Fırat Üniversitesi Deneysel ve Hesaplamalı Mühendislik Dergisi

FUJECE

U-Net ile Derin Steganografi: Video içerisine Metni Gizleme ve Yeniden Elde Etme

Mahmut SİNECEN [1]*

[1] Bilgisayar Mühendisliği Bölümü, Mühendislik Fakültesi, Adnan Menderes Üniversitesi , Aydın, Türkiye.
[1]mahmut@adu.edu.tr

Geliş Tarihi: 09.9.2025
Kabul Tarihi: 27.01.2026

Düzeltme Tarihi: 23.12.2025

doi: https://doi.org/10.62520/fujece.1780558
Araştırma Makalesi

Alıntı: M. Sinecen, "U-Net ile derin steganografi: video içerisine metni gizleme ve yeniden elde etme", Fırat Üni. Deny. ve Hes. Müh. Derg., vol. 5, no 1, pp. 345-362, Şubat 2026.

Öz

Video tabanlı steganografi, görüntü tabanlı yaklaşımlara kıyasla sunduğu yüksek veri taşıma kapasitesi ve gelişmiş görsel algılanamazlık özellikleri nedeniyle son yıllarda artan bir ilgi görmektedir. Bu çalışmada, U-Net mimarisi kullanılarak video içeriği içerisine metinsel bilgilerin gizlenmesi ve geri elde edilmesi amacıyla derin öğrenmeye dayalı bir steganografik yöntem önerilmektedir. Geleneksel en düşük anlamlı bit (LSB) tabanlı tekniklerden farklı olarak, önerilen yaklaşım; dayanıklılığı ve görsel kaliteyi artırmak amacıyla ilgi alanı (ROI) seçimi ve yama (patch) tabanlı yerleştirme stratejilerini kullanmaktadır. Metinsel veriler öncelikle görüntü tabanlı yamalara dönüştürülmekte ve eğitilmiş bir gizleme ağı aracılığıyla video karelerinin seçilen bölgelerine gömülmektedir. Gizlenen bilginin geri elde edilmesi için karşılık gelen bir çıkarım ağı kullanılmakta ve ardından metnin çıkarımı optik karakter tanıma (OCR) yöntemi ile gerçekleştirilmektedir. Deneysel sonuçlar, stego videolarda yüksek görsel bütünlük korunurken karakter geri kazanım doğruluğunun %81 ile %88 arasında değiştiğini göstermektedir. Önerilen ROI güdümlü U-Net tabanlı çerçeve, video akışlarında güvenli ve algılanamaz metin gizleme için etkili ve ölçeklenebilir bir çözüm sunmaktadır.

Anahtar kelimeler: Derin stenografi, U-Net, Video gizleme, Metin çıkarımı, OCR

*Yazışılan yazar

İntihal Kontrol: Yes – Turnitin
Şikayet: fujece@firat.edu.tr

Firat University Journal of Experimental and Computational Engineering

Deep Steganography with U-Net: Hiding and Revealing Text in Video

Mahmut SİNECEN [1]*

[1] Computer Engineering Department, Engineering Faculty Adnan Menderes University, Aydın, Türkiye.
[1]mahmut@adu.edu.tr

Received: 09.9.2025
Accepted: 27.01.2026
Revision: 23.12.2025
doi: https://doi.org/10.62520/fujece.1780558
Research Article

Citation: M. Sinecen "Deep steganography with U-Net: hiding and revealing text in video", Firat Univ. Jour.of Exper. and Comp. Eng., vol. 5, no 1, pp. 345-362, February 2026.

Abstract

Video-based steganography has attracted increasing attention due to its high payload capacity and improved imperceptibility compared to image-based approaches. In this study, a deep learning–based steganographic framework is proposed to embed and recover textual information within video content using the U-Net architecture. Unlike traditional least significant bit (LSB)–based techniques, the proposed method utilizes region-of-interest (ROI) selection and patch-based embedding to enhance robustness and visual quality. Textual data are first encoded into image patches and embedded into selected regions of video frames via a trained hiding network. A corresponding revealing network is employed to recover the hidden information, followed by an optical character recognition (OCR) pipeline for text extraction. Experimental results demonstrate character recovery accuracies between 81% and 88% while preserving high visual fidelity in the stego videos. This ROI-guided U-Net framework provides an effective and scalable solution for secure and imperceptible text hiding in video streams.

Keywords: Deep steganography, U-Net, Video hiding, Text recovery, OCR

*Corresponging author

Plagiarism Checks: Yes – Turnitin
Complaints: fujece@firat.edu.tr

1. Introduction

Steganography refers to the practice of hiding secret information within innocuous digital carriers such as images, audio, or video files in a manner that conceals the very existence of communication. The growing demand for secure and covert communication channels has renewed interest in steganography as an effective method for embedding sensitive data into unobtrusive media [1]. Unlike cryptography, which protects information by transforming it into an unintelligible form, steganography focuses on concealing the presence of the message itself within a cover object [2, 3]. Although both techniques aim to safeguard information, they differ fundamentally in their objectives and operational principles. This distinction underscores steganography's unique capability to provide plausible deniability, making it particularly suitable for covert information exchange [4, 5].

Rooted in the broader field of information hiding, steganography exploits various multimedia formats including images, audio, and video as cover media while striving to preserve imperceptibility against unauthorized observers [6, 7]. Historically, steganographic techniques have been employed in applications ranging from secure communication to digital watermarking; however, their dual-use nature also raises concerns due to potential misuse in illicit activities [8]. The primary objective of effective steganography is imperceptibility, ensuring that the embedded data remains undetectable not only to the human visual system but also to advanced computational steganalysis tools [9, 10]. Consequently, the continuous evolution of steganographic methods has been accompanied by parallel advances in steganalysis, giving rise to an ongoing arms race between data hiding and detection techniques [11, 12].

In recent years, deep learning has significantly influenced both steganography and steganalysis by enabling data-driven feature learning and improved modeling of complex patterns in multimedia content. Deep neural networks have facilitated more adaptive and robust embedding strategies, while simultaneously enhancing the accuracy of steganalytic detection methods, particularly in forensic analysis scenarios [13, 14]. Within this framework, the present study proposes a U-Net–based approach for embedding textual information into video data and subsequently recovering it. The remainder of this paper is organized as follows. Section 2 reviews related work, Section 3 describes the proposed methodology, Section 4 presents the experimental results, and Section 5 discusses the findings and concludes the study.

2. Literature Review

Steganography encompasses methods for storing and transmitting data by embedding it into different carriers. While early approaches often favored spatial-domain techniques, contemporary methods have shifted toward transform-domain and deep learning–assisted techniques [10, 11]. Video steganography presents unique challenges due to the dynamic structure of frames, requiring the preservation of confidentiality and robustness against various video compression standards.

The Least Significant Bit (LSB) technique, a classical method, has been widely used for a long time thanks to its simplicity. However, because it is vulnerable to statistical analysis, modern studies tend to employ a more complex and reliable approach [15, 16]. In this context, the use of deep neural networks has provided significant progress; combined with adaptive embedding techniques, researchers have developed systems more resistant to steganalysis [17].

In recent years, approaches based on Generative Adversarial Networks (GANs) have stood out. GAN-based methods can produce content statistically very similar to the cover media, thereby making the detection of hidden messages more difficult [18]. Moreover, neural networks are heavily utilized not only in steganography but also in steganalysis, with new detection algorithms being developed to discriminate between original and embedded content [19, 20].

Although the roots of steganography date back to ancient times, multimedia steganography has gained prominence in modern digital applications. The high-dimensional and complex nature of multimedia data, including images and videos, enables imperceptible embedding while offering improved robustness against

various manipulations and statistical steganalysis techniques [21, 22]. In this context, recent deep learning–based approaches aim to distribute secret information across multiple learned representations and, in the case of video data, across temporal frames to enhance imperceptibility and maintain robustness under compression or degradation.

Additionally, the literature highlights several strategies to improve the reliability and imperceptibility of steganographic systems, including controlling embedding rates, exploiting perceptual limitations of the human visual system, and leveraging learned feature representations through deep neural networks [4, 9]. In parallel, increasingly sophisticated analysis techniques developed for detecting hidden information continue to fuel the ongoing "arms race" between steganography and steganalysis [2, 17].

In conclusion, deep learning–based steganographic approaches have demonstrated improved imperceptibility and adaptive embedding capabilities compared to classical techniques, particularly using generative and encoder–decoder architectures [23]. While most prior studies focus on reconstructing image-based secrets, the present work extends this paradigm by incorporating an OCR-based post-processing stage, enabling direct textual information recovery from the reconstructed secret images.

Unlike existing learned or generative steganography approaches that primarily focus on image carriers such as encoder–decoder hiding/revealing networks, GAN-based formulations, reversible secret-to-image transformations, and diffusion-based generative schemes [10, 17, 23, 24] the proposed approach introduces an ROI-guided, patch-based U-Net architecture tailored specifically for video data. By embedding text-encoded image patches only into visually complex regions, the method enhances imperceptibility while maintaining reliable recovery performance. Moreover, integrating an OCR pipeline enables direct textual information retrieval, which is not the primary objective in most prior deep or generative steganography studies (Table 1).

Table 1. Comparison of deep learning–based generative/learned steganography approaches

Study	Carrier	Learning paradigm	Payload type	ROI-guided	Retrieval mechanism	Scope/Notes
Baluja (2017)	Image	CNN-based hiding + reveal networks (trained as a pair)	Image	No	Direct reconstruction (reveal network)	Image-in-image hiding; not designed for text extraction
Volkhonskiy et al. (2020)	Image	GAN-based (DCGAN/SEGAN-style) container generation and/or learned embedding	Binary secret message (and key)	No	Decryption/extraction network ("Bob")	Focus on image carriers; emphasizes (anti-) steganalysis via generative containers
Zhou et al. (2022)	Image (generated)	Flow-based generative steganography (Glow), reversible secret-to-image transformation (S2IRT)	Secret message	No	Reversible (inverse) transformation for message extraction	Coverless/generative formulation; high-capacity + high extraction accuracy reported
Wei et al. (2023)	Image	Diffusion-based generative steganography	Secret data	No	Diffusion-based generation with implicit message recovery	Diffusion-based generative steganography; message is embedded/recovered through the diffusion generation process
Proposed method	Video	ROI-guided patch-based U-Net hiding/revealing networks	Text (image-encoded patches)	Yes	Revealer network + OCR pipeline	Video carrier + explicit text retrieval; embedding constrained to visually complex ROIs (high imperceptibility with recoverable text)

3. Materials and Methods

The study focuses on embedding textual information into a video file and subsequently recovering the hidden text in a readable form. The input text is first converted into an image-based representation, where individual characters or character groups are encoded as small image patches. In parallel, the input video is decomposed into individual frames. For each selected frame, visually complex regions are identified and used as regions of interest (ROIs). Text-encoded image patches are then embedded into these ROIs using a deep learning–based hiding network, while preserving the perceptual quality of the video frames. After processing all selected frames, the resulting stego frames are recombined to form the final stego video. During the retrieval stage, a corresponding revealing network extracts the embedded patches from the stego frames, and the

recovered image patches are subsequently processed using an optical character recognition (OCR) pipeline to reconstruct the original text (Figure 1).

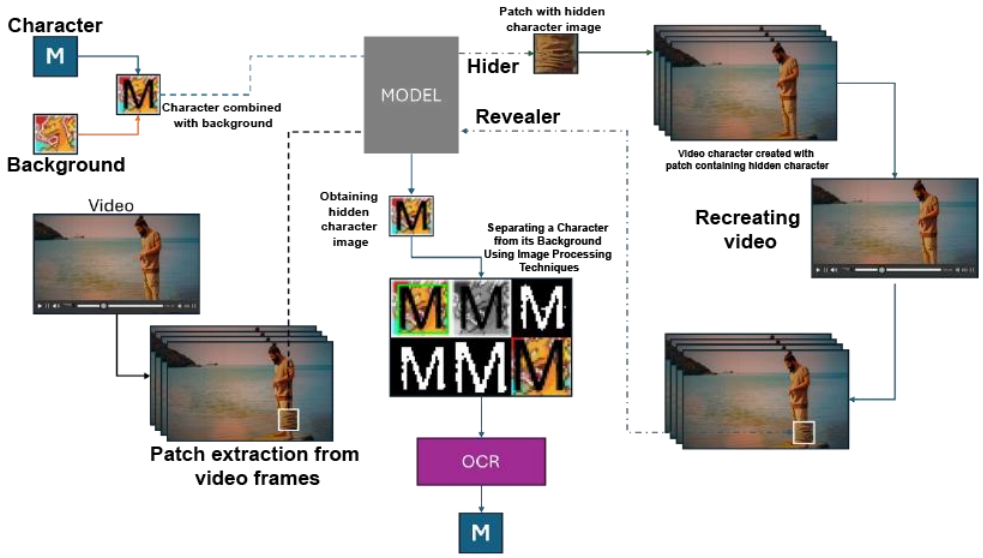

Figure 1. Overview of the proposed video-based steganography framework, including text-to-image encoding, ROI-guided patch embedding using hiding and revealing networks, and OCR-based text reconstruction

3.1. Materials

The experimental setup was constructed using a single publicly available video source downloaded from YouTube. The original video has a resolution of 1920 × 1080 pixels and a frame rate of 30 frames per second. The content of the video includes natural scenes with moderate motion and texture variations, making it suitable for evaluating imperceptible data embedding in realistic conditions.

From the complete video stream, a total of 4.500 frames were extracted, out of which 150 frames were selectively used for training and evaluation. This selection was performed to balance computational cost while maintaining experimental consistency and representative visual diversity. The selected frames were processed individually within the proposed steganographic pipeline.

Textual information was encoded into image-based payloads prior to embedding. Uppercase English characters (A–Z) were rendered using a fixed font configuration and converted into grayscale image patches. To analyze the impact of payload size on embedding performance and visual quality, three different patch sizes were employed: 32 × 32, 48 × 48, and 64 × 64 pixels. These patches were embedded into visually complex regions of selected video frames identified through region-of-interest (ROI) analysis.

All experiments were conducted on a workstation equipped with an NVIDIA RTX 3090 GPU and an AMD Ryzen 9 5950X CPU, providing sufficient computational power for deep learning–based training and inference. The software environment was configured using Python 3.10 and the PyTorch deep learning framework (version 2.3.1) with CUDA 11.8 support. Auxiliary libraries were utilized for image and video processing, numerical computation, visualization, and evaluation, including OpenCV, NumPy, Matplotlib, and Pandas. For perceptual quality assessment, the PyTorch-MSSSIM library was employed. Optical Character Recognition (OCR) experiments for text recovery were conducted using EasyOCR and Tesseract. Video decoding and frame extraction were handled using ImageIO and FFmpeg-based tools.

3.2. Method

3.2.1. Video frame extraction

This subsection describes the preprocessing steps applied to the carrier video prior to embedding. Using the yt-dlp library, a publicly available video is downloaded from YouTube at 1080p (1920×1080 pixels) and 30 frames per second. Each input video is decomposed into individual frames using OpenCV. Frames are converted to RGB color space and normalized to the [0,1] range.

3.2.2. ROI (Region of Interest) Selection

Regions with high spatial complexity are initially identified using Sobel edge detection, followed by thresholding and morphological operations to suppress noise and enhance prominent structures. Based on the resulting edge magnitude map, bounding boxes with sufficient texture density are selected as candidate embedding regions, since visually complex areas are less sensitive to subtle modifications.

In the proposed framework, this region-of-interest (ROI) selection strategy is directly integrated into the patch-based hiding process. When combining the secret text images with the video frames, a cover patch is extracted from the selected ROI, and the ROI is constrained to have the same spatial dimensions as the corresponding secret text image. The ROI location is determined by evaluating the magnitude of horizontal and vertical gradient responses obtained from the Sobel edge detector. By selecting the region with the highest gradient energy, the most visually complex area of the frame is chosen, thereby making the embedded text significantly more difficult to perceive in the resulting stego video.

To compute the gradient magnitude used for ROI selection, each grayscale frame is convolved with the Sobel operator using 3×3 kernels that approximate first-order derivatives in the horizontal and vertical directions (Eqs. 1–2).

$$K_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}, K_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \quad (1)$$

$$G_x = K_x * I, \quad G_y = K_y * I \quad (2)$$

Here "*" denotes 2-D convolution. These kernels can be seen as a separable combination of smoothing with (1, 2, 1) and differencing (derivative) with (−1, 0, 1), which makes them more robust to noise [25, 26].

The per-pixel gradient vector, the magnitude and the direction were calculated using Eqs. 3-4-5.

$$\nabla I = [G_x, G_y] \quad (3)$$

$$M = \sqrt{G_x^2 + G_y^2} \quad (4)$$

$$\theta = atan2(G_y, G_x) \quad (5)$$

With this method, the patch extracted from the frame is detected after passing through successive stages as illustrated in Figure 2.

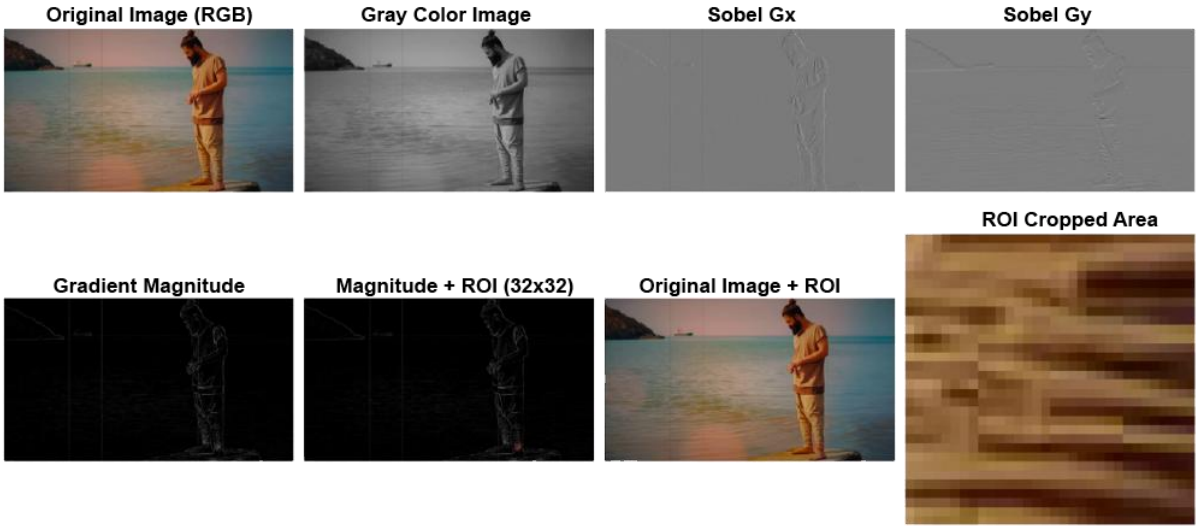

Figure 2. ROI detection process using the Sobel operator. The gradient magnitude map is thresholded and post-processed to identify visually complex regions. The selected ROI (x = 1400, y = 968, w = 32, h = 32) is used as the cover patch for text embedding

The primary motivation for using gradient magnitude in ROI detection is that it effectively measures the strength of local intensity variations, producing high responses at object boundaries and textured regions. This property is well suited for identifying visually complex areas that can better mask embedding distortions. In addition, the gradient magnitude is invariant to edge orientation, enabling robust ROI selection without prior knowledge of dominant directions [27]. Due to the inherent smoothing effect of the Sobel operator, the method is less sensitive to high-frequency noise, allowing thresholding and adaptive thresholding strategies to operate more reliably [28]. Finally, the sequential application of thresholding, morphological merging, and connected component analysis enables efficient extraction of bounding boxes, yielding a simple and computationally efficient ROI selection pipeline.

3.2.3. Text-to-ımage encoding

Text strings are rendered into grayscale image patches using a fixed font type and size, and these patches are used as payload inputs to the hiding network. Each character in the input text is first converted into an individual image, referred to as a secret image.

A predefined background image with moderate visual complexity is combined with each character image prior to embedding. Empirical observations during preliminary experiments indicated that character-only images (i.e., images containing only foreground text on a uniform background) tend to suffer from severe degradation or partial disappearance after the embedding and recovery process. This effect is attributed to the low structural and intensity variation in such images, which makes them more vulnerable to distortion during deep feature transformation and reconstruction.

To mitigate this issue, a background image is introduced to provide additional spatial texture and intensity variation, thereby stabilizing the learning and reconstruction of character features. The character is rendered in the center of the background image, ensuring consistent alignment across samples.

Character image sizes of 32×32, 48×48, and 64×64 pixels were evaluated independently to analyze the impact of payload size on embedding performance and recovery accuracy. As illustrated in Figure 3, the secret text “MAHMUTSINECEN” was rendered using the Arial font and placed centrally on the background

image, with the font size occupying approximately 80% of the image area. This process resulted in 13 individual character images, each stored as a separate secret patch.

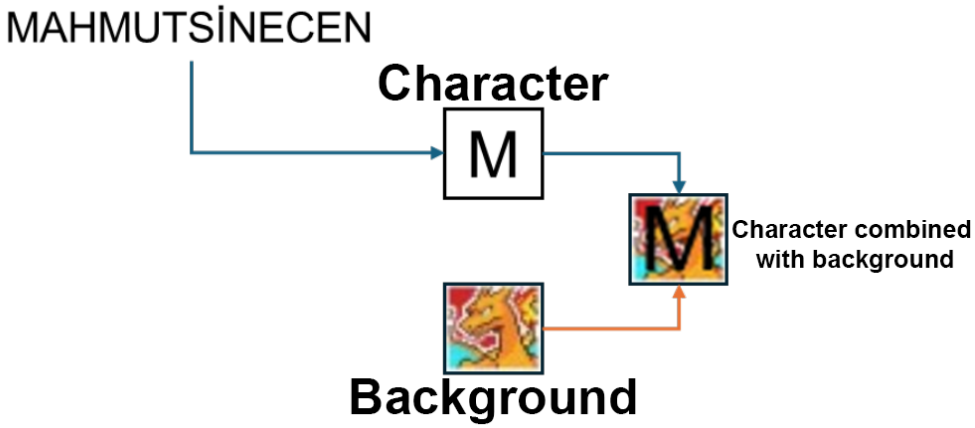

Figure 3. Generation of secret image patches by combining individual text characters with a predefined background image prior to embedding

3.2.4. U-Net-based hiding and revealing networks

The proposed framework consists of two complementary deep learning components: a hiding network and a revealing network. The hiding network takes a cover video frame together with an encoded text patch as input and generates a stego frame in which the secret information is imperceptibly embedded. Conversely, the revealing network operates on the stego frame to reconstruct the hidden text patch. To effectively capture both global contextual information and fine-grained spatial details during these processes, skip connections are employed to enable multi-scale feature fusion.

Accordingly, a U-Net–based architecture is adopted due to its proven capability to preserve spatial resolution while modeling high-level contextual representations. The contracting and expanding paths of the U-Net architecture allow the network to learn robust hierarchical features, which is particularly beneficial for localized, patch-scale information hiding tasks. Building upon the standard encoder–decoder U-Net structure [29], the network is further enhanced with residual convolutional blocks, Group Normalization, and SiLU (Swish) activation functions to improve training stability and convergence behavior. In addition, Squeeze-and-Excitation (SE) blocks are incorporated to emphasize informative inter-channel dependencies, enabling adaptive allocation of embedding capacity to textured and edge-rich regions while restricting modifications in visually smooth areas [30].

This architectural configuration yields a computationally efficient and stable backbone for both hiding and revealing operations, making it well suited for localized, patch-based steganographic embedding and recovery [30-32]. During the retrieval phase, the revealing network receives not only the stego patch (containing both cover and embedded information) but also multi-scale high-pass and Laplacian feature summaries. These inputs, resulting in a total of 12 channels (RGB stego patch concatenated with residual feature maps), are included to amplify subtle embedding traces distributed over fine textures. Such high-pass

residual–based representations are commonly adopted in steganalysis and have been shown to facilitate more accurate recovery of hidden information by enhancing weak signal components [33].

The internal structure of the proposed network, including the residual convolutional block (GroupNorm + SiLU + SE with residual connections) and the U-shaped downsampling and upsampling paths, is illustrated in Figure 4.

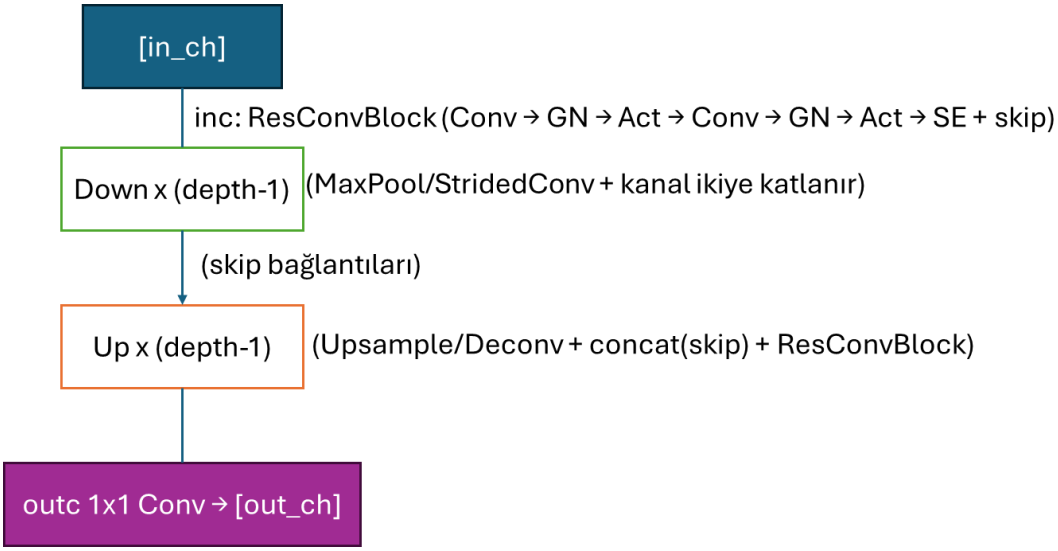

Figure 4. Internal architecture of the proposed U-Net–based hiding and revealing networks, illustrating the residual convolutional blocks, skip connections, and downsampling/upsampling paths

Training strategy

The training procedure is designed to ensure stable convergence and balanced optimization of both hiding and revealing tasks. To this end, a two-stage training strategy is employed.

In the first stage, a warm-up phase is applied in which only the revealing network is trained using stego–secret pairs. This partial supervision allows the revealing network to learn a reliable stego-to-secret mapping before being exposed to the joint optimization process. Such pre-training stabilizes subsequent training and prevents early-stage error amplification when both networks are optimized simultaneously.

In the second stage, the hiding and revealing networks are trained jointly in an end-to-end manner. Optimization is performed using the AdamW optimizer with decoupled weight decay, combined with a ReduceLROnPlateau learning rate scheduler to adaptively adjust the learning rate based on validation performance. This strategy enables stable convergence while mitigating overfitting.

To enhance robustness against common video distortions and post-processing operations, data augmentation techniques are applied during training. These include additive color noise, mild Gaussian blur, JPEG compression artifacts, and gamma correction. Such augmentations encourage the network to learn invariant embedding patterns that remain recoverable under realistic degradation scenarios.

In addition, channel masking is employed during embedding. Specifically, secret patches are embedded into a single-color channel (e.g., the blue channel) using a Modified RGB (MRGB) representation. This constraint reduces perceptual distortion in luminance-sensitive channels and further improves visual imperceptibility.

Training losses

The overall training objective is formulated as a multi-component loss function that jointly balances visual fidelity of the stego frames and accurate recovery of the embedded secret content.

To preserve the visual quality of the cover frames, a cover reconstruction loss is defined as a combination of the L1 loss and the Structural Similarity Index Measure (SSIM). This loss encourages the generated stego frames to remain perceptually close to the original cover frames while penalizing pixel-level deviations[34].

For secret recovery, a weighted L1 loss is employed, where higher weights are assigned to darker regions of the secret patches to improve character legibility after extraction. In addition, a gradient-based L1 loss is introduced to preserve edge structures and character contours, which are critical for successful OCR-based text recognition. SSIM is further incorporated to enforce structural consistency between the original and recovered secret patches.

To improve robustness, a residual-energy regularization term is added to penalize excessive high-frequency embedding artifacts. Moreover, controlled Gaussian noise is injected into the stego representations during training. This noise injection acts as a regularizer, encouraging the revealing network to recover secret information under slight perturbations and enhancing resilience against compression and signal degradation.

Hider (Embedding)

U-Net (6→4): the 6-channel input (cover + secret) yields 3 residual channels + 1 mask. The mask uses sigmoid, the residual uses tanh, and with a color-channel constraint and factor α, the residual is added onto the cover (Figure 5). The color-channel constraint (e.g., write only to the B channel) and α are set parametrically. Formal definitions and Eqs. (6–10) follow the paper; the mask limits writing to a single channel and α is a small mixing coefficient (in this study, the B channel was used with a small α).

$$X = [C, S] \in \mathbb{R}^{6xsxs} \tag{6}$$

$$Y = UNet(X) \tag{7}$$

$$m = \sigma(Y_4) \tag{8}$$

$$r = \tanh(Y_{1:3}) \odot m \odot M_{rgb} \tag{9}$$

$$\hat{C} = clip(C + \alpha\, r, 0,1) \tag{10}$$

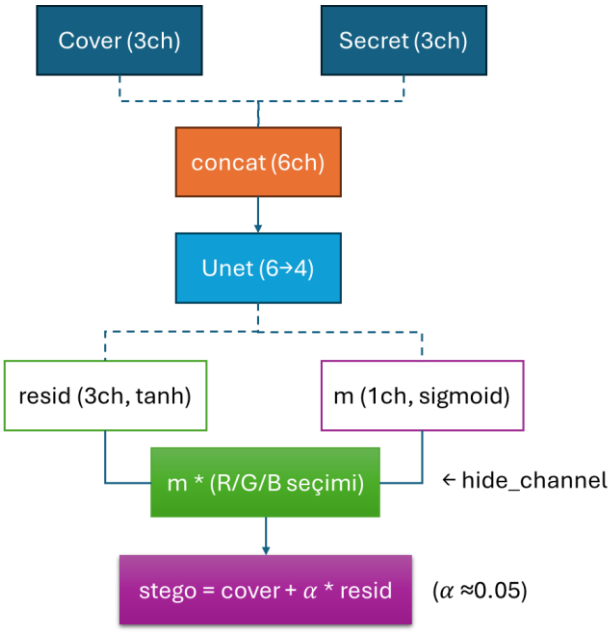

Figure 5. Hider (Embedder) diagram

Revealer (Recovery)

Using multi-scale high-pass (HP) emphasizes faint traces of the hidden signal distributed over fine textures [33]. In this study, three-scale high-pass summaries (two different HP + Laplacian) were produced and concatenated as 9 channels (Figure 6). U-Net (12→3) with a sigmoid output performs recovery, taking the stego patch plus its multi-scale HP/Laplacian derivatives as input (Eqs. 11–12).

$$U = concat\left(\hat{C}, HP_1(\hat{C}), HP_2(\hat{C}), Lap(\hat{C})\right) \in \mathbb{R}^{12xsxs} \tag{11}$$

$$\hat{S} = \sigma\left(UNet(U)\right) \tag{12}$$

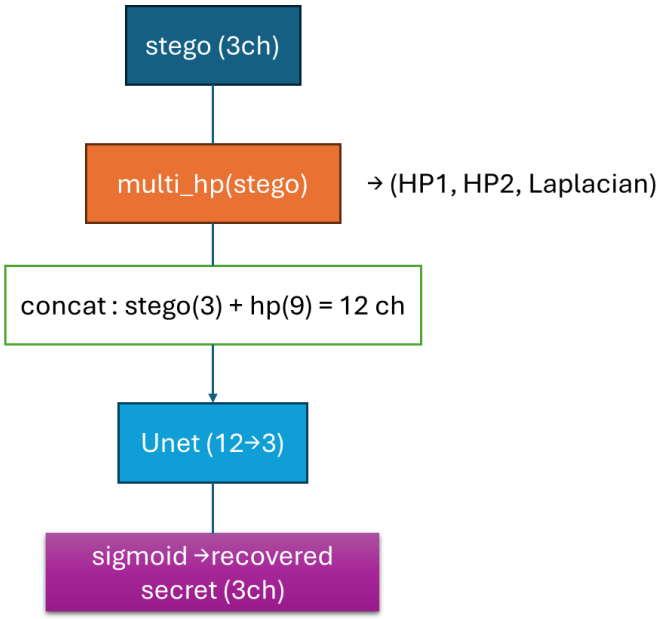

Figure 6. Revealer (Extractor) diagram

3.2.5. Loss functions and training

SSIM is used to maintain a perceptual balance for both cover and secret. Optimization uses AdamW [35] and the ReduceLROnPlateau scheduler, which lowers the learning rate when a monitored metric (e.g., validation loss or accuracy) stops improving for a period, allowing the model to proceed with smaller steps when it gets "stuck" [36-39]. Training proceeds in two stages: a short revealer pre-training, then joint training. The components include a cover loss limiting visual distortion (Eq. 13), a secret loss promoting accurate recovery and quality (Eq. 14) with brightness-aware weighting (Eq. 15), a residual energy term that accentuates visible traces for the stego image (Eq. 16), and the total loss (Eq. 17).

$$\mathcal{L}_{cover} = \left\|\hat{C} - C\right\|_1 + \lambda_{ssim}\left(1 - SSIM\left(\hat{C}, C\right)\right) \tag{13}$$

$$\mathcal{L}_{secret} = \left\|\omega \odot \left(\hat{S} - S\right)\right\|_1 + \mu\left\|\nabla\hat{S} - \nabla S)\right\|_1 + \eta\left(1 - SSIM\left(\hat{S}, S\right)\right) \tag{14}$$

$$\omega = 0.3 + 0.7\left(1 - Gray(S)\right)^2 \tag{15}$$

$$\mathcal{L}_{resid} = \|r\|_2^2 \tag{16}$$

$$\mathcal{L} = \lambda_1\mathcal{L}_{cover} + \beta\mathcal{L}_{secret} + \rho\mathcal{L}_{resid} \tag{17}$$

The total loss function is defined as:

$$L_{total} = \lambda_{loss}L_{hide} + (1 - \lambda_{loss})L_{reveal} \tag{18}$$

Where L_{hide} represents reconstruction loss and L_{reveal} denotes cover reconstruction loss.

The complete set of training parameters used during the optimization process is provided in Table 2.

Table 2. Training parameters

Parameter	Value
Epochs	100
Batch size	8
Learning rate	0.0001
Optimizer	AdamW
Loss weight (λ_{loss})	0.75
Input size	256 × 256

3.3. Constructing the video with hidden text

Using the proposed U-Net–based hiding network, stego patches containing the embedded character information were generated for each selected video frame. These stego patches were subsequently placed back into their corresponding regions of interest (ROIs) at the original spatial locations from which the cover patches were extracted. This ensured spatial consistency and prevented any unintended distortion outside the embedding regions.

After all selected frames were processed, the modified frames were reassembled to reconstruct the final stego video. Video encoding was performed using OpenCV with an FFmpeg-based codec, while preserving the original spatial resolution and frame rate of the input video. Maintaining identical encoding parameters ensured that any observed distortions were attributable solely to the embedding process rather than to re-encoding artifacts.

No additional post-processing or compression was applied beyond standard video encoding, allowing a fair assessment of the embedding-induced distortions.

4. Results

Figure 7 shows the histogram changes in each color channel for (i) the patch from the video frame, (ii) the backgrounded character image to be hidden, and (iii) the embedded and recovered images produced by the U-Net model. Only minor distortions were observed in images; the recovered images carrying the secret message largely represent the original character. As seen in Figure 7, the frequency distributions of the Red, Green, and Blue channel pixels in the original image do not change significantly, and as a result there is no perceptible visual degradation.

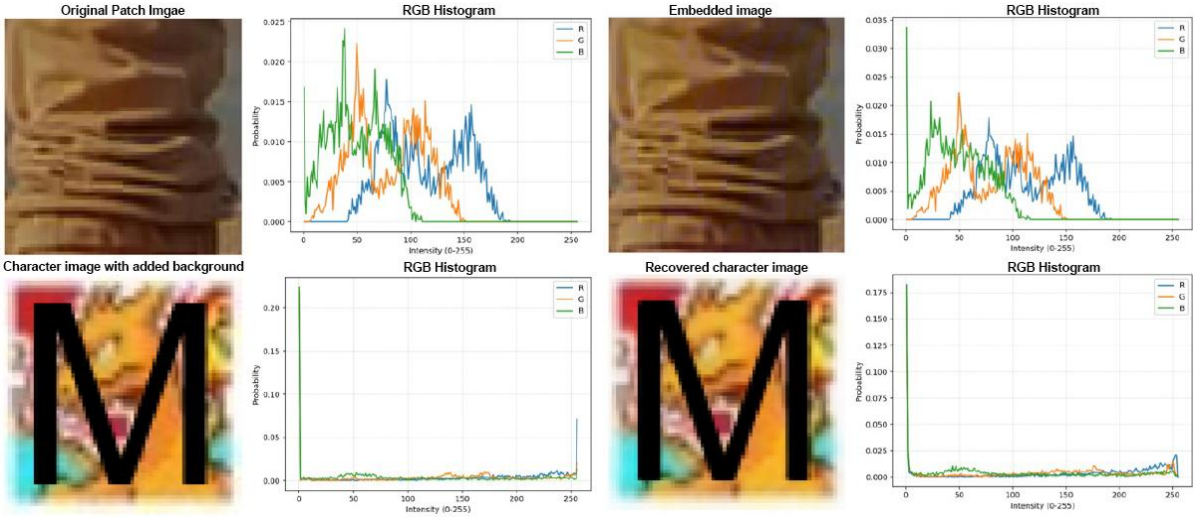

Figure 7. Illustrates the RGB histogram distributions for (i) the original patch extracted from the video frame, (ii) the character image combined with a background prior to embedding, and (iii) the corresponding embedded and recovered character images generated by the proposed U-Net–based framework

The results indicate that the embedding process introduces only minimal alterations to the pixel intensity distributions across the Red, Green, and Blue channels. In particular, the histogram profiles of the embedded patches remain closely aligned with those of the original cover patches, suggesting that the statistical properties of the video content are largely preserved. This behavior is essential for maintaining imperceptibility and reducing the risk of detection by both human observers and histogram-based steganalysis methods.

The recovered character images exhibit histogram distributions that closely resemble those of the original character images with background, demonstrating that the revealing network successfully reconstructs the embedded information with minimal loss. Despite slight smoothing effects introduced by the embedding and recovery stages, the character structures remain visually intact and sufficiently distinguishable, which directly supports the high OCR-based character recovery accuracies reported in subsequent evaluations.

4.1. Quantitative evaluation

To convert the recovered character images back into meaningful text, the character must be separated from its background and recognized via OCR. Different preprocessing variants were tested before OCR to assess their impact on performance. The methods used for these variants are listed in Table 3. For each variant, an index string is generated (e.g., thr40_adnone_mopen_k1_ma10_pad2_easy), and the recognition accuracy is computed by comparing the OCR output with the ground truth. Table 4 shows sample index values among the 288 variants. Figure 8 plots accuracies for 32/48/64-pixel images by variant index. The top-performing OCR results are summarized in Table 5.

Table 3. Methods and variants used to separate the character from the background prior to OCR

Process	Variations
Threshold	[40, 80, 120]
Adaptive Thresholding	[“none”, “gauss”]
Morphological Operation	[“none”, “open”, “close”]
Kernel	[0, 1]
Noise Elimination	[5, 10]
Pad	[1, 2]
OCR	[“Tesseract”, “EasyOCR”]

Table 4. Example variant index values

Variation	Variation Index
thr80_adnone_mclose_k1_ma10_pad2_easy	1
	. . .
thr80_adnone_mclose_k1_ma10_pad1_easy	20
	. . .
thr40_adgauss_mopen_k0_ma5_pad1_easy	96
	. . .
thr120_adgauss_mopen_k1_ma10_pad1_tess	126
	. . .
thr120_adnone_mclose_k1_ma5_pad2_tess	256
	. . .
thr40_adgauss_mclose_k1_ma5_pad1_tess	288

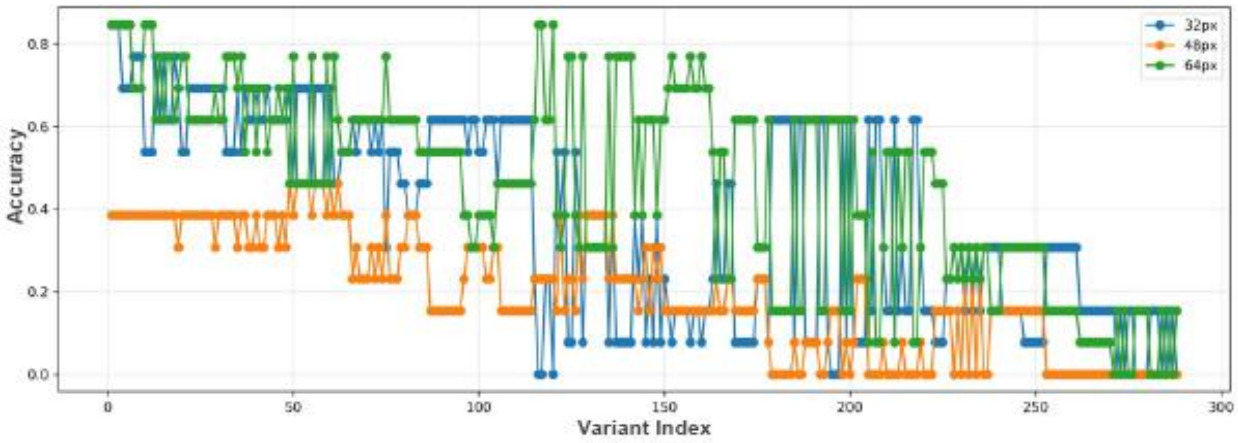

Figure 8. Accuracy of 32/48/64-pixel images by variant index

Table 5. OCR accuracy results

Pixel Size	Variant	Accuracy (%)
64	thr40_adnone_mopen_k1_ma10_pad2_easy	0.88
48	thr80_adnone_mclose_k0_ma10_pad2_easy	0.81
32	thr40_adnone_mclose_k0_ma10_pad2_easy	0.83

The error matrix of the highest accuracy results found because of hiding and retrieving the "MAHMUTSİNECEN" text used within the scope of the study is shown in Figure 9.

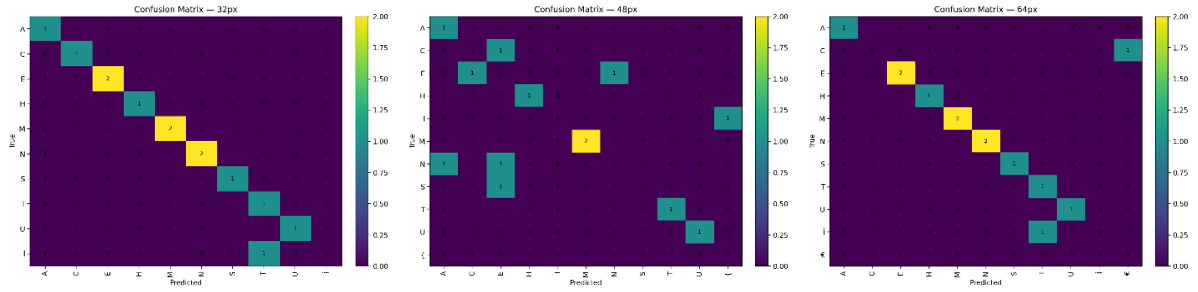

Figure 9. Confusion matrices for the recovered secret text at 32/48/64-pixel image sizes

4.2. Ablation study

The ablation study demonstrates the importance of ROI selection and high-pass feature extraction. Removing ROI selection significantly reduces both recovery accuracy and visual quality, confirming its effectiveness in preserving imperceptibility. As shown in Table 6, these results quantitatively confirm that ROI-guided embedding and high-pass–assisted recovery are critical components of the proposed framework, jointly contributing to both imperceptibility and reliable text reconstruction.

Table 6. Ablation study results

Configuration	Accuracy (%)	SSIM
Full model	87.9	0.94
No ROI	79.3	0.89
No HP filters	82.1	0.91
RGB embedding	80.7	0.88

5. Discussion

Steganography aims to conceal the existence of secret communication by embedding information within an innocuous carrier while preserving the carrier's perceptual integrity. From this perspective, video-sharing platforms that are publicly accessible worldwide provide a particularly suitable medium for steganographic applications, as videos can be distributed and consumed without arousing suspicion. Without specialized analysis tools and knowledge of the embedding mechanism, hidden messages remain indistinguishable from benign visual content.

Motivated by these considerations, the present study proposes a deep learning–based framework for embedding and retrieving textual information from video data. Unlike classical least significant bit (LSB)–based methods, which rely on direct bit-level replacement and are vulnerable to statistical steganalysis, the proposed approach leverages a U-Net–based architecture combined with ROI-guided embedding. This design enables the secret information to be adaptively distributed within visually complex regions of video frames, thereby improving imperceptibility and resistance to detection. Moreover, restricting access to the hidden content through a trained neural network further increases the difficulty of unauthorized recovery.

One practical challenge observed in the experiments is the trade-off between payload resolution and computational cost. Increasing the resolution of embedded character patches significantly raises the computational burden during training and inference, as model complexity and memory consumption grow accordingly. In this study, patch sizes of 32×32, 48×48, and 64×64 pixels were selected to balance recovery accuracy, processing time, and visual quality. While higher resolutions could mitigate the impact of video compression artifacts, they may also increase the risk of perceptible distortions within the carrier frames.

Although the experiments in this study were conducted on uppercase English characters, extending the pipeline to diacritics commonly used in Turkish (e.g., ğ, ö, ü, and dotted i) remains challenging due to OCR

and preprocessing sensitivity. These errors stem primarily from binarization, morphological filtering, and OCR model limitations, rather than from the embedding process itself. As a result, OCR accuracy constitutes a bottleneck in the end-to-end system performance.

Compared to GAN-based steganography approaches, this ROI-guided U-Net framework demonstrates improved training stability and reduced computational complexity, as it avoids adversarial optimization and mode-collapse issues commonly associated with GANs. The encoder–decoder structure with skip connections enables effective multi-scale feature fusion while maintaining stable convergence. Nevertheless, the dependency on OCR for final text recovery remains a limitation of the current implementation.

Future research will focus on enhancing robustness against video compression artifacts introduced by different codecs and extending the framework to support multilingual text. Additional investigations will explore alternative deep learning architectures, parameter optimization strategies, and hardware configurations to reduce training time and enable real-time or near-real-time text retrieval. Furthermore, improving OCR robustness—particularly for low-resolution characters, diacritics, and punctuation—will be a key objective to ensure accurate sentence-level reconstruction.

6. Author Contribution Statement

In the study, Author 1 contributed to the idea, design, literature review, experimental study and writing of the article.

7. Ethics Committee Approval and Conflict of Interest

Ethics committee approval is not required for this study. “There is no conflict of interest with any person/institution in the prepared article.”

8. Ethical Statement Regarding the Use of Artificial Intelligence

No artificial intelligence-based tools or applications were used in the preparation of this study. The entire content of the study was produced by the author in accordance with scientific research methods and academic ethical principles.

9. References

[1] M. A. Idakwo, M. B. Muazu, A. E. Adedokun and B. O. Sadiq, "An extensive survey of digital image steganography: State of the art," arXiv:2404.19548, 2024.
[2] I. I. Araujo and H. Kazemian, "Vulnerability exploitations using steganography in PDF files," *Int. J. Comput. Netw. Appl.*, vol. 7, no. 1, p. 10, 2020.
[3] F. Nabi and M. M. Afzal, "Image steganography: Critical findings through some novel techniques," *Int. J. Innov. Technol. Explor. Eng.*, vol. 9, no. 5, pp. 878–890, 2020.
[4] H. Li *et al*., "Smaller is bigger: Rethinking the embedding rate of deep hiding," 2023.
[5] K. Koptyra and M. R. Ogiela, "Distributed steganography in PDF files—Secrets hidden in modified pages," *Entropy*, vol. 22, no. 6, p. 600, 2020.
[6] A. D. Wiranata and R. T. Aldisa, "Aplikasi steganografi menggunakan least significant bit (LSB) dengan enkripsi Caesar Chipper dan Rivest Code 4 (RC4) menggunakan bahasa pemrograman JAVA," *J. JTIK (J. Teknol. Inf. dan Komun.)*, vol. 5, no. 3, pp. 277–281, 2021.
[7] M. A. Ali Khodher and T. W. Aldeen Khairi, "Review: A comparison steganography between texts and images," *J. Phys.: Conf. Ser.*, vol. 1591, no. 1, p. 012024, 2020.
[8] J. Kose, O. B. Chia and V. Baboolal, "Review and test of steganography techniques," 2020.
[9] Q. Li *et al*., "A novel grayscale image steganography scheme based on chaos encryption and generative adversarial networks," *IEEE Access*, vol. 8, pp. 168166–168176, 2020.
[10] P. Wei, Q. Zhou, Z. Wang, Z. Qian, X. Zhang and S. Li, "Generative steganography diffusion," 2023.
[11] A. K. Sahu and M. Sahu, "Digital image steganography and steganalysis: A journey of the past three decades," *Open Comput. Sci.*, vol. 10, no. 1, pp. 296–342, 2020.
[12] J. Fridrich, *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge, U.K.: Cambridge Univ. Press, 2012.
[13] L. Zeng, W. Lu, W. Liu and J. Chen, "Deep residual network for halftone image steganalysis with stego-signal diffusion," *Signal Process.*, vol. 172, p. 107576, 2020.
[14] J. Wang *et al*., "Fighting malicious media data: A survey on tampering detection and deepfake detection," 2022.
[15] S. Rahman *et al*., "A novel and efficient digital image steganography technique using least significant bit substitution," *Sci. Rep.*, vol. 15, no. 1, p. 107, 2025.
[16] B. Chen, W. Luo, P. Zheng and J. Huang, "Universal stego post-processing for enhancing image steganography," *J. Inf. Secur. Appl.*, vol. 55, p. 102664, 2020.
[17] D. Volkhonskiy, I. Nazarov and E. Burnaev, "Steganographic generative adversarial networks," in *Proc. 12th Int. Conf. Mach. Vis. (ICMV 2019)*, SPIE, 2020, p. 97.
[18] J. Liu *et al*., "Recent advances of image steganography with generative adversarial networks," *IEEE Access*, vol. 8, pp. 60575–60597, 2020.
[19] S. Rahman *et al*., "A comprehensive study of digital image steganographic techniques," *IEEE Access*, vol. 11, pp. 6770–6791, 2023.
[20] R. Chaganti, V. Ravi, M. Alazab and T. D. Pham, "Stegomalware: A systematic survey of malware hiding and detection in images, machine learning models and research challenges," 2021.
[21] R. Apau, M. Asante, F. Twum, J. Ben Hayfron-Acquah and K. O. Peasah, "Image steganography techniques for resisting statistical steganalysis attacks: A systematic literature review," *PLoS One*, vol. 19, no. 9, p. e0308807, 2024.
[22] Y. Sanjalawe, S. Al-E'mari, S. Fraihat, M. Abualhaj and E. Alzubi, "A deep learning-driven multi-layered steganographic approach for enhanced data security," *Sci. Rep.*, vol. 15, no. 1, p. 4761, 2025.
[23] Z. Zhou, Y. Su, Q. M. J. Wu, Z. Fu and Y. Shi, "Secret-to-image reversible transformation for generative steganography," 2022.
[24] M. Shukor, B. B. Damodaran, X. Yao and P. Hellier, "Video coding using learned latent GAN compression," in *Proc. 30th ACM Int. Conf. Multimedia*, New York, NY, USA: ACM, 2022, pp. 2239–2248.
[25] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed. New York, NY, USA: Pearson, 2018.

[26] I. Sobel and G. Feldman, "A 3×3 isotropic gradient operator for image processing," 1968.
[27] *scikit-image: Image processing in Python*. [Online]. Available: https://scikit-image.org/
[28] *OpenCV documentation index*. [Online]. Available: https://docs.opencv.org/
[29] O. Ronneberger, P. Fischer and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," 2015.
[30] J. Hu, L. Shen and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 7132–7141.
[31] Y. Wu and K. He, "Group normalization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 3–19.
[32] P. Ramachandran, B. Zoph and Q. V. Le, "Searching for activation functions," in *Proc. Int. Conf. Learn. Represent. (ICLR), Workshop Track*, 2018.
[33] J. Fridrich and J. Kodovsky, "Rich models for steganalysis of digital images," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 3, pp. 868–882, 2012.
[34] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.
[35] S. Baluja, "Hiding images in plain sight: Deep steganography," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 1–11.
[36] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019.
[37] *ReduceLROnPlateau — PyTorch documentation*. [Online]. Available: https://docs.pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.html
[38] *ReduceLROnPlateau — Keras API documentation*. [Online]. Available: https://keras.io/api/callbacks/reduce_lr_on_plateau/
[39] I. Goodfellow, Y. Bengio and A. Courville, "Optimization for training deep models," in *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016, pp. 271–325.