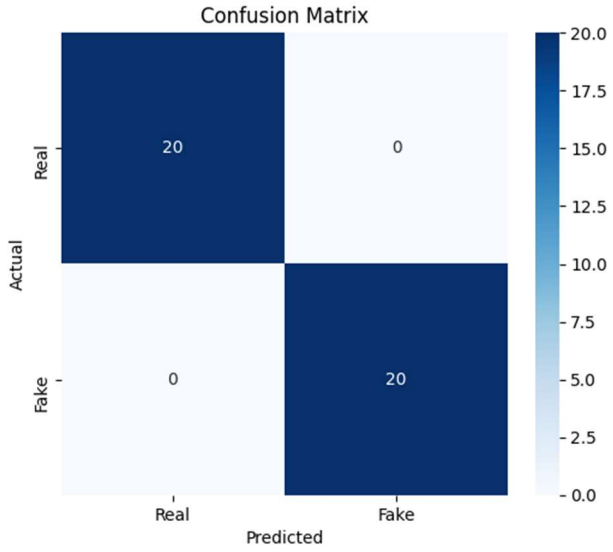

Fig. 6 Confusion Matrix shows a distribution of correct classifications of RLO and deepfake control audio.

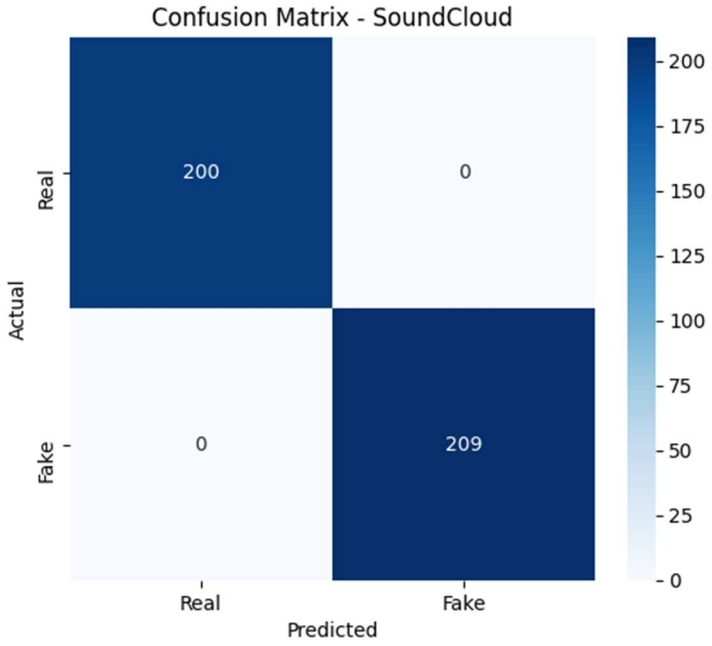

Fig. 7 Confusion Matrix shows a distribution of correct classifications of RLO and deepfake audio uploaded to SoundCloud.

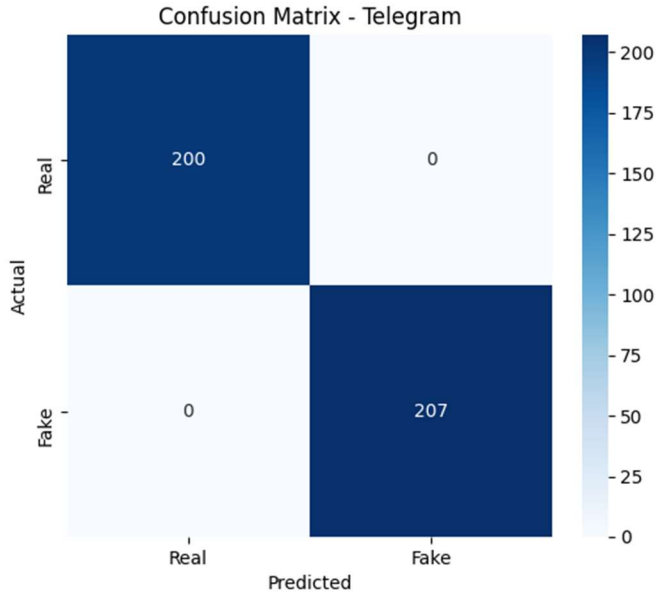

Fig. 8. Confusion Matrix shows a distribution of correct classifications of RLO and deepfake audio uploaded to Telegram.

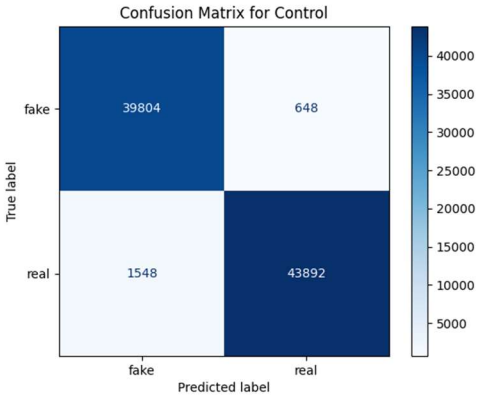

Fig. 9. Control test of Xception model on 50 randomly selected deepfake videos.

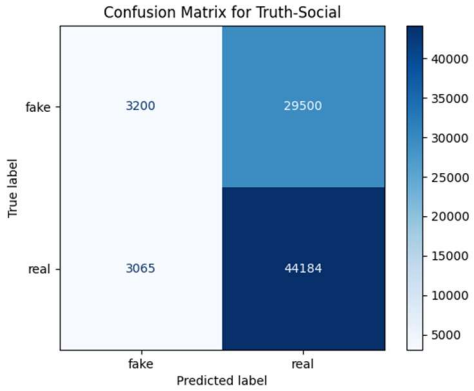

Fig. 10. Evaluation of Xception model on collection of deepfake videos uploaded to Truth-Social shows staggering misclassification of fake videos, majority Haotian AI samples.

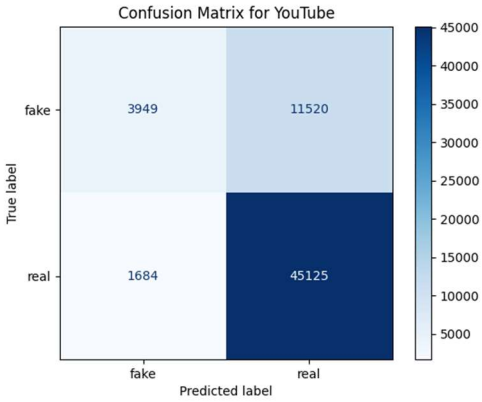

Fig. 11. Evaluation of Xception model on deepfake videos uploaded to YouTube with slight misclassification of deepfake videos.

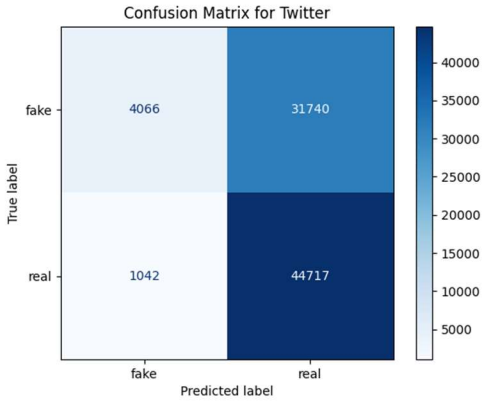

Fig. 12. Evaluation of Xception model on deepfake videos uploaded to Twitter with staggering misclassification of deepfake videos, majority Haotian AI samples.

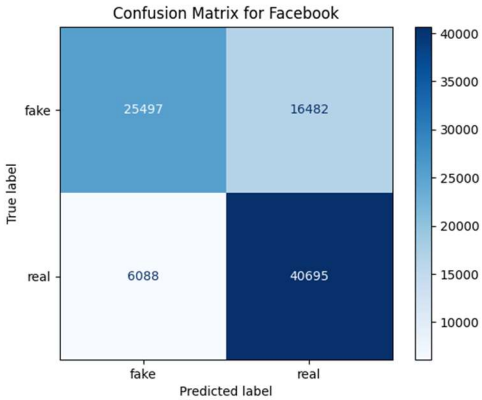

Fig. 13. Evaluation of Xception model on Facebook videos, with some successful classification of deepfake videos due to majority original deepfake content.

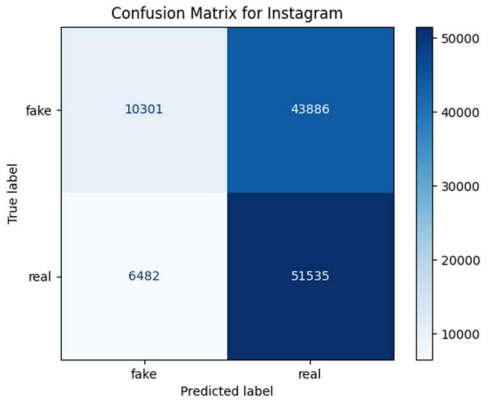

Fig. 15. Evaluation of Xception model on deepfake videos with staggering misclassification due to Haotian AI deepfake samples.

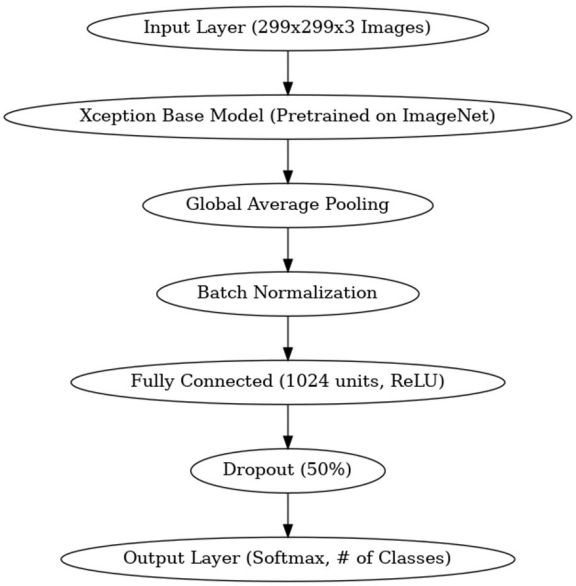

Fig. 16. Xception model architecture.

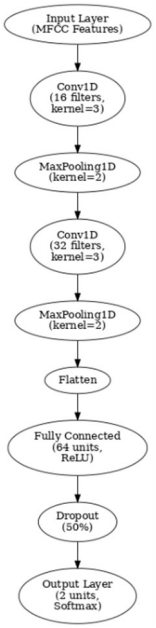

Fig. 17. Light Convolutional Neural Network (LCNN) model architecture.

```
Train Loss: 0.0026 Acc: 1.0000
Val Loss: 0.0009 Acc: 1.0000
Model saved to E:\cwf016\DFSC 6312 Research Project\deepfake-audios\Control\saved_models\audio_lcnn_model.pth
Test Accuracy: 1.0000
Test F1 Score: 1.0000
Test Precision: 1.0000
```

Fig. 18. Evaluation metrics of LCNN audio deepfake detection model

```
Classification Report:
                precision    recall  f1-score   support

 deepfake-videos       0.85      0.94      0.89     37646
real-life-videos       0.93      0.85      0.89     39863

        accuracy                           0.89     77509
       macro avg       0.89      0.89      0.89     77509
    weighted avg       0.89      0.89      0.89     77509
```

Fig. 19. Evaluation metrics of Xception video deepfake detection model.