

Unified Steganography via Implicit Neural Representation

Qi Song¹, Ziyuan Luo¹, Xiufeng Huang¹, Sheng Li², Renjie Wan^{1*}

¹Department of Computer Science, Hong Kong Baptist University

²School of Computer Science, Fudan University

{qisong, ziyuanluo, xiufenghuang}@life.hkbu.edu.hk

lisheng@fudan.edu.cn, renjiewan@hkbu.edu.hk

Abstract

Digital steganography is the practice of concealing for encrypted data transmission. Typically, steganography methods embed secret data into cover data to create stega data that incorporates hidden secret data. However, steganography techniques often require designing specific frameworks for each data type, which restricts their generalizability. In this paper, we present U-INR, a novel method for steganography via Implicit Neural Representation (INR). Rather than using the specific framework for each data format, we directly use the neurons of the INR network to represent the secret data and cover data across different data types. To achieve this idea, a private key is shared between the data sender and receivers. Such a private key can be used to determine the position of secret data in INR networks. To effectively leverage this key, we further introduce a key-based selection strategy that can be used to determine the position within the INRs for data storage. Comprehensive experiments across multiple data types, including images, videos, audio, and SDF and NeRF, demonstrate the generalizability and effectiveness of U-INR, emphasizing its potential for improving data security and privacy in various applications.

Keywords

Digital steganography, data hiding, implicit neural representation.

1 Introduction

Digital steganography aims to hide information within digital data, such as images, audio, or video, to achieve encrypted transmission of information. Typically, steganography methods [22, 26, 41, 61, 65] embed **secret data** into **cover data** to create **stega data** that incorporates hidden secret information. Receivers can extract previously hidden secret data from transmitted stega data.

However, these approaches are often tailored to a specific data format, such as images [22, 29], audio [10], or video [20, 40, 58] media. For each data type, current solutions [9, 22, 40, 65] have to design specific encoders and extractors to process the data, which lacks the flexibility to be universally applied across different data types. For example, the frameworks [22, 29] designed for image hiding cannot be used for audio or video steganography. This limitation poses a significant challenge for users who want to hide data across various media types without being restricted to specific data formats. Besides, existing methods heavily rely on external extractors to extract hidden information, which introduces critical security vulnerabilities. Attackers could exploit these extractors to corrupt [14, 66] or expose [34, 60] the secret data. Such inherent insecurity fundamentally undermines the reliability of conventional

*Corresponding author

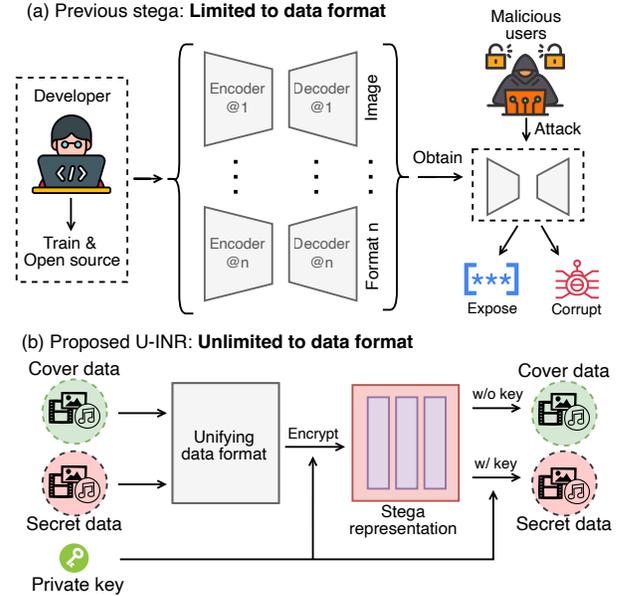


Figure 1: Illustration of our scenario. (a) Previous steganography approaches require designing specific frameworks for different data formats. Besides, malicious users could exploit the steganography encoder/decoder to expose or corrupt the secret data. (b) Our U-INR can work on various data formats like image, video and others. Besides, U-INR bypasses the need for external encoder/decoder architectures that could expose attack surfaces, ensuring exclusive access to key holders.

steganography in practical deployments, where robustness against malicious attacks is paramount.

We envision a new scenario where secret data can be embedded into cover data without data-type constraints. As demonstrated in Fig. 1, in our new scenario, the data hiding and extraction do not rely on format-specific encoders and extractors, preventing the limitations of specific data formats in previous steganography methods. The data senders and receivers use a private key that defines how secret data is embedded and extracted, eliminating the reliance and the risks of attacks on the external components. This scenario bypasses the need for external encoder/decoder architectures that could expose attack surfaces, ensuring exclusive access to key holders and preserving confidentiality and security.

We propose to overcome the challenges posed by various data formats through representing different types of data in a unified manner, thereby eliminating the need to manage different data formats. Our approach employs Implicit Neural Representations

(INRs) [51], which inherently satisfy this requirement by providing a unified framework for encoding diverse multimedia data (e.g., images, audio, videos and others). Besides, prior works [13, 16] have established that neural networks contain significant redundancy, with certain neurons being removable without degrading model performance. As INR naturally owns the network-based representation capability, it motivates us to use separate neurons in the INR to store the cover data and the hiding data, which enables seamless information embedding while preserving the perceptual quality of the cover media.

INR’s distinctive capability has inspired several works for data hiding, yet existing methods [7, 11, 33] still remain limited to single modalities, overlooking INRs’ across-modality representation capability [38, 51]. Recent INRSteg [53] suggests combining INRs for data steganography across different data modality. However, this approach explicitly modifies the default network structure of the INR, making it noticeable to attentive attackers. Besides, since the user must know the details of modifications, such a strategy is not conducive to the recipient’s retrieval. The above weakness underscores a necessity for a novel method that is less detectable by attentive attackers and a more appropriate way for receivers to retrieve the hidden data.

To address the above problems, we propose a novel strategy in which receivers and senders share a consensus. This consensus establishes a pre-defined agreement between sender and receiver that identifies specific neurons for encoding cover data versus those allocated for secret data embedding. Then, the sender and receivers can use such a consensus for data hiding and extraction, respectively. In our design, the identified neurons within the INR are utilized to represent the secret data, while the remaining neurons are optimized to encode the cover data. We refer to this optimized INR with the cover and secret data as a **stega representation**. During the extraction, this method minimizes detectability by vigilant attackers and simplifies the retrieval process for receivers, ensuring a more efficient and secure steganography.

We introduce an **implicit consensus mechanism**, which can effectively identify the positions of the parameters storing the secret data. In our design, the private shared key helps identify the positions used for hiding secret data. We achieve this by using this key to initialize the weights of the INR and then sorting and selecting the weights based on their values. The data sender and receiver can use this key to achieve data hiding and retrieval. Unauthorized users without the correct keys cannot obtain the secret data from the stega representation. We call this strategy **U-INR**, which allows users to conceal data within the parameters of INR and extract the hidden data using a designated key. By leveraging INR’s inherent flexibility and adaptability, our approach facilitates the seamless integration of hidden data into the existing data structure, enabling more secure communication and data transmission. In summary, our contributions can be summarized as:

- We introduce a novel steganography framework, unifying multimedia data representation for data hiding and retrieval.
- We develop a parameter-level embedding technique that integrates secret data directly within INRs, enabling cross-modal steganography without dedicated encoders or format-specific extractors.

- We introduce an implicit consensus mechanism to securely identify hidden data positions, enhancing data transmission safety against unauthorized access.

Extensive experiments have been conducted across various INR-based representations to demonstrate the generalizability and effectiveness of our method, resulting in an advanced improvement compared to existing steganography methods.

2 Related work

2.1 Traditional steganography

Steganography [50, 56, 60, 62–64] involves concealing secret data within a carrier medium, creating a covert information container [10, 39]. In image steganography, a cover image serves as the vessel for embedding a secret image [2]. Traditional approaches, such as spatial-based techniques [19, 21, 42, 45, 47], often employ strategies like Least Significant Bits (LSB), pixel value differencing (PVD) [45], and manipulation of multiple bit-planes [42] or color palettes [18, 19, 44]. However, these methods can introduce statistical anomalies that are detectable by steganalysis tools. To better prevent detection, adaptive strategies [25, 46] have been developed. These strategies focus on making the presence of secret data more invisible by minimizing embedding distortion and optimizing data coding to maintain visual indistinguishability. Similarly, transform-based methods [6, 23], including JSteg [47] and Discrete Cosine Transform (DCT) steganography [5, 17], have struggled to achieve substantial payload capacities. Recent advancements in steganographic techniques, leveraging the power of deep learning [22, 26, 40, 41, 61, 65], significantly enhance the capacity and security of data concealment. These methods employ neural networks [27, 28, 30, 36] to intricately analyze and subtly modify complex data attributes, resulting in more sophisticated and less detectable forms of data steganography. Baluja [2] pioneers a deep-learning approach capable of embedding a full-sized image within another. GAN [49] is used to create synthetic container images with probability map techniques for minimal distortion embedding [46, 55]. U-Net generators [59] integrated with adversarial frameworks have also been introduced, targeting distortion minimization [54]. Additionally, three-player game models such as SteganoGAN [61] and HiDDeN [65] employ auto-encoder architectures in an adversarial manner to enhance resistance to steganalysis. PUSNet [29] proposes extracting a steganography network from a common network, enabling the covert transmission of the steganography network. However, these methods overlook the potential of using neurons directly to represent secret data for steganography. The type of data limits these methods, as the corresponding pipeline must be designed according to each data type.

2.2 INR-based steganography

Implicit Neural Representation (INR) utilizes neural networks to learn continuous functions for data like images and shapes [38, 51]. It has advanced generative modeling, 3D reconstruction [31], and compression, demonstrating high-quality results from limited data [8, 12, 43, 51]. INR provides a groundbreaking approach to data representation by enabling continuous functions across multiple data modalities. Studies [4, 7, 11, 33, 53] explore the use of implicit neural representations (INR) for steganography. Existing works

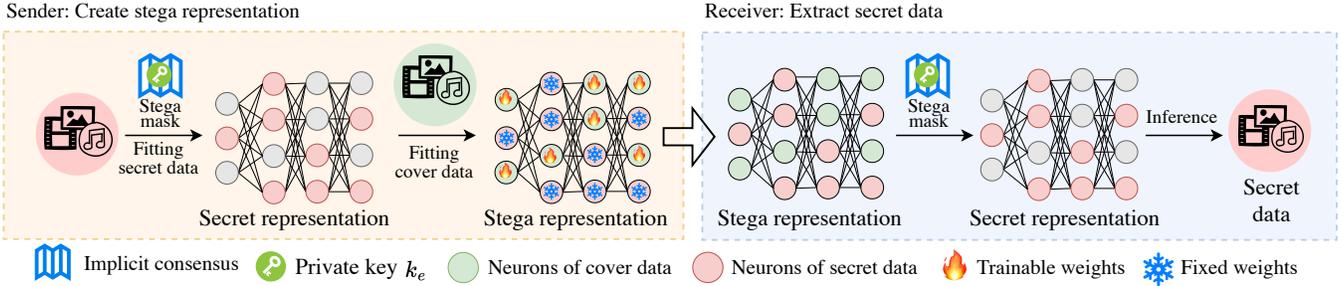


Figure 2: Framework of U-INR. Our architecture establishes secure synchronization between multimedia sender and receiver through an implicit consensus using a shared private key k_e . This key precisely maps the weight positions distinguishing secret payloads from cover data in the neural representation. Enforcing consensus-based parameter coordination through the implicit neural network’s weight-sharing mechanism eliminates the need for external auxiliary modules.

[35, 52] attempt INR-based steganography, they still treat INR like traditional data formats [39, 65], where secret data is extracted from the outputs of INR [38, 51]. StegaNeRV [4] uses an additional extractor to retrieve hidden information from the reconstructed results, ignoring the possibility of directly using neurons to represent the hidden information. Using partial neuron weights to represent the secret data is a straightforward idea [7, 11, 33, 53]. Methods like [7, 11, 33] address data steganography within a single modality, overlooking the capability of INR methods [51] to represent multiple data types. The recent INRSteg [53] proposes combining several different INRs to achieve steganography, but this changes the original network structure of the INR, making it easily noticeable by attentive attackers. Moreover, this concatenation method is not conducive to the recipient’s retrieval, as the user must know which neurons were used to store the hidden information. This highlights the need for a novel method that is more imperceptible to attentive attackers and easier for recipients to retrieve.

3 Problem Formulation

3.0.1 Traditional steganography. Traditional steganography frameworks are inherently constrained by data format dependencies, requiring specialized encoders and extractors for each media type. Given a cover data C (e.g., an image, audio, or video) and a secret data S , the objective is to generate stega data C^* using an embedding function tailored for the specific data format:

$$C^* = E(C, S). \quad (1)$$

Once the stega data C^* is received, the secret message S is extracted using a format-specific extractor:

$$S = D(C^*). \quad (2)$$

Traditional steganography methods are fundamentally limited by their dependence on specific data formats. Developing a format-specific framework requires significant effort as each media type (e.g., images, audio, video) has unique characteristics to implement steganography. This results in a lack of flexibility, preventing the seamless application of steganography techniques across different media types. Besides, the reliance on these specialized extractors introduces critical vulnerabilities. Attackers can target these extractors to access or corrupt the hidden. This reliance on D introduces critical vulnerabilities, as attackers can exploit these extractors to

access or corrupt the secret data [14, 34, 60, 66]. Each tailored extractor D represents a potential attack surface, increasing the risk of security breaches. Consequently, the robustness of traditional steganography methods is compromised, particularly in environments where security against malicious attacks is paramount.

3.0.2 Our proposed scenario. We propose a unified steganography paradigm via Implicit Neural Representations (INRs) to transform multimedia data from various modalities into a unified representation. Our approach operates directly on the INR’s parameter space, eliminating the need for format-specific operations.

Consider an INR network parameterized by Θ , representing the cover data C . The secret message S is embedded directly within the parameters of Θ :

$$\Theta^* = \Theta(C, S, k_e), \quad (3)$$

where k_e is a private key identifying the positions within the parameter space where the secret data is stored. The stega data C^* is represented by the INR network Θ^* :

$$C^* = \Theta^*(\cdot). \quad (4)$$

The secret data S could be retrieved using the extraction function from the stega INR Θ^* :

$$S = \Theta^*(k_e). \quad (5)$$

The private key k_e ensures that only authorized users can access the secret message, thereby enhancing security.

Our framework takes advantage of the unique representation capability of INR, offering a unified and efficient solution for steganography across multiple media types. As the need for external extractors is eliminated, we also address the critical vulnerability [14, 34, 60, 66] in traditional steganography.

4 Proposed method

The overview framework of U-INR is depicted in Fig. 2. We utilize Implicit Neural Representation (INR) to transform multimedia data from various modalities into a unified representation. This enables a unified steganography framework applicable to diverse data types. Then, instead of depending on external modules for hiding and extraction, we introduce an implicit consensus mechanism between data senders and receivers. This mechanism supports both data hiding and extraction, mitigating risks associated with external components.

4.1 Unified multimedia data representation

Traditional steganography methods are inherently constrained to multimedia data types due to their reliance on format-specific frameworks. These approaches struggle to generalize across modalities as format-specific frameworks cannot harmonize heterogeneous data. In contrast, our work leverages Implicit Neural Representations (INRs) [38, 51] to dissolve such weakness. We propose encoding diverse multimedia data into a unified neural representation, adapting to arbitrary data types like images, audio, and 3D scenes seamlessly.

INR employs continuous functions representing various data formats, such as images, videos, audio, and 3D scenes. An INR inputs spatial or temporal coordinates and outputs the corresponding information at the coordinates, such as image pixel values or amplitude for audio. Generally, an INR F_{Θ} can be denoted as:

$$\mathbf{y} = F_{\Theta}(\mathbf{u}), \quad (6)$$

where \mathbf{u} represents the input coordinates, which can be either spatial (for images and 3D scenes) or temporal (for audio), θ denotes the learned parameters of the neural network, and \mathbf{y} is the output information such as pixel or amplitude values. This framework allows a unified approach to modeling different data types using a flexible neural network-based function F_{Θ} . For instance, in the context of image representation, an INR would receive a 2D coordinate $\mathbf{u} \in \mathbb{R}^2$ and output a pixel value $\mathbf{c} \in \mathbb{R}^3$. Similarly, for audio signals, a one-dimensional temporal coordinate $t \in \mathbb{R}$ would be mapped to an audio amplitude $\mathbf{a} \in \mathbb{R}$. INR has already demonstrated strong representational capabilities across various modalities of data. Given these unique advantages, we propose leveraging INR to unify multimedia data format. Then, the data sender and receiver are relieved from directly handling the format of steganographic data. They only need to determine the position of the neurons for secret data, thereby alleviating the threat caused by external encoders/extractors.

4.2 Implicit consensus mechanism

The primary motivation behind our implicit consensus stems from a critical vulnerability in traditional steganographic approaches: their dependence on external explicit extractors [29, 65]. These external explicit extractors present a significant security weakness—they serve as a clear attack surface that adversaries can exploit to detect, corrupt, or expose hidden information [14, 34, 60, 66]. We alleviate such vulnerability by designing an implicit data hiding and extraction mechanism. Our approach lets the sender and receiver cryptographically agree on which parameters contain hidden data using only a pre-shared private key k_e without requiring any external extraction mechanism during transmission. This shift from explicit extractors to implicit consensus significantly strengthens security by removing a primary attack vector. Moreover, this approach inherently supports format-agnostic steganography, as the implicit mechanism operates on neural parameters regardless of the underlying data modality, addressing both the security and flexibility limitations that have constrained traditional steganographic methods. The whole process for making the implicit consensus is as follows.

4.2.1 Obtaining private key. In this work, the private key replaces the external extraction mechanism during the data distribution.

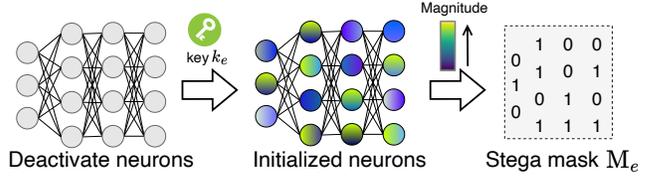


Figure 3: Implicit Consensus. The initialized weight values of the INR are used to identify and select the weights for secret data based on the magnitudes.

This private key enables both the hiding and extraction of data within our framework. Specifically, the private key determines the locations for data hiding. Prior to distribution, data senders utilize the private key to generate the stega representation. Subsequently, the sender uses a designated method to securely share the private key with the recipients. Upon receiving the steganographic data, the recipients apply the pre-shared private key to retrieve the hidden information. In our setup, we use a pre-arranged **private key** k_e . Rather than explicitly transmitting the secret data’s position along with the cover representation, users can use pre-shared Arabic numerals as the private key k_e , and only receivers with the correct key can access the secret data.

4.2.2 Creating stega mask M_e . With the private key k_e , we can create a stega mask M_e by determining the position of the secret data. The stega mask maintains a binary matrix corresponding to neuron positions, enabling selective neural modulation for secure data embedding. The stega mask M_e can be used to extract the secret data as it records the position of the secret data. Compared to the previous methods [7, 11, 33], our approach allows receivers to directly obtain the location of secret data based on the pre-arranged key k_e , thereby avoiding potential risks during data transmission. Once the stega data is received, these positions can be retrievable to ensure the receiver can regain access to the secret data. This way, the sender and receiver can reliably identify and extract the secret information embedded within the model’s parameters using the shared key.

With the private key k_e , the data sender could effectively determine the position of secret data via our proposed **implicit consensus** (Fig. 3). The algorithm details of implicit consensus are depicted in algorithm 1. For an implicit neural network $\mathbb{N}[W](\cdot)$, where $\mathbb{N}[\cdot]$ and W denote the architecture and weights respectively, we have the secret representation $\mathbb{N}[W \odot M_e](\cdot)$. Here, \odot is the element-wise product, and M_e is a stega mask indicating secret representation positions. We leverage initialized weights from the seed key k_e . This method selects significant weights by magnitude, preserving network performance. Given a stega ratio S and total weight count N , the algorithm initializes weights W_e using $\mathbb{N}[\cdot]$ and k_e , ensuring reproducibility. Weights are sorted by absolute value to identify significance. A threshold t_S is set as the p -th largest weight, where $p = \lfloor S \cdot N \rfloor$ and $\lfloor \cdot \rfloor$ is floor function, selecting the top $S \cdot 100\%$ weights. The stega mask M_e is generated by marking weights exceeding t_S as 1, others as 0. This binary mask highlights significant weights, encoding the secret representation effectively.

This approach serves multiple purposes. First, it ensures that only the most impactful weights are retained, which helps maintain

Algorithm 1 Creating stega mask M_e via implicit consensus

Require: Neural network architecture $\mathbb{N}[\cdot]$, seed key k_e , stega ratio \mathcal{S} , total weight count \mathcal{N} , initialization function \mathcal{I}

Ensure: Stega mask M_e

- 1: **Initialize Weights:**
 - 2: Initialize weights W_e using $\mathcal{I}(\mathbb{N}[\cdot], k_e)$
 - 3: **Sort Weights:**
 - 4: Sort the weights W_e in descending order based on their absolute values.
 - 5: **Determine Threshold:**
 - 6: Calculate $p = \lfloor \mathcal{S} \cdot \mathcal{N} \rfloor$
 - 7: Identify the threshold $t_{\mathcal{S}}$ as the value of the p -th largest weight in W_e
 - 8: **Generate Stega Mask:**
 - 9: **for** each weight w_i in W_e **do**
 - 10: **if** $|w_i| > t_{\mathcal{S}}$ **then**
 - 11: Set $M_e[i] = 1$
 - 12: **else**
 - 13: Set $M_e[i] = 0$
 - 14: **end if**
 - 15: **end for**
 - 16: **Output the Stega Mask:**
 - 17: **return** M_e
-

the model’s performance while embedding the secret representation. Second, by focusing on weight magnitude, we avoid the pitfalls of selecting arbitrary weights, which could lead to sub-optimal performance or even model failure. Lastly, this method provides a systematic and reproducible way to generate the stega mask, making it easier to maintain consistency across different model iterations and experiments. Overall, the implicit mechanism enhances the robustness and reliability of the U-INR framework.

4.3 Encrypting stega representation

After obtaining the stega mask M_e , users can construct the stega representation using secret and cover data. To obtain the secret representation $\mathbb{N}[W \odot M_e]$, we first optimize the representation for the secret data based on the stega mask M_e . Next, we fix these optimized secret weights and optimize the remaining weights $\mathbb{N}[W \odot \bar{M}_e]$ for the secret data to create the final stage representation. The following sections provide a detailed illustration of this process.

4.3.1 Fitting secret data. Our weight selection strategy gives a choice to obtain the stega mask M_e based on the private key k_e . Given secret data (e.g., image, video, or other types), its information values y_{se} and corresponding coordinate \mathbf{u} , we can build the secret representation by

$$\hat{y}_{se} = \mathbb{N}[W \odot M_e](\mathbf{u}), \quad (7)$$

where \hat{y}_{se} is the predicted secret data, and the secret weights determined by stega mask M_e are optimized via eq. (9).

4.3.2 Fitting cover data. After optimizing the weights for the secret data, we fix those weights and update the remaining weights to store the cover data. The process can be denoted as

$$\hat{y}_{st} = \mathbb{N}[W_{\text{fix}} \odot M_e \cup W \odot \bar{M}_e](\mathbf{u}), \quad (8)$$

where \bar{M}_e is a binary mask complementing M_e , and y_{st} refers to the stega data at corresponding coordinate \mathbf{u} .

The above process can be achieved via standard INR procedure [38, 51], which can be formulate as

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^N (y_{gt}^{(i)} - \hat{y}^{(i)})^2, \quad (9)$$

where N is the number of samples, $y_{gt}^{(i)}$ is the ground truth value for the i -th sample, and $\hat{y}^{(i)}$ is the predicted value for the i -th sample. When the optimization settles down, we can get the stega representation with both cover and secret data. The implementation details of each data modality are provided in section 7.1.

4.4 Decrypting secret representation

After obtaining the stega representation, the ordinary users without the correct key can only obtain the whole data stored in the stega representation via standard INR inference eq. (6) as:

$$\mathbb{N}[W](\mathbf{u}) \rightarrow \hat{y}_{st}, \quad (10)$$

where \hat{y}_{st} is the output values of stega data at coordinates \mathbf{u} . Ordinary users can only obtain stega data through standard inference as secret data is stored in particular INR neurons. Even if attackers suspect hidden data exists, searching the high-dimensional neuron space becomes computationally prohibitive. Users with the correct key k_e can regain the stega mask M_e locating the weights for secret data via our implicit consensus (section 4.2). Thus, the secret data can be obtained through:

$$\mathbb{N}[W \odot M_e](\mathbf{u}) \rightarrow \hat{y}_{se}, \quad (11)$$

where $W \odot M_e$ denotes the weights for secret representation. This M_e can be obtained with key k_e via the **implicit consensus mechanism** strategy in section 4.2.

5 Experiments

5.1 Experimental settings

5.1.1 Evaluation. To demonstrate the effectiveness of our method, we conduct experiments on different types of data, including images, videos, audio, and 3D scenes. For images, following established work [29], we evaluate the performance on three dataset, including DIV2K [1], 1,000 images randomly selected from ImageNet [48], and COCO [32] dataset. For 3D scenes, we experiment with 4 classic scenes from LLFF [37] and NeRF-blender [38], respectively. We select the original test set in SIREN [51] for audio and video. We adopt PSNR, SSIM [57], Averaged Pixel-wise Discrepancy (APD), and RMSE to measure the visual quality. Higher PSNR and SSIM values indicate better image embedding and recovery performance. Lower RMSE and APD values suggest improved performance in these tasks. We adopt the MSE mean and MSE standard deviation for audio data to evaluate the quality of audio representation. We test the representation quality to evaluate the impact of the stega ratio \mathcal{S} on the performance of the secret and stega representation under different ratios. Besides this, we also test the robustness of our method against network pruning [13, 24] to analyze potential threats.

Table 1: Performance comparisons on different datasets. “↑”: the larger the better, “↓”: the smaller the better. Stega ratio S denotes the ratio of parameters used to represent secret data. The results of our U-INR are highlighted in cyan.

Methods	Cover/Stega-image pair											
	DIV2K				COCO				ImageNet			
	PSNR(dB)↑	SSIM↑	APD↓	RMSE↓	PSNR(dB)↑	SSIM↑	APD↓	RMSE↓	PSNR(dB)↑	SSIM↑	APD↓	RMSE↓
HiDDeN [65]	28.19	0.9287	8.01	11.00	29.16	0.9318	6.91	9.60	28.87	0.9234	7.43	10.21
Baluja [3]	28.42	0.9347	7.92	10.64	29.32	0.9374	7.04	9.36	28.82	0.9303	7.68	10.21
HiNet [22]	44.86	0.9922	1.00	1.53	46.47	0.9925	0.81	1.30	46.88	0.9920	0.81	1.26
PUSNet [29]	38.15	0.9792	2.30	3.33	39.09	0.9772	2.01	2.96	38.94	0.9756	2.21	3.06
U-INR ($S = 0.3$)	38.32	0.9890	2.17	2.72	39.70	0.9889	1.55	2.39	39.06	0.9813	1.61	2.45
U-INR ($S = 0.5$)	35.15	0.9740	3.59	4.70	35.50	0.9737	3.55	3.65	35.32	0.9770	2.51	3.63

Methods	Secret/Recovered image pair											
	DIV2K				COCO				ImageNet			
	PSNR(dB)↑	SSIM↑	APD↓	RMSE↓	PSNR(dB)↑	SSIM↑	APD↓	RMSE↓	PSNR(dB)↑	SSIM↑	APD↓	RMSE↓
HiDDeN [65]	28.42	0.8695	7.62	9.94	28.81	0.8576	7.20	9.54	28.23	0.8435	7.83	10.47
Baluja [3]	28.53	0.9036	7.53	10.66	29.13	0.9091	6.61	9.80	27.63	0.8909	8.33	12.26
HiNet [22]	28.66	0.8507	7.25	9.68	28.08	0.8181	7.80	10.49	27.94	0.8159	8.03	10.83
PUSNet [29]	26.88	0.8363	8.75	11.95	26.96	0.8211	8.71	12.14	26.28	0.8028	9.58	13.43
U-INR ($S = 0.3$)	34.50	0.9727	3.82	5.26	34.92	0.9802	3.53	4.86	34.03	0.9701	3.97	5.53
U-INR ($S = 0.5$)	37.11	0.9851	2.84	3.86	37.53	0.9873	2.73	3.41	36.82	0.9802	2.90	4.12

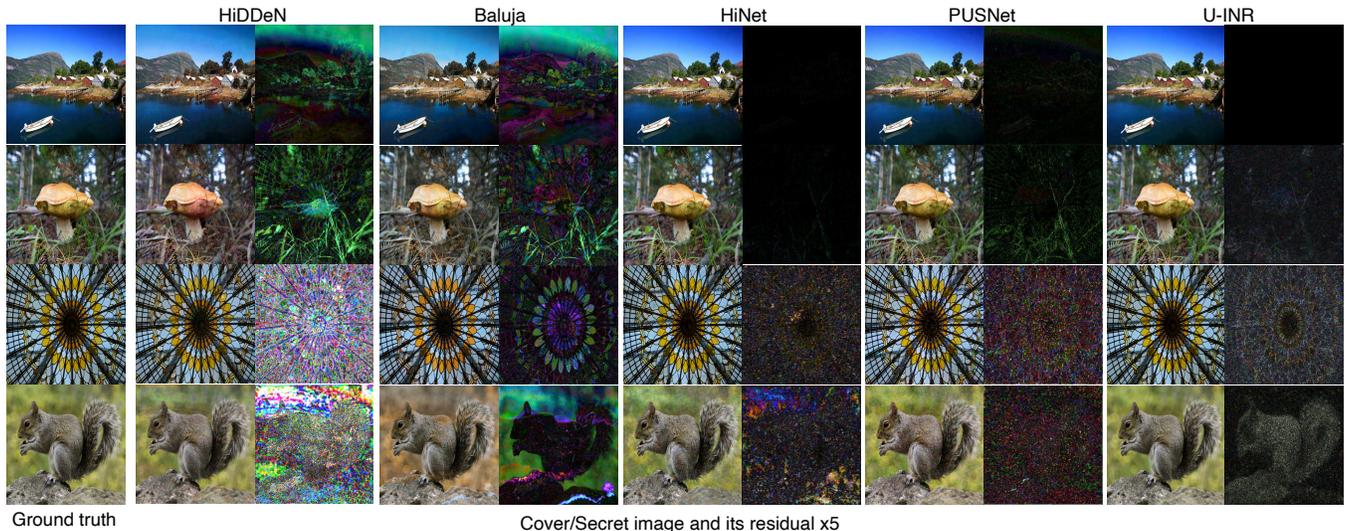


Figure 4: Examples of the stega and recovered images generated using different schemes. The left is the original image, and the right represents $\times 5$ magnified residuals. The cover/stega and secret/recovered images are given in the first and last rows. For our U-INR, we use stega ratio $S = 0.3$ as it balances the quality of cover and secret representation.

5.1.2 Benchmarks. To evaluate the performance of our method, we compare our U-INR against existing DNN-based steganographic methods, including HiDDeN [65], Baluja [3], HiNet [22], and PUSNet [29]. We implement the aforementioned models on the DIV2K training dataset for fair comparisons and evaluate their performance under the same settings [29]. To illustrate the generalizability of our U-INR, we evaluate our U-INR on other types of data, including audio, video, 3D scene, and signed distance function.

5.2 Results on different data types

5.2.1 Results on 2D image. To compare the capabilities of our method in steganography, we evaluate our U-INR and other image steganographic methods on various 2D image datasets [1, 32, 48]. In Table 1, we provide two sets of experimental results with different stega ratios $S = \{0.3, 0.5\}$, denoting the ratios of the parameters for secret representation. Although these steganography methods [2, 22, 29, 65] are specifically designed for 2D images, our U-INR achieves a better quality of the recovered secret image

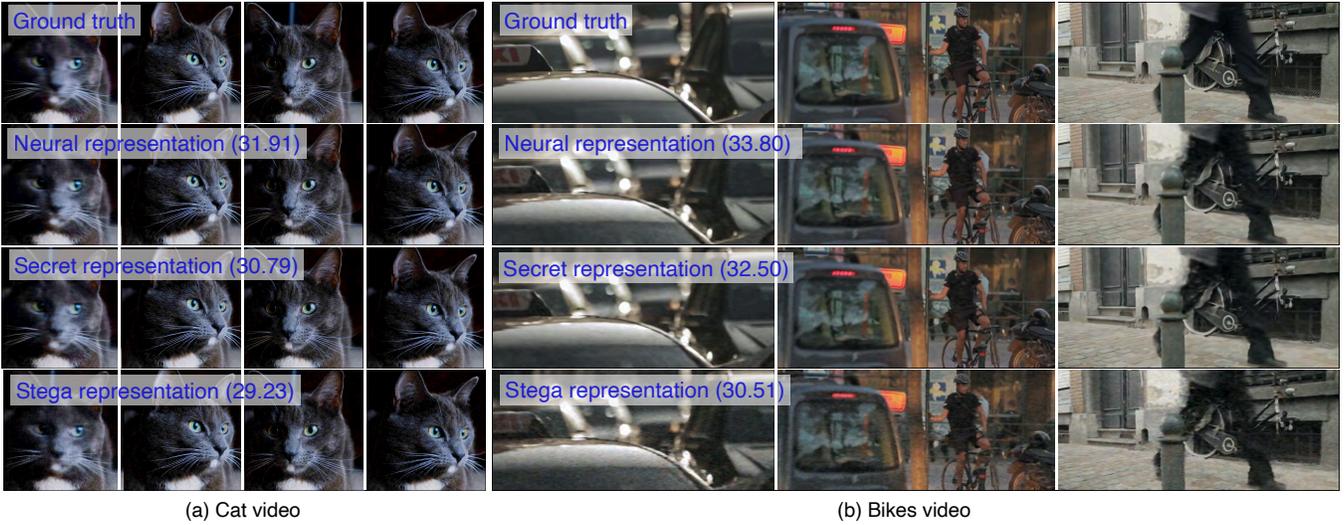


Figure 5: Quantitative and qualitative results of our method when applying to video data. The bike video and cat video are adopted as secret and stega representations. Compared to the normal neural representation, the quality of the stega and secret representations only experiences a slight decrease.

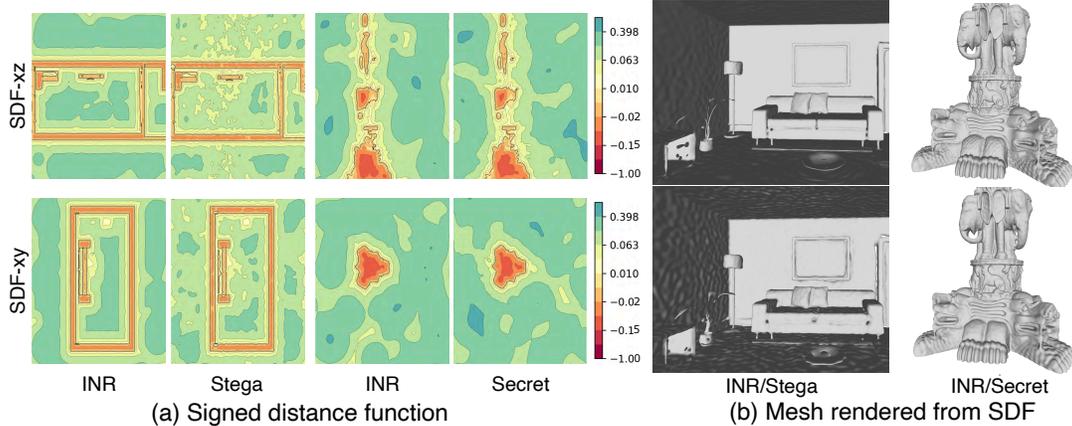


Figure 6: A case study on Signed distance function. Thai statue (Secret) is extracted from the representation of Room (Stega).

while keeping comparable results in the quality of the stega image. Aligning with the visual samples in Fig. 4, our U-INR achieves higher capacity steganography with a less obvious residual map. We successfully embed secret image representations into stega representations within the INR parameters while preserving the original image quality.

5.2.2 *Results on video data.* To demonstrate the generalizability of our approach, we extend our U-INR to video data and present the quantitative and qualitative results in Fig. 5. We implement our U-INR on Cat and Bike videos from SiREN [51]. Comparing data as secret/stega representation with standard neural representation, our method minimally impacts quality. Our results showcase the ability to conceal and retrieve the hidden video content while maintaining the visual fidelity of the original video. Users can obtain hidden high-quality videos while ensuring invisibility.

5.2.3 *Results on SDF data.* We extend our U-INR to Signed Distance Function (SDF) data and present the results in Fig. 6. We apply

our U-INR to the signed distance function and embed secret SDF representations within the network parameters. Our experimental results demonstrate the capability to conceal and retrieve the hidden SDF content while maintaining the geometric fidelity of the original structures. Users can obtain hidden high-quality SDF data while ensuring invisibility.

5.2.4 *Results on 3D scene.* We implement our U-INR on 3D scene representations with Neural Radiance Fields (NeRFs) [38]. We train the 3D presentation with multi-view images and embed secret 3D data within the parameters of NeRF models. Quantitative and qualitative results for each scene when used as stega and secret data are reported in Table 2 and Fig. A.5. Compared to the original implicit neural representation, the quality of the stega and secret representations slightly declines, representing the 3D scene effectively. Our experiments demonstrate that our approach can effectively conceal and recover the hidden 3D content while preserving the high-quality rendering of the original 3D scenes. This capability

Table 2: We implement our U-INR on 3D scenes with neural radiance field (NeRF) [38]. We report the metric results when the scenes are adopted as stega representation and secret representation. The visualized results are presented in the appendix.

		PSNR(dB) \uparrow	SSIM \uparrow	LPIPS \downarrow
Blender	Lego	26.30	0.9356	0.1256
	Lego (Secret)	25.92	0.9284	0.1400
	Lego (Stega)	25.62	0.9182	0.1511
	Hotdog	32.86	0.9778	0.0671
	Hotdog (Secret)	32.45	0.9751	0.0782
	Hotdog (Stega)	32.40	0.9733	0.0830
LLFF	Flower	27.50	0.8543	0.1641
	Flower (Secret)	26.62	0.8202	0.2050
	Flower (Stega)	26.60	0.8146	0.2270
	Room	30.64	0.9313	0.1674
	Room (Secret)	29.92	0.9160	0.2059
	Room (Stega)	29.74	0.9109	0.2265

opens up new avenues for secure data transmission and information hiding in the context of steganography 3D models.

5.2.5 Results on audio data. To showcase the versatility of our approach, we extend our U-INR to audio data and present the results in Table 3. The visualized results are presented in the appendix. We implement our U-INR on Bach and Counting audio [51], embedding secret audio messages within the network parameters. The stega representations yielded enhanced performance, potentially attributable to the inherently lower complexity of audio data relative to other data modalities. Our results demonstrate the ability to conceal and retrieve the hidden audio content while preserving the auditory fidelity of the original signals. Users can obtain hidden high-quality audio while ensuring invisibility.

5.3 Impacts of stega ratio

Stega ratio \mathcal{S} indicates the percentage of INR parameters used for secret representation. Thus, selecting the right ratio depends on whether the priority is to preserve “secret” information or maintain broad coverage. We evaluate the impacts of selecting a suitable ratio for U-INR to identify the suitable ratio that trades off the secret and stega representation quality. We experiment across a range of sparsity levels from 10% to 90% on DIV2K test-set [1]. As shown in Fig. 7, a higher ratio will decrease stega quality and improve secret representation quality. The experimental results reveal a trade-off when adjusting the sparsity ratios, indicating the trade-off choice in the intermediate range of 30% to 70%.

5.4 Threaten analysis

Pruning techniques are commonly used to compress and speed up neural networks by removing redundant parameters, which may impact the integrity of secret data. To test the resilience of our U-INR method against potential threats, we simulate pruning attacks on INR networks with stega representation. We evaluate the robustness of the stega representation against two pruning strategies, including magnitude-based pruning [13] and random pruning [24]. For instance, we progressively prune increasing percentages of the INR parameters for each method and measure the deterioration.

Table 3: The audios Bach and Counting [51] are representation with INRs. The Bach and Counting are adopted as secret and stega representations, respectively. The experimental results are evaluated 10 times.

	MSE Mean \downarrow	MSE Standard Dev. \downarrow
Bach	1.980×10^{-4}	5.527×10^{-4}
Bach (Secret)	2.387×10^{-4}	6.101×10^{-4}
Bach (Stega)	2.236×10^{-5}	6.024×10^{-5}
Counting	8.834×10^{-4}	5.374×10^{-3}
Counting (Secret)	9.718×10^{-4}	5.594×10^{-3}
Counting (Stega)	5.728×10^{-4}	3.915×10^{-3}

Table 4: Results of pruning weights in INR. The secret* representation has not been erased by the magnitude-based pruning method, as its weights have larger weight values than the stega representation. We further presented the analysis in the appendix.

Pruning strategy		0%	1%	5%	10%	20%
Random	INR	38.48	35.39	30.61	28.06	22.06
	Stega	33.46	30.24	25.83	23.46	17.85
	Secret	35.08	32.31	28.97	26.86	21.05
Magnitude	INR	38.48	37.99	35.42	29.63	22.45
	Stega	33.46	33.43	30.87	25.07	18.19
	Secret*	35.08	33.86	33.86	33.86	33.86

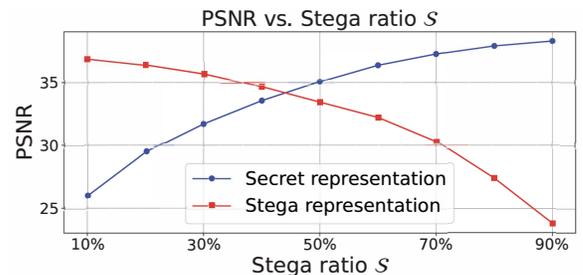


Figure 7: Representation performance under different stega ratios.

As shown in Table 4, with higher attack strength, the quality of the representation declines. However, the secret representation is more robust against both pruning strategies than the stega representation, indicating the resilience of the method against advanced adversarial attacks.

6 Conclusion

This paper presents U-INR, a novel unified steganographic method leveraging implicit neural representations (INRs) to embed covert data across diverse media types. Our approach uniquely incorporates secret information within a portion parameter of the INR network, facilitating secure and discreet data transmission. Comprehensive experiments across multiple data modalities demonstrate U-INR’s exceptional capacity for embedding information securely while preserving INR fidelity. These results underscore the method’s robustness and its potential for broad applications in enhancing data security and privacy.

References

- [1] Eirikur Agustsson and Radu Timofte. 2017. NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study. In *CVPRW*.
- [2] Shumeet Baluja. 2017. Hiding images in plain sight: Deep steganography. In *NeurIPS*.
- [3] Shumeet Baluja. 2020. Hiding Images within Images. *TPAMI* (2020).
- [4] Monsij Biswal, Tong Shao, Kenneth Rose, Peng Yin, and Sean Mccarthy. 2024. StegaNeRV: Video Steganography using Implicit Neural Representation. In *CVPR*.
- [5] Alexia Briassouli, Panagiotis Tsakalides, and Athanasios Stouraitis. 2005. Hidden messages in heavy-tails: DCT-domain watermark detection using alpha-stable models. *TMM* (2005).
- [6] Yambem Jina Chanu, Kh Manglem Singh, and Themrichon Tuithung. 2012. Image steganography and steganalysis: A survey. *IJCA* (2012).
- [7] Lifeng Chen, Jia Liu, Wenquan Sun, Weina Dong, and Fuqiang Di. 2024. NeRF in NeRF: An Implicit Representation Watermark Algorithm for NeRF. *Preprint* (2024).
- [8] Zhiqin Chen and Hao Zhang. 2019. Learning implicit fields for generative shape modeling. In *CVPR*.
- [9] Nedeljko Cvejk and Tapio Seppanen. 2002. Increasing the capacity of LSB-based audio steganography. In *IEEE MMSP*.
- [10] Fatiha Djebbar, Beghdad Ayad, Karim Abed Meraim, and Habib Hamam. 2012. Comparative study of digital audio steganography techniques. *Eurasip. J. Audio Spee.* (2012).
- [11] Weina Dong, Jia Liu, Lifeng Chen, Wenquan Sun, Xiaozhong Pan, and Yan Ke. 2024. Implicit Neural Representation Steganography by Neuron Pruning. *Preprint* (2024).
- [12] Emilien Dupont, Adam Goliński, Milad Alizadeh, Yee Whye Teh, and Arnaud Doucet. 2021. Coin: Compression with implicit neural representations. In *ICLRW*.
- [13] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M Roy, and Michael Carbin. 2020. Pruning neural networks at initialization: Why are we missing the mark?. In *ICLR*.
- [14] S Geetha, S Subburam, S Selvakumar, Seifedine Kadry, and Robertas Damasevicius. 2021. Steganogram removal using multidirectional diffusion in fourier domain while preserving perceptual image quality. *Pattern. Recogn. Lett.* (2021).
- [15] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*.
- [16] Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. *NeurIPS* (2015).
- [17] Stefan Hetzl and Petra Mutzel. 2005. A graph-theoretic approach to steganography. In *CMS*.
- [18] Dongdong Hou, Weiming Zhang, Kejiang Chen, Sian-Jheng Lin, and Nenghai Yu. 2018. Reversible data hiding in color image with grayscale invariance. *TCSVT* (2018).
- [19] Shoko Imaizumi and Kei Ozawa. 2014. Multibit embedding algorithm for steganography of palette-based images. In *PSIVT*.
- [20] Jun Jia, Zhongpai Gao, Dandan Zhu, Xiongkuo Min, Menghan Hu, and Guangtao Zhai. 2022. RIVIE: Robust inherent video information embedding. *TMM* (2022).
- [21] Yue Jiang, Kejiang Chen, Wei Yan, Xuehu Yan, Guozheng Yang, and Kai Zeng. 2024. Robust Secret Image Sharing Resistant to JPEG Recompression Based on Stable Block Condition. *TMM* (2024).
- [22] Junpeng Jing, Xin Deng, Mai Xu, Jianyi Wang, and Zhenyu Guan. 2021. HiNet: Deep Image Hiding by Invertible Network. In *CVPR*.
- [23] Inas Jawad Kadhim, Prashan Premaratne, Peter James Vial, and Brendan Halloran. 2019. Comprehensive survey of image steganography: Techniques, Evaluations, and trends in future research. *Neurocomputing* (2019).
- [24] Jaeho Lee, Jihoon Tack, Namhoon Lee, and Jinwoo Shin. 2021. Meta-learning sparse implicit neural representations. In *NeurIPS*.
- [25] Bin Li, Ming Wang, Jiwu Huang, and Xiaolong Li. 2014. A new cost function for spatial image steganography. In *ICIP*.
- [26] Chenxin Li, Brandon Y Feng, Zhiwen Fan, Panwang Pan, and Zhangyang Wang. 2023. StegaNeRF: Embedding invisible information within neural radiance fields. In *JCCV*.
- [27] Guobiao Li, Sheng Li, Meiling Li, Zhenxing Qian, and Xinpeng Zhang. 2023. Towards deep network steganography: from networks to networks. *arXiv preprint arXiv:2307.03444* (2023).
- [28] Guobiao Li, Sheng Li, Meiling Li, Xinpeng Zhang, and Zhenxing Qian. 2023. Steganography of steganographic networks. In *AAAI*.
- [29] Guobiao Li, Sheng Li, Zicong Luo, Zhenxing Qian, and Xinpeng Zhang. 2024. Purified and Unified Steganographic Network. In *CVPR*.
- [30] Guobiao Li, Sheng Li, Zhenxing Qian, and Xinpeng Zhang. 2024. Cover-separable Fixed Neural Network Steganography via Deep Generative Models. In *ACM MM*.
- [31] Ruiqi Li and Yiu-ming Cheung. 2024. Variational Multi-scale Representation for Estimating Uncertainty in 3D Gaussian Splatting. *NeurIPS* (2024).
- [32] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *ECCV*.
- [33] Jia Liu, Peng Luo, and Yan Ke. 2023. Hiding functions within functions: Steganography by implicit neural representations. *arXiv preprint arXiv:2312.04743* (2023).
- [34] Qin Liu, Jiamin Yang, Hongbo Jiang, Jie Wu, Tao Peng, Tian Wang, and Guojun Wang. 2022. When deep learning meets Steganography: Protecting inference privacy in the dark. In *INFOCOM*.
- [35] Ziyuan Luo, Qing Guo, Ka Chun Cheung, Simon See, and Renjie Wan. 2023. CopyrNeRF: Protecting the copyright of neural radiance fields. In *ICCV*.
- [36] Zicong Luo, Sheng Li, Guobiao Li, Zhenxing Qian, and Xinpeng Zhang. 2023. Securing Fixed Neural Network Steganography. In *ACM MM*.
- [37] Ben Mildenhall, Pratul P. Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. 2019. Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines. *TOG* (2019).
- [38] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*.
- [39] Tayana Morkel, Jan HP Eloff, and Martin S Olivier. 2005. An overview of image steganography. In *ISSA*.
- [40] Chong Mou, Youmin Xu, Jiechong Song, Chen Zhao, Bernard Ghanem, and Jian Zhang. 2023. Large-capacity and flexible video steganography via invertible neural network. In *CVPR*.
- [41] Ramadhan J Mstafa, Younis Mohammed Younis, Haval Ismael Hussein, and Muhsin Atto. 2020. A new video steganography scheme based on Shi-Tomasi corner detector. *IEEE Access* (2020).
- [42] Bui Cong Nguyen, Sang Moon Yoon, and Heung-Kyu Lee. 2006. Multi bit plane image steganography. In *IWDW*.
- [43] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. 2019. Occupancy flow: 4D reconstruction by learning particle dynamics. In *ICCV*.
- [44] Michiharu Niimi, Hideki Noda, Eiji Kawaguchi, and Richard O Eason. 2002. High capacity and secure digital steganography to palette-based images. In *ICIP*.
- [45] Feng Pan, Jun Li, and Xiaoyuan Yang. 2011. Image steganography method based on PVD and modulus function. In *ICECC*.
- [46] Tomáš Pevný, Tomáš Filler, and Patrick Bas. 2010. Using high-dimensional image models to perform highly undetectable steganography. In *IH*.
- [47] Niels Provos and Peter Honeyman. 2003. Hide and seek: An introduction to steganography. *IEEE S & P* (2003).
- [48] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. 2015. Imagenet large scale visual recognition challenge. *IJCV* (2015).
- [49] Haichao Shi, Jing Dong, Wei Wang, Yinlong Qian, and Xiaoyu Zhang. 2018. SSGAN: Secure steganography based on generative adversarial networks. In *PCM*.
- [50] Yun-Qing Shi, Xiaolong Li, Xinpeng Zhang, Hao-Tian Wu, and Bin Ma. 2016. Reversible data hiding: Advances in the past two decades. *IEEE access* (2016).
- [51] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. 2020. Implicit neural representations with periodic activation functions. In *NeurIPS*.
- [52] Qi Song, Ziyuan Luo, Ka Chun Cheung, Simon See, and Renjie Wan. 2024. Protecting NeRFs' Copyright via Plug-And-Play Watermarking Base Model. In *ECCV*.
- [53] Sojeong Song, Seoyun Yang, Chang D Yoo, and Junmo Kim. 2024. Implicit Steganography Beyond the Constraints of Modality. In *ECCV*.
- [54] Weixuan Tang, Bin Li, Shunquan Tan, Mauro Barni, and Jiwu Huang. 2019. CNN-based adversarial embedding for image steganography. *TIFS* (2019).
- [55] Weixuan Tang, Shunquan Tan, Bin Li, and Jiwu Huang. 2017. Automatic steganographic distortion learning using a generative adversarial network. *Singal Proc. Let.* (2017).
- [56] Jinyuan Tao, Sheng Li, Xinpeng Zhang, and Zichi Wang. 2018. Towards robust image steganography. *TCSVT* (2018).
- [57] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *TIP* (2004).
- [58] Xinyu Weng, Yongzhi Li, Lu Chi, and Yadong Mu. 2019. High-capacity convolutional video steganography with temporal residual modeling. In *ICMR*.
- [59] Jianhua Yang, Danyang Ruan, Jiwu Huang, Xiangui Kang, and Yun-Qing Shi. 2019. An embedding cost learning framework using GAN. *TIFS* (2019).
- [60] Zijin Yang, Kejiang Chen, Kai Zeng, Weiming Zhang, and Nenghai Yu. 2023. Provably secure robust image steganography. *TMM* (2023).
- [61] Kevin Alex Zhang, Alfredo Cuesta-Infante, Lei Xu, and Kalyan Veeramachaneni. 2019. SteganoGAN: High capacity image steganography with GANs. *arXiv preprint arXiv:1901.03892* (2019).
- [62] Xinpeng Zhang. 2011. Reversible data hiding in encrypted image. *IEEE Sign. process. letters* (2011).
- [63] Xiang Zhang, Fei Peng, and Min Long. 2018. Robust coverless image steganography based on DCT and LDA topic classification. *TMM* (2018).
- [64] Xinpeng Zhang and Shuozhong Wang. 2006. Efficient steganographic embedding by exploiting modification direction. *IEEE Comm. letters* (2006).
- [65] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. 2018. Hidden: Hiding data with deep networks. In *ECCV*.

[66] Zhiying Zhu, Sheng Li, Zhenxing Qian, and Xinpeng Zhang. 2021. Destroying robust steganography in online social networks. *Inform. Sciences* (2021).

7 Appendix

7.1 Implementation details

In our experiment, we implement SIREN¹ [51] for image, video, audio, SDF data. We implement NeRF² [38] for 3D scenes. All experiments are conducted on Ubuntu 18.04 with one NVIDIA V100. The weights W of INR are initialized using Xavier algorithm [15]. Following the established works [29, 51], we conduct our experiments using the same settings. For 2D images, the model processes 2D feature vectors (x, y) with four hidden layers of 256 units each, outputting a 3D RGB vector. It is trained on datasets like the DIV2K-test, a 1,000 subset of Imagenet-1k, and the COCO test set using the Adam optimizer with a batch size of 1, a learning rate of 1×10^{-4} , over 5,000 epochs. For videos, it processes 3D feature vectors (x, y, t) with three hidden layers of 1,024 units each, outputting a 3D RGB vector trained on Cat and Bikes videos with similar optimization parameters over 100,000 epochs. For audio, the model handles 1D feature vectors t with three hidden layers of 256 units each and an initial frequency $\omega_0 = 3,000$, trained on Counting and Bach audio data [51] with a learning rate of 1×10^{-4} over 1,000 steps. The SDF model processes 3D feature vectors (x, y, z) through three hidden layers of 256 units, outputting a 1D distance value, trained on data from TurboSquid and the Stanford 3D Scanning Repository with a batch size of 1,400 over 10,000 steps. For 3D scenes, NeRF uses a 5D input vector (x, y, z, θ, ϕ) with an 8-layer MLP of 256 channels per layer, processing 4,096 rays per batch, trained on LLFF and NeRF-blender datasets with a learning rate of 5×10^{-4} and a decay step of 500,000, simplifying the original NeRF strategy.

7.2 Additional experimental results

7.2.1 Impacts of stega ratios. We also provide visualized results that showcase the ratio \mathcal{S} , a critical parameter. As shown in Fig. A.2, the sparsity patterns and their impact on the model’s performance are depicted. The visuals aim to demonstrate how varying levels of stega ratio \mathcal{S} affect the representation performance of INR, elucidating the role of the ratio.

7.2.2 Impacts of pruning attack. We also provide visualized results that showcase the impacts of the pruning ratio \mathcal{S} . As shown in Fig. A.3. The results demonstrate how varying levels of pruning ratio \mathcal{S} affect the representation performance of stega and secret representation.

7.2.3 Visualized audio data. Fig. A.4 presents the visualization results of the audio data to complement the study. These visualizations illustrate the characteristics and patterns within the audio samples. Representations of audio data are provided in Fig. A.4, serving as a visual reference to understand our method for the audio data better.

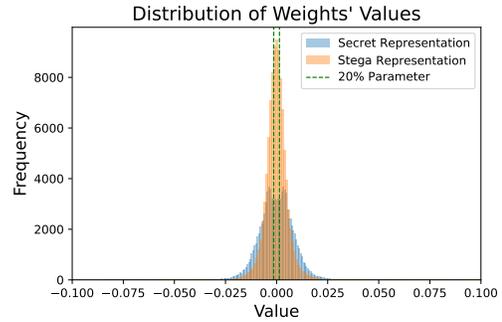


Figure A.1: Distribution of weight’s values. The weights for the stega representation are relatively small. Thus, during magnitude-based pruning, there is a tendency to prune the weights associated with the stega representation more. This will have a greater impact on the stega representation and a limited effect on the hidden secret representation.

7.3 Distribution of weight’s values

To better understand our method, we have provided the distribution of weight values for “stega” and “secret representation”. A standard SIREN [51] network for image representation has approximately 190,000 parameters. We visualize the range of parameter aggregation from -0.1 to 0.1 and provide the distribution of weight’s values in Fig. A.1. We observe that within the stega representation, the weight values of the secret representation are relatively high. Therefore, in magnitude-based pruning, the stega representation is more significantly affected. We present a series of results to portray the effects of pruning attacks on the system Fig. A.3 and illustrate the outcomes of these attacks, emphasizing the resilience of the proposed method. The visuals document the robustness of the approach, providing a breakdown of the system’s capability to withstand pruning attacks and maintain performance integrity.

¹<https://github.com/vsitzmann/siren>

²<https://github.com/yenchenlin/nerf-pytorch>

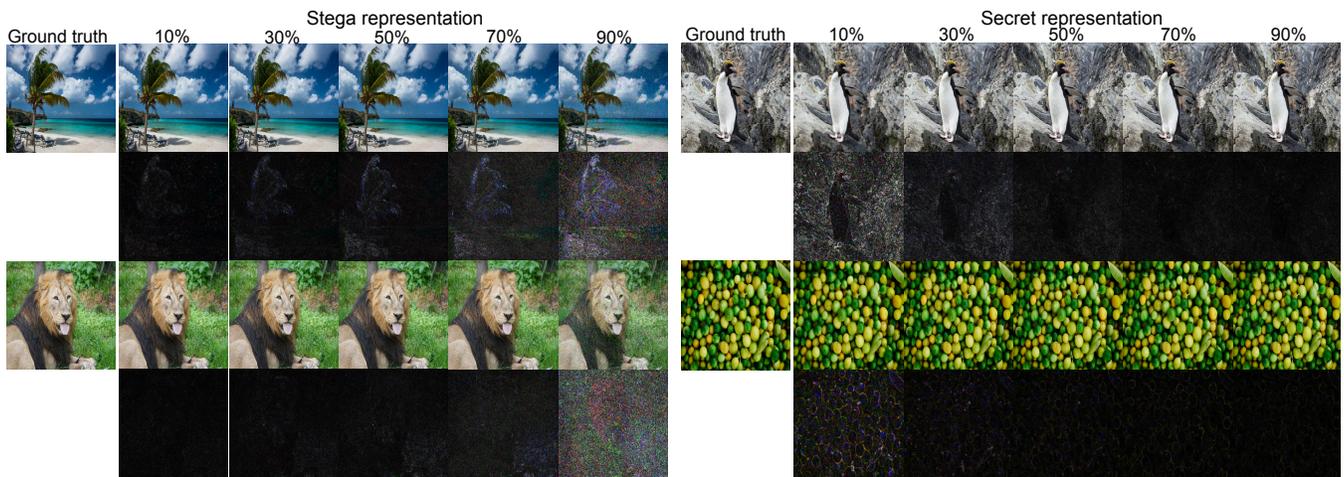


Figure A.2: Qualitative results of stega representation and secret representation with different stega ratios S . S denotes the ratio of parameters used for representing secret data. The residual images ($\times 5$) are located beneath each picture.

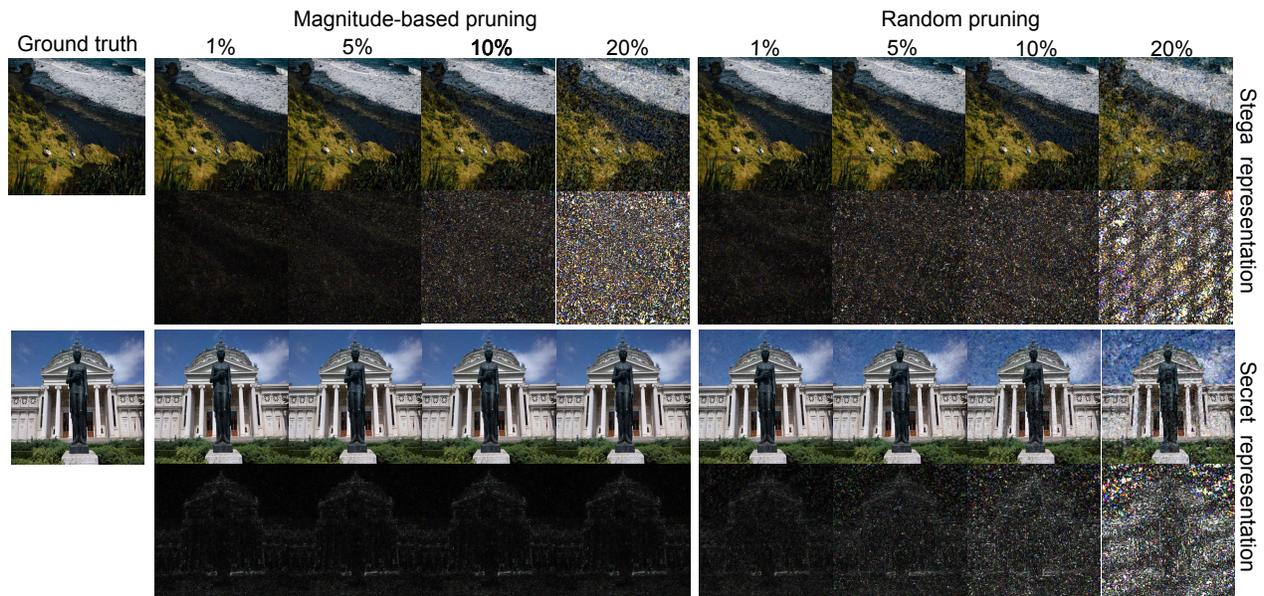


Figure A.3: Qualitative results of random pruning and magnitude-based pruning attack on INR, stega, and secret representation. Each neural network for representation is randomly pruning 1%, 5%, 10%, and 20% parameters. The residual images ($\times 5$) are located beneath each picture.

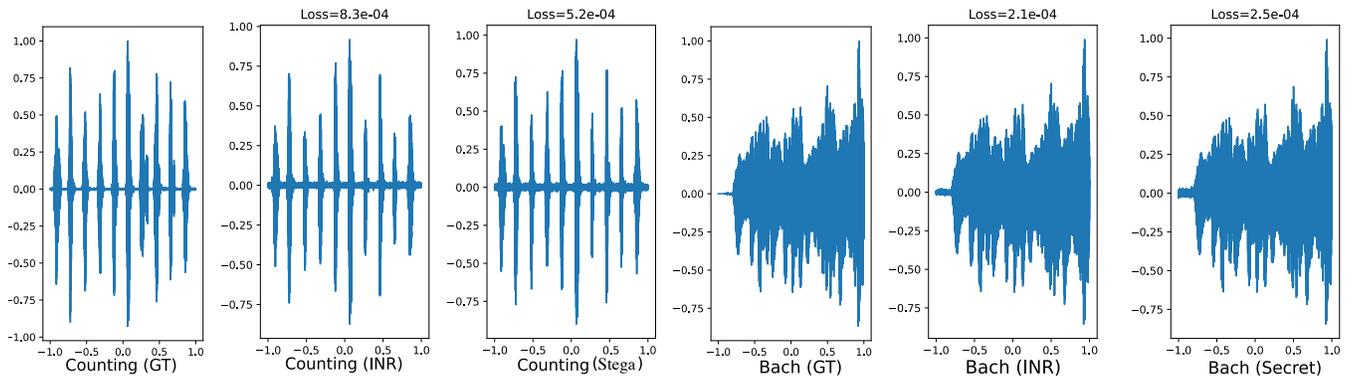


Figure A.4: A case study on audio format data. Bach (secret) is obtained from the representation of Counting (stega). The audio Counting (stega) is better than Counting (INR), which might be due to the audio data being simpler than other data formats.

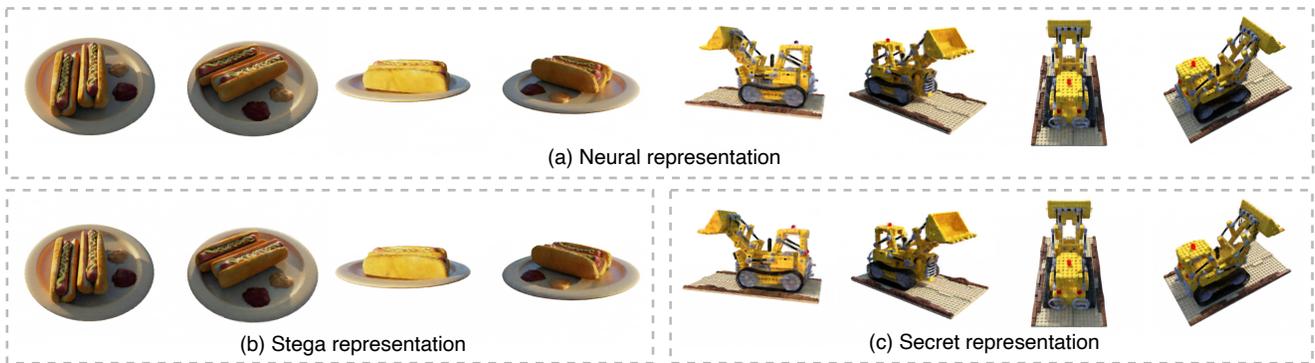


Figure A.5: Visual results of our proposed method on 3D scene [38]. We embed a secret representation (lego) into the stega representation (hotdog). (a) INR-based representation denotes the baseline results of NeRF when rendering both scenes. Scene (c) secret representation is obtained from (b) stega representation.