

A Comprehensive Content Verification System for ensuring Digital Integrity in the age of Deep Fakes

Ravi Kanth Kaja
Microsoft
ravi.kaja@microsoft.com

Abstract – In an era marked by the widespread sharing of digital content, the need for a robust content-integrity verification goes beyond the confines of individual social media platforms. While verified profiles (such as blue ticks on platforms like Instagram and X) have become synonymous with credibility, the content they share often traverses a complex network of interconnected platforms, by means of re-sharing, re-posting, etc., leaving a void in the authentication process of the *content* itself. With the advent of easily accessible AI tools (like DALL-E, Sora, and the tools that are explicitly built for generating deepfakes & face swaps), the risk of misinformation through social media platforms is growing exponentially. This white paper introduces a solution, a *Content Verification System*, designed to authenticate images and videos shared as posts or stories across the digital landscape. Going beyond the limitations of blue ticks, this system empowers individuals and influencers to validate the authenticity of their digital footprint, safeguarding their reputation in an interconnected world.

This solution serves as a vital defense against the increasing risk posed by deepfakes generated by advanced AI. Just as X seeks to combat the issue of bots on its platform, this Content Verification System is intended to counter the proliferation of morphed, forged, and deepfake content that circulates across the internet.

Keywords – Deep Fakes, Generative AI, Steganography, Watermarking, Cryptography, Content Integrity

1 PROBLEM STATEMENT

In 2022, the digital realm witnessed a surge in deepfake incidents, from a deceptive video of Ukrainian President Volodymyr Zelensky urging soldiers to lay down arms to a misleading clip of US President Joe Biden calling for a national draft. Even Russian President Vladimir Putin and Hollywood star Tom Cruise fell victim to manipulated videos, adding to the growing concern.

Fast forward to 2024, and the dark side of AI manifested in sexually explicit deepfake images of Taylor Swift circulating on social media, sparking heated discussions among privacy advocates. On the other side, in a first-of-its-kind incident, a multinational company's Hong Kong office suffered a staggering financial loss of HK\$200 million due to a sophisticated deepfake scam.

As artificial intelligence continues to advance rapidly, the realism of deepfakes is poised to increase, making it progressively challenging to distinguish them from authentic content. Deepfakes have the potential to manipulate public sentiments, tank stock prices, influence voters, or even create public unrest. A system to validate the authenticity and integrity of the online content is the need of the hour.

2 MAIN CONTENT

2.1 PROPOSED SOLUTION

Ongoing research is actively exploring the identification of deepfakes through Machine Learning, Generative Adversarial Networks (GAN), behavioral sciences, and by considering the existing constraints of generative AI. However, the continual advancements in AI are expected to heighten the difficulty of achieving success with these approaches.

The alternate solution proposed here is to develop a *content verification system* in which

- An individual or an entity registers their content on the content verification platform
- The recipients of the content can verify the authenticity and the integrity of the content


Content in this context refers to images, audio or videos that are intended to be shared on social media platforms or internet.

The proposed solution can be achieved with zero dependency on any of the social media platforms, by providing a *browser extension* interface for both registering the content as well as for showing the verification tag on the verified content (as shown below).



[Image by luis_molinero on Freepik]

Figure 1: An example social media post with content verification tag.

Refer to the  verification tag on the top-right corner of the image

2.1.1 TYPES OF CONTENT AND SCOPE

Images, Audio & Video are the 3 types of content which could benefit from a content verification system. In this paper, images have been taken under scope, but the solution proposed can be implemented for audio and video content as well.

In case of video content, scene change detection can be achieved by comparison of localized histograms in between scenes, and scene averages can be calculated. The same solution that is implemented for images can be extended by applying it on scene averages.

Note: For the rest of this paper, Content & Image will be used interchangeably

2.1.2 LIMITATIONS OF DIGITAL SIGNATURE AND PROPOSED ALTERNATIVE

The default solution of *digital signatures* for the content is not applicable in this scenario as:

1. **PROBLEM:** The metadata (like .exif) of the image is stripped as soon as it is uploaded onto social media making it impossible to maintain a *persistent identity* for the image while it is shared or exchanged across users and platforms.

SOLUTION: An identity is generated for the image when it is onboarded to content verification system and this information is embedded into the image using *steganographic techniques*. The identity of the image can be retrieved at any later point in time by extracting the embedded information.

Image steganography is a technique of hiding data or information within an image in such a way that it is concealed from plain view.

2. **PROBLEM:** When an image is shared across social media, it goes through multiple transformations like compression, change of brightness, format change, resize, cropping, etc. Hence, validating the *integrity* of the image by strict comparison of the contents of the image would lead to false positives (or true negatives) and is not an acceptable solution.

SOLUTION: Image authentication & integrity check implementation using *selective authentication*.

Selective authentication aims to provide robustness against any specific and desired manipulations while detecting any malevolent operations.

2.2 CONTENT VERIFICATION SCHEME

2.2.1 CONTENT REGISTRATION

1. Extract Features of the Image. *Features* refer to the characteristics or attributes that describe the image content
2. Save it in the database/register of Content Verification System
3. Create a Barcode Encoded with the address of the image in the database
4. Embed the Barcode as an invisible watermark into the Image
5. Return the Image for sharing across social media/internet

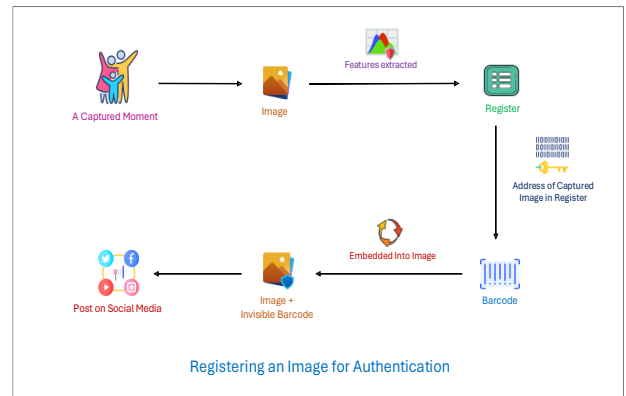


Figure 2: Content Registration Scheme explained

2.2.2 CONTENT VERIFICATION

1. Extract the barcode/watermark from the Image
2. Decode the barcode to get the address in the database for the Image
3. Fetch the original image's features from the database/Register
4. Extract the features from the image under verification
5. Compare the features and return a confidence score on the integrity of the image under verification

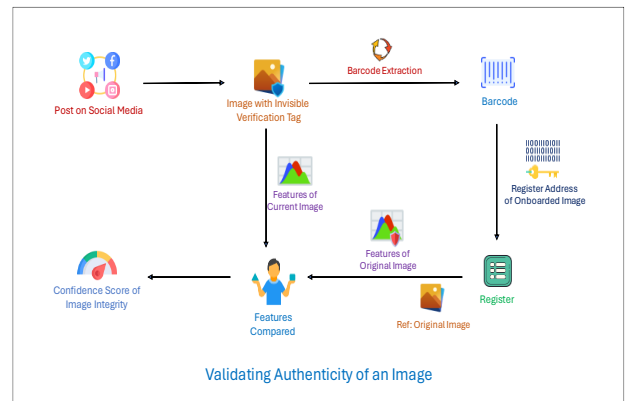


Figure 3: Content Verification Scheme explained

2.3 DESIGN CONSIDERATIONS

2.3.1 CONTENT REGISTRATION

2.3.1.1 FEATURE EXTRACTION

An approach leveraging Discrete Cosine Transform is employed for feature extraction and formulation of image signature. The Discrete Cosine Transform (DCT) is a mathematical technique used in image processing and signal compression, in which, an image is represented by a matrix of coefficients representing the amplitude distribution of cosine waves which constitute the final image. Various techniques have been evaluated based on the report provided by Adil Haouzia & Rita Noumeir in ‘Methods for image authentication: a survey’, and DCT has been chosen, as it stands out amongst the rest, in its robustness against image modifications like jpeg compressions, filtering, brightening etc. The algorithm employed involves breaking down an image into 8x8 pixel blocks, and performing DCT on each individual block to have a more localized feature definition for each individual block.

```
def perform_dct(img, blk_size):  
    image_size = img.shape  
    dct = np.zeros(image_size)  
  
    #Perform blockSize x blockSize DCT on image  
    for i in range(0, image_size[0], blk_size):  
        for j in range(0, image_size[1], blk_size):  
            #Adjust mid-grey to zero in 8-bit image  
            blk_adj = np.ones((blk_size, blk_size)) * 128  
            mid_adj = (img[i:(i+blk_size), j:(j+blk_size)] - blk_adj)  
            #Normalize the block [0,1]  
            normalized_blk = np.float32(mid_adj)  
            #DCT using OpenCV2 & Assigning to numpy dynamic array element  
            dct[i:(i+blk_size), j:(j+blk_size)] = cv2.dct(normalized_blk)  
  
    return dct
```

Figure 4: Python code for Feature Extraction through Discrete Cosine Transform (DCT)

2.3.1.2 IMAGE SCALING ACROSS SOCIAL PLATFORMS

As image manipulations are common for any content when it is exchanged across social media platforms, a feature definition of one dimension might not always match with the feature definition of a scaled-up/scaled-down image. Below is a reference to some of the standard dimensions & aspect ratios of different social platforms as of today. When an image is onboarded for verification, its dimensions are recalibrated to the closest match of recommended aspect ratios and the features are extracted from the resized images.

Platform	Instagram	Facebook	X	LinkedIn
Profile Photo	320 x 320	170 x 170	400 x 400	400 x 400

Landscape	1080 x 566	1200 x 630	1600 x 900	1200 x 627
Portrait	1080 x 1350	630 x 1200	1080 x 1350	627 x 1200
Square	1080 x 1080	1200 x 1200	1080 x 1080	1080 x 1080
Stories/ Reels	1080 x 1920	1080 x 1920	-	-
Cover Photo	-	851 x 315	1500 x 1500	1128 x 191

Table 1: Commonly used/supported dimensions for various content types across different social media platforms (As of January 2024)

2.3.1.3 CONTENT ID CREATION

A content ID which is used as a pointer in the database is created. As the content ID needs to be unique, a hashing algorithm is used for its generation. However, the size of content ID needs to be as minimal as possible, as its information needs to be embedded in the host image, and larger size of content ID would mean increased size of the host image, and the possibility of visible distortions in the host image due to increased energy of the embedded watermark.

2.3.1.4 STEGANOGRAPHY CONSIDERATIONS

The content ID generated in the above steps needs to be embedded into the host image through steganography to ensure that the visual experience for the host image remains intact post the addition of it. Following aspects and design choices have been taken into consideration to make this a robust solution:

- **FAST FOURIER TRANSFORM:** As the content is expected to go through multiple modifications across social platforms, frequency domain is chosen as a preference over spatial domain for the steganography process as it is more robust to the changes that are generally expected. Fast Fourier Transform (FFT) has been chosen for this process considering its resistance to certain steganalysis techniques. FFT transforms an image from its spatial domain (pixel values) into the frequency domain. In the frequency domain, an image is represented as a combination of different sinusoidal waveforms, each with its own frequency, phase, and magnitude.

```
def embed_watermark(host, watermark, xmap, margins, alpha):
    if len(watermark.shape) == 2:
        watermark = np.repeat(watermark[:, :, np.newaxis], 3, axis=2)

    n_host = normalize_rgb(host)
    n_watermark = normalize_rgb(watermark)

    fft_host = np.fft.fft2(n_host, None, (0, 1))

    zero_filler_x = n_host.shape[0]//2-margins[0]*2
    zero_filler_y = n_host.shape[1]-margins[1]*2
    buffer = np.zeros((zero_filler_x, zero_filler_y, 3))
    buffer[:n_watermark.shape[0], :n_watermark.shape[1]] = n_watermark

    xh, xw = xmap[:2]

    fft_host[+margins[0]+xh, +margins[1]+xw] += buffer * alpha
    fft_host[-margins[0]-xh, -margins[1]-xw] += buffer * alpha

    ifft_host = np.fft.ifft2(fft_host, None, (0, 1))
    ifft_host = ifft_host.real
    ifft_host = np.clip(ifft_host, 0, 1)

    return ifft_host
```

Figure 5: Python code for watermarking the Content ID through Fast Fourier Transform (FFT)

- **ERROR CORRECTION CODING THROUGH 2D QR CODES:** To be able to successfully retrieve the embedded information from the watermarked image, an efficient Error Correction Coding is required. 2D QR Codes which have out of the box error correction coding capabilities have been chosen to address this requirement. The content ID is first encoded into a 2D QR Code with required error correction input, and the generated QR Code is watermarked into the host image. It is important to note that the higher the error correction code value, more the energy of the QR Code, and more the introduced distortions to the host image.

```
def generate_qrcode(text):
    text = f'<{text}>'
    rs = RSCodec(10)
    encoded_text = rs.encode(text.encode('utf-8'))

    qr = qrcode.QRCode(
        version=1,
        error_correction=qrcode.constants.ERROR_CORRECT_Q,
        box_size=10,
        border=4,
    )

    qr.add_data(encoded_text)
    qr.make(fit=True)
    qr_img = qr.make_image(fill_color="black", back_color="white")

    return qr_img
```

Figure 6: Python code for QR Code generation with Error Correction

- **MASKING THE WATERMARK WITH CRYPTOGRAPHY:** A watermarked image should further be protected from attackers who attempt to tamper the watermark and make it unfit for

verification. To achieve this, the index and sequence of frequency coefficients of the host image which are used to embed the watermark needs to be unpredictable and unreadable. This is achieved by using a *Master Key* which is only accessible to the Content Verification Platform, and generating random numbers using this Master Key as a seed, and creating a pattern of indexes from this random numbers, which are used to embed the watermark into the host image.

```
def generate_xmap(shape, key = None):
    xh = np.arange(size[0])
    xw = np.arange(size[1])

    if key:
        random.seed(key)
        for i in range(shape[0], 0, -1):
            j = random.randint(0, i)
            xh[i-1], xh[j] = xh[j], xh[i-1]

        for i in range(shape[1], 0, -1):
            j = random.randint(0, i)
            xw[i-1], xw[j] = xw[j], xw[i-1]

    xh = xh.reshape((-1, 1))

    return xh, xw
```

Figure 7: Python code for generating indices cryptographically for watermark embedding

2.3.2 CONTENT VERIFICATION

2.3.2.1 QR CODE CLEANUP

A watermark extracted from the host image often comes with some distortions and noise, especially when the host image has been altered in its dimensions or brightness. While the error correction coding helps in maintaining the structural integrity of the QR Code, *denoising* the image, and applying *thresholding* on the extracted watermark has been implemented to get a cleaned image which is more readable for any QR Code decoding.

2.3.2.2 FEATURE EXTRACTION IN GREY SCALE

For the sake of simplicity, feature extraction for image verification is performed on the grey scale versions of the original image and the image requiring verification. However, the same approach can be extrapolated by performing the feature extraction and comparison across each of the 3 channels, for luminous intensity of Red, Blue and Green if required.

2.3.2.3 FEATURE COMPARISON METRICS

Following 2 metrics have been implemented for the feature comparison, and to assess the similarity between the original image, and image under verification.

- **MEAN SQUARED ERROR (MSE):** MSE measures the average squared difference between pixel values in two images. A lower MSE indicates that the two images are more similar or have less distortion, while a higher MSE suggests greater dissimilarity or more significant distortion.

- **PEAK SIGNAL-TO-NOISE RATIO (PSNR):** PSNR measures the ratio of the peak signal level (the maximum possible pixel value) to the MSE between two images. PSNR is measured in decibels, and a higher PSNR value indicates better image quality and less distortion.

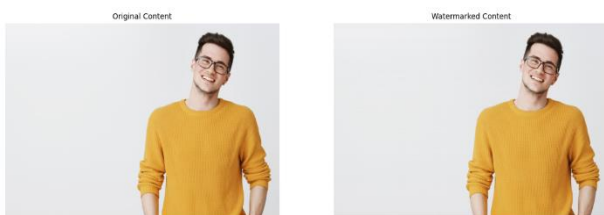
2.4 EXPERIMENT RESULTS

The *Framework for Content Verification System* has been tested with different scenarios, the results, and some of the key findings have been documented below.

2.4.1 SCENARIO: CONTENT REGISTRATION

A host image with the tag ‘Original Content’ in the below picture is taken as input, and a content ID generated for it is encoded into a 2D QR Code with error correction applied, and the resultant QR Code is embedded into the host image, the output of which is the image with the tag ‘Watermarked Content’ shown below.

Results: Watermarked Image is generated with no observable distortions, maintaining similarity to its original image.



[Image by cookie_studio on Freepik]

Figure 8: Images: Original & Watermarked

2.4.2 SCENARIO: CONTENT VERIFICATION OF A GENUINE IMAGE

An image with the tag ‘Validation Content’ in the below picture is taken as input, and is validated against its reference, the original content which is tagged as ‘Registered Content’ in the below picture. DCT is performed for 8x8 pixel blocks of grey scale images and they are compared with MSE & PSNR metrics.

Results: DCT derived from both the images look visually identical. MSE & PSNR results support this observation.



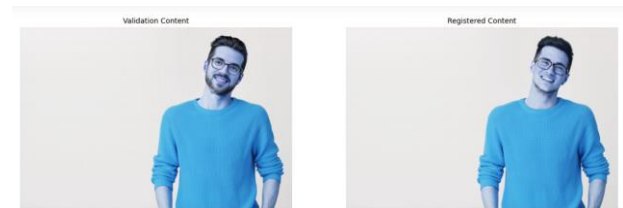
[Image by cookie_studio on Freepik]

Figure 9: Content Verification Images: Image for Validation & Registered Image

Mean Squared Error: 0.0
Peak Signal-to-Noise Ratio: inf

2.4.3 SCENARIO: CONTENT VERIFICATION OF SLIGHTLY/MODERATELY MORPHED IMAGE

An image with the tag ‘Validation Content’ in the below picture is taken as input, and is validated against its reference, ‘Registered Content’. The ‘Validation Content’ is *morphed*, by *face-swapping a new face* into the original image.



[Image by cookie_studio on Freepik]

Figure 10: Face-Swapped Image & Original Image

DCT is performed for 8x8 pixel blocks of grey scale images and they are compared with MSE & PSNR metrics.

Results: DCT derived from both the images have visible distinctions for the facial features. Higher MSE & relatively lower PSNR corroborate to this dissimilarity.



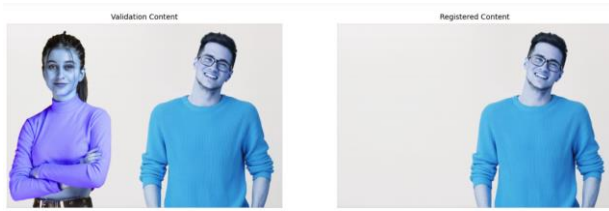
[Image by cookie_studio on Freepik]

Figure 11: Content Verification Images: Face-Swapped Image & Original Image

Mean Squared Error: 115.5239
Peak Signal-to-Noise Ratio: 27.5040

2.4.4 SCENARIO: CONTENT VERIFICATION OF HIGHLY MORPHED IMAGE

An image with the tag ‘Validation Content’ in the below picture is taken as input, and is validated against its reference, ‘Registered Content’. The ‘Validation Content’ is morphed, by *adding a new person* into the original image.



[Image by cookie_studio on Freepik]

Figure 12: Morphed Image & Original Image

DCT is performed for 8x8 pixel blocks of grey scale images and they are compared with MSE & PSNR metrics. Significantly high MSE & significantly low PSNR imply the high levels of distinction between the 'Validation Content' and the 'Registered Content'.

Results: DCT derived from both the images have clear visible distinctions



[Image by cookie_studio on Freepik]

Figure 13: Content Verification Images: Morphed Image & Original Image

Mean Squared Error: 2151.5111
Peak Signal-to-Noise Ratio: 14.8033

2.4.5 SCENARIO: IMPACT ON WATERMARK WHEN IMAGE IS BRIGHTENED

The image is brightened, introducing high energy changes to the frequencies of the image, including the frequency coefficients where watermark is embedded. Post-extraction operations of cleaning up the extracted watermark by performing denoising and thresholding are performed.

Results: The denoised QR Code was successfully decoded

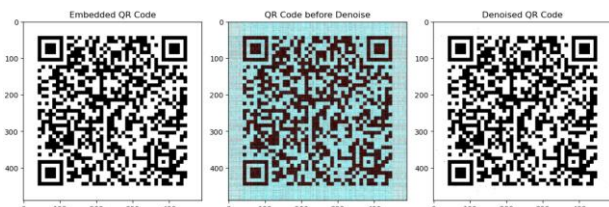


Figure 14: QR Codes: Embedded, extracted (Before Denoising), extracted (After Denoising)

2.5 FUTURE WORK

The results from the experiment were very promising, and the results recommend the following areas of improvement and exploration:

1. Improving efficiency in watermarking with lesser impact to the size of the original host image.
2. Certain transformations to the original image have been found to distort the watermark making it difficult to extract it. A redundant system, where watermarks are embedded at different frequencies, or where they are embedded in both spatial and frequency domains can make the image identification through watermarks more robust.
3. Mean Squared Error and Peak Signal-to-Noise Ratio have been observed to be more sensitive to certain changes to the images like brightness and filters. A redundant feature comparison system, which covers both spatial and frequency domains might be more useful and needs exploration.
4. There is scope for exploring alternatives to QR Code based watermarking to achieve error correction coding, which might be less in energy and can have lesser (or negligible) distortions to the original host image.

2.6 RECENT ADVANCES IN THE FIELD USING ALTERNATE APPROACHES

2.6.1 SOLUTIONS LEVERAGING WATERMARKING:

1. There is research proposing a hardware/device-based watermarking where every image/video captured from a device will be embedded with a unique watermark defining the source hardware of the created content.

Limitations with this Approach: Privacy concerns have been raised as personally identifiable information of the content creators will be embedded into the images/videos without an option to consent or disagree for their identification. It could take a long time for the implementation, distribution and adoption of these devices, and the risks of deep-fakes are expected to surface much sooner than the time it takes to implement these solutions.

Further, if an image is deep-faked, while its source can be identified, there is no real-time solution to validate its integrity, if it has been morphed or modified.

2. Another approach which is in adoption with the recent advances in generative AI involves embedding watermarks for every content that is generated by an AI platform.

Limitations with this Approach: With easy access to generative AI tools, and open-sourced frameworks, a bad actor can still generate fake content without any embedded watermark. There is no definitive way to differentiate a deep-fake content from a genuine one.

2.6.2 SOLUTIONS LEVERAGING ML & AI:

While this paper proposes a solution that validates the content against a registered entity, there is active research in the field of AI where GAN based models are built to identify if an image or a video is a deepfake or not. AI based solutions which look for facial cues that the current generative AI models haven't mastered yet, or solutions that look for anomalies in the scene transitions of a video are a few examples of current research areas.

3 CONCLUSION

In conclusion, this paper underscores the critical need for a robust content-integrity verification system in the era of widespread digital content sharing. This system empowers individuals and influencers to affirm the authenticity of their digital footprint, fortifying their reputations in an increasingly interconnected world.

As a significant defense against the rising menace of deepfakes, our framework not only provides a practical solution but also serves as a proactive measure against the potential proliferation of morphed, forged, and manipulated content on the internet. Much like the proactive stance taken by platforms like X in combating bots, our Content Verification System contributes to the ongoing battle against the spread of deceptive content, safeguarding the integrity of shared media.

The inclusion of code snippets for our implemented framework offers a tangible demonstration of the system's functionality, while discussions on alternative technologies provide valuable insights into the broader landscape of content-integrity solutions. This paper also highlights the vast scope for further development and refinement of the proposed Content Verification System.

As we navigate the evolving challenges of the digital age, the pursuit of innovative solutions remains paramount, and our work stands as a significant step forward in the ongoing efforts to maintain trust and authenticity in the digital landscape.

REFERENCES

- [1] Broda, M., Hajduk, V., Levický, D. and Kovac, O., 2016. Image steganography with using QR code and cryptography. *26th Conference Radioelektronika*.
- [2] Meenpal, A., Majumdar, S. and Balakrishnan, A., 2020. Digital Watermarking Technique using Dual Tree Complex Wavelet Transform. *First IEEE International Conference on Power, Control and Computing Technologies (ICPC2T), India*.
- [3] Vashistha, A., Nallusamy, R., Das, A. and Paul, S., 2010. Watermarking video content using visual cryptography and scene averaged image. *2010 IEEE International Conference on Multimedia and Expo, Singapore*, pp. 1641-1646. doi: 10.1109/ICME.2010.5583256.
- [4] Ibrahim, D.R., Teh, J.S. and Abdullah, R., 2021. An overview of visual cryptography techniques. *Multimed Tools Appl*, 80, pp.31927–31952.
- [5] Haouzia, A. and Noumeir, R., 2007. Methods for image authentication: a survey. *Springer Science + Business Media, LLC*.
- [6] Meerwald, P., 2000. Robustness and Security of Wavelet based Watermarking Algorithms.
- [7] Khalaf, A.A.M., Fouad, O., Hussein, A., Hamed, H., Kelash, H. and Ali, H., Hiding data in images using DCT steganography techniques with compression algorithms. 17. doi: 10.12928/telkomnika.v17i3.
- [8] Rana, M.S., Nob, M.N., Murali, B. and Sung, A.H., 2022. Deepfake Detection: A Systematic Literature Review. *IEEE Access*, 10, pp.25494-25513. doi: 10.1109/ACCESS.2022.3154404.
- [9] K. Sharma, "Tackling the deepfake menace – The dark side of AI," Times of India, March 1, 2024. [Online]. Available: <https://timesofindia.indiatimes.com/blogs/ai-musings/tackling-the-deepfake-menace-the-dark-side-of-ai/>. [Accessed: Mar 2, 2024]
- [10] CBS News, "Taylor Swift deepfakes spread online, sparking outrage," CBS News, January 26, 2024. [Online]. Available: <https://www.cbsnews.com/news/taylor-swift-deepfakes-online-outrage-artificial-intelligence/>. [Accessed: Mar 2, 2024]
- [11] H. Chen and K. Magramo, "Finance worker duped into paying out \$25 million to fraudsters using deepfake technology," *CNN*, Feb. 4, 2024.
- [12] [Online]. Available: <https://edition.cnn.com/2024/02/04/asia/deepfake-cfo-scam-hong-kong-intl-hnk/>. [Accessed: Mar 2, 2024]
- [13] R. Reddy, "24 Deepfake Statistics – Current Trends, Growth, and Popularity (December 2023)," ContentDetector.ai, Dec. 13, 2023. [Online]. Available: <https://contentdetector.ai/articles/deepfake-statistics>. [Accessed: Mar. 3, 2024].