Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2024.054542

ARTICLE

A Dynamic YOLO-Based Sequence-Matching Model for Efficient Coverless Image Steganography

Jiajun Liu[1], Lina Tan[1,*], Zhili Zhou[2], Weijin Jiang[1], Yi Li[1] and Peng Chen[1]

[1]School of Computer Science, Hunan University of Technology and Business, Changsha, 410205, China

[2]Institute of Artificial Intelligence, Guangzhou University, Guangzhou, 510555, China

*Corresponding Author: Lina Tan. Email: loebmail@126.com

Received: 31 May 2024 Accepted: 19 September 2024 Published: 18 November 2024

ABSTRACT

Many existing coverless steganography methods establish a mapping relationship between cover images and hidden data. One issue with these methods is that as the steganographic capacity increases, the number of images stored in the database grows exponentially. This makes it challenging to build and manage a large image database. To improve the image library utilization and anti-attack capability of the steganography system, we propose an efficient coverless scheme based on dynamically matched substrings. We utilize You Only Look Once (YOLO) for selecting optimal objects and create a mapping dictionary between these objects and scrambling factors. Using this dictionary, each image is effectively assigned to a specific scrambling factor, which is then used to scramble the receiver's sequence key. To achieve sufficient steganography capability with a limited image library, all substrings of the scrambled sequences have the potential to hide data. After matching the secret information, the ideal number of stego images will be obtained from the database. According to experimental results, this technology outperforms most previous works in terms of data load, transmission security, and hiding capacity. It can recover an average of 79.85% of secret information under typical geometric attacks, and only approximately 200 random images are needed to achieve a capacity of 19 bits per image.

KEYWORDS

Coverless; steganography; object detection; YOLO

1 Introduction

Steganography is a method used to transmit secret information through various media such as videos, images, audio, or text. Despite significant advancements in image steganography [1–5], modifying image pixels or transform domain coefficients always leads to changes in the statistical characteristics of the images, which can be detected by some steganalysis techniques. In contrast, coverless steganography [6] directly selects images as stego ones based on mapping rules. Zheng et al. [7] improved this method by incorporating Scale-Invariant Feature Transform (SIFT) feature points, enhancing resistance to rotation and scaling attacks. Zou et al. [8] introduced a model based on average pixel values to increase steganography capacity. In [9,10], sub-block coefficients in the transform domain are used to create robust feature sequences, which demonstrate improved resistance to noise

attacks compared to previous models but are sensitive to geometric attacks. In 2023, Tan et al. [11] focused on image features and designed a low-similarity feature for mapping sequences, which reduces the database load to some extent.

Deep learning, especially reinforcement learning [12], is increasingly being utilized in steganography and coverless steganography based on mapping rules. In 2020, Luo et al. [13] proposed a steganography method based on Faster Region-based Convolutional Network (Faster RCNN), which established a mapping dictionary between objects and labels for concealing secret information. This method is resilient against geometric attacks but less tolerant to noise compared to previous works. In the same year, Liu et al. [14] employed Densely Connected Convolutional Networks (DenseNet) [15] to extract high-dimensional Convolutional Neural Network (CNN) features for hash sequence mapping. Another approach [16] hides secret information through semantic feature extraction and image object region segmentation. Additionally, a coverless scheme using camouflage images and CNN features was introduced [17], which offers more flexible capacity settings and robustness against image attacks. However, these techniques usually require building a sizable image database, posing challenges in maintaining uniqueness and preventing feature collisions among those extracted from different images. Tan et al. [18] introduced a novel approach to coverless image steganography, leveraging human pose data for enhanced concealment and robustness against steganalysis, achieving up to 92% average robustness across multiple datasets.

Cover image generation is another method to reduce database load, which uses generative models, such as Generative Adversarial Networks (GAN) [19]. In 2017, using Auxiliary Classifier Generative Adversarial Network (ACGAN) was proposed to create stego images by converting hidden data into noise and feeding it into the ACGAN network [20]. However, the generated images lacked realism. Zhang et al. [21] used GAN to create several texture images and map them to binary sequences. However, the diversity of images decreases when their number exceeds a certain threshold. To boost hiding capacity, Chen et al. [22] used SIFT [23] to select images and Star Generative Adversarial Network (StarGAN) [24] to create new images by mapping facial features to hidden data. While this improves security and robustness, the capacity is still limited. A multi-domain image transformation method [25] was proposed to handle the lack of cover images. It hides messages with a generator and recovers them with a classifier. Another method [26], inspired by neural network irreversibility, used gradient descent to update the noise vector for data extraction. In 2024, Wen et al. [27] designed a method to convert stego images into realistic ones, creating more high-quality images. Another scheme [28] used GANs to generate anime characters for secure communication, improving image quality with super-resolution GAN (SRGAN). This method offers high embedding capacity and robustness with low distortion, surpassing existing text-hiding techniques. While these methods reduce the load on image databases, the resulting images lack quality and diversity, making them easier for attackers to detect. Additionally, image content might be lost or altered during network operations, affecting data extraction. Especially, Setiadi et al. [29] reviewed and classified steganographic methods, challenges, and datasets, and Table 1 compares several methods.

Table 1: Comparison of coverless steganography

Ref.	Advantages	Disadvantages
Ours	An ultra-low database load.	Less robust against noise attacks than frequency domain-based methods.

(Continued)

Table 1 (continued)

Ref.	Advantages	Disadvantages
[6–11]	Without altering the image.	A heavily loaded database and low robustness against geometric attacks.
[13–18]	Better robustness against geometric attacks.	Lower robustness against noise attacks compared to frequency domain-based methods.
[19–22,25–28]	Smaller database load.	The generated stego images are not natural enough.

In coverless steganography, we are exploring the idea of making the length of hidden information in each cover image variable rather than fixed. This is aimed at enhancing the hiding capacity while reducing the burden on the database. A sequence of length u generates (u2 + u)/2 binary substrings, which can be represented as an image. This approach allows for dynamic matching of secret information fragments with substrings of varying lengths. The larger the u is, the more substrings are yielded for information matching, and relatively fewer images are needed in the database. To enhance the limited robustness of coverless steganography, we leverage high-level semantic features for flexible mapping rules. Recent advancements in object detection models, especially YOLO, have shown superior real-time performance and higher average accuracy compared to techniques like Deformable Part Models (DPM) and Region-based Convolutional Neural Networks (R-CNN). This solution uses YOLO mainly due to its fast inference speed and stable feature extraction performance. The main contributions of this paper are as follows:

1. Given the superior detection accuracy and inference time of YOLOs, we explored an Optimal Object Filtering Algorithm (OOFA) using YOLO-v5 to establish a robust mapping rule between the secret information and cover images. Tests conducted under geometric and noise attacks demonstrate the remarkable robustness of our method, which reaches nearly 87.4%.
2. To increase the capacity for hiding information within images, we extend the sequence key length sufficient to dynamically match the secret information with multiple substrings. Experiments have shown that, under ideal conditions, our approach outperforms existing methods, with an average hiding capacity of 19 bits per image.
3. To mitigate the impact of hiding capacity on image database load, we hide the information indirectly in the keys rather than in the cover images. In contrast to other approaches, ours requires fewer cover images to meet the same hiding capacity.

The rest of this paper is organized as follows. Section 2 introduces fundamental tools applied in the scheme; Section 3 presents the main idea of the proposed method; Section 4 provides experimental results and analysis; Section 5 consists of the conclusion, where we discuss future directions.

2 Relevant Model

The YOLO algorithm series employs a single-stage neural network to perform object detection, including positioning and classification [30]. While YOLO-v8 may have a higher theoretical accuracy, YOLO-v5 has a smaller model and simpler architecture, maintaining a relatively high accuracy rate and running faster. This makes it more suitable for real-time object detection tasks. Therefore, when

considering both accuracy and real-time performance, it is more appropriate to choose YOLO-v5 over YOLO-v8 for object detection.

2.1 YOLO-v5

Without detailed grid partitioning, multiple targets are often present within the same grid [31]. However, the YOLO-v5 effectively addresses this shortcoming. The model uses two parameters to control its depth and breadth. It employs Mosaic data augmentation during training and adaptive image scaling during inference, enhancing inference speed by 37%. To prevent overfitting, it uses the DropBlock mechanism, which drops continuous regions of an activation map, forcing remaining units to learn better features. YOLO-v5 also incorporates a partial cross-stage network (CSPNet) to efficiently connect layers and enhance feature fusion, improving its performance.

At the output end, the CIoU loss function is adopted as the coordinate loss [32], which considers not only the overlapping area of the predicted bounding box(b) and the ground truth bounding box($\widehat{\mathrm{b}}$), but also the distance between the center point of them. CIoU converges faster with fewer iterations and improves average precision and average recall for object detection. The effectiveness of the YOLO-v5 network in our system is verified by the experimental results.

$$\mathrm{L}_{\mathrm{CIoU}} = 1 - \mathrm{IoU} + \frac{\mathrm{D}^2\left(\mathrm{b}, \widehat{\mathrm{b}}\right)}{\mathrm{C}^2\left(\mathrm{b}, \widehat{\mathrm{b}}\right)} + \alpha \cdot v \tag{1}$$

$$v = \left(\frac{2}{\pi}\left(\arctan\left(\frac{\widehat{w}}{\widehat{h}}\right) - \arctan\left(\frac{w}{h}\right)\right)\right)^2 \tag{2}$$

where IoU for assessing the overlap between box(b) and box($\widehat{\mathrm{b}}$), $\mathrm{D}^2\left(\mathrm{b}, \widehat{\mathrm{b}}\right)$ denoting the distance between the center points of box(b) and box($\widehat{\mathrm{b}}$), $\mathrm{C}^2\left(\mathrm{b}, \widehat{\mathrm{b}}\right)$ as the length of the diagonal of box(b) and box($\widehat{\mathrm{b}}$), α for weighting the aspect ratio penalty term. And v as a variable quantifying area discrepancies between box($\widehat{\mathrm{b}}$) and box(b). $\widehat{w}$ and $\widehat{h}$ represent the width and height of box($\widehat{\mathrm{b}}$), while w and h represent the width and height of box(b).

2.2 YOLO-v5 Models

The YOLO-v5 model [33] includes four variations: YOLO-v5s, YOLO-v5m, YOLO-v5l, and YOLO-v5x. The depth and width parameters for these models are detailed in Table 2. YOLO-v5s has the smallest network depth and width, while the other three models are progressively deeper and wider versions of YOLO-v5s. Smaller network models have lower performance requirements on mobile devices, making them easier to deploy. Therefore, YOLO-v5s is employed in our scheme.

Table 2: YOLO-v5 models

Parameter	YOLO-v5s	YOLO-v5m	YOLO-v5l	YOLO-v5x
Depth	0.33	0.67	1.00	1.33
Width	0.50	0.75	1.00	1.25

3 The Proposed Method

3.1 Overall Framework

Fig. 1 illustrates the proposed framework, which consists of three main components: preprocessing, information hiding, and information extraction. Preprocessing is aimed at improving the efficiency of the sender in hiding information. During this stage, each receiver has a unique sequence key (SK). Once YOLO-v5 identifies all cover images in the database, a mapping dictionary must be established to generate the scrambling factors. These factors will then be used to scrambled the sequence keys, leading to the development of a data architecture based on the relationships between the sequence keys, scrambling factors, scrambled sequences, and cover images. Information hiding refers to the sender matching steganographic images based on the secret information. After secret data matches substrings of scrambled sequences, the sender transmits the position key and corresponding stego images to the receiver. Information extraction involves the receiver recovering the secret information from the set of steganographic images.

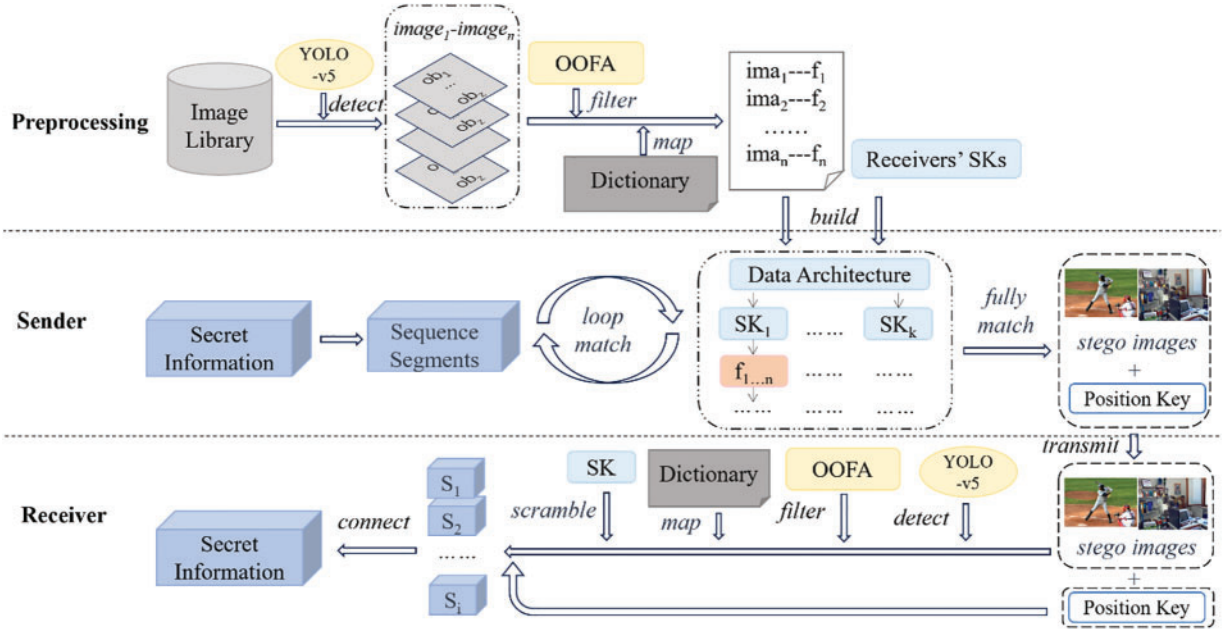

Figure 1: Flowchart of the coverless image steganography algorithm

3.2 Preprocessing

In our system, the sender and each receiver agree on a specific sequence key while ensuring perfect confidentiality. Preprocessing tasks like establishing the mapping dictionary, producing the scrambling factors, and creating the inverted index, are necessary before transferring the secret information.

3.2.1 Mapping Dictionary Creation

A mapping dictionary needs to be built to convert the object labels of images recognized by YOLO-v5 into scrambling factors. After all images in the database are detected by YOLO-v5, a label list is obtained for all the objects. These labels are sorted alphabetically and each object will be assigned a unique scrambling factor according to the mapping dictionary. The mapping rules can be updated periodically to ensure security, such as in ascending or descending order.

3.2.2 Sequence Key Distribution

Each receiver must be issued a unique sequence key of length t whose value is empirically set to 10000 following capacity and time cost testing. There are two approaches to assigning each receiver a specific sequence key. One way is to create a unique sequence by using the recipient ID as a random seed and reordering the initial sequence by the pseudo-random number generator. Another way is to manually distribute a bit-string of length t to each recipient, which exhibits a high degree of randomness, and when combined with proper security measures, it becomes exceedingly challenging for any third party to deduce the assigned bit strings. Considering that the system is more vulnerable to attacks if the attacker is skilled with the pseudo-random function and receiver IDs, we choose the first method for key distribution.

3.2.3 Scrambling Factor Generation

The previously created mapping dictionary has established a mapping relationship between the image objects and scrambling factors. YOLO-v5 may find multiple objects from an image since it handles multi-object classification and localization. It's worth noting that those objects with lower category probabilities may not be detected again after the image is attacked, which would result in poor robustness of the steganography model. Therefore, our algorithm will eliminate those undesirable bounding boxes and keep only the ones with the highest category probability. The generation process of scrambling factors is given as follows:

Step 1: Detect an input image c with $YOLO$ to get all the objects $OB = ob_1||ob_2|| \ldots ||ob_n$. The area of the bounding box and class probability of each object are recorded as $A(ob)$ and $P(ob)$, respectively.

Step 2: In general, objects with a larger area or higher category probability tend to exhibit better robustness. Therefore, we set thresholds P and A to filter out objects with low robustness performance. After filtering out objects whose bounding box area is less than A, we select the one (only one) as ob_{opt} with a class probability exceeding P and maximum among the remaining objects.

$$OB_{filtered} = \{ob_i \in OB | A_i > A\} \tag{3}$$

$$ob_{opt} = \underset{ob_i \in OB_{filtered}}{\arg\max}\{P_i | P_i > P\} \tag{4}$$

Step 3: If the ob_{opt} cannot be found, it means that the image is not suitable as a stego image and should be discarded. Otherwise, a scrambling factor is generated according to the dictionary D, which will be used as a random seed input into the random function to scramble the sequence key.

$$f = \begin{cases} NUll, & ob_{opt} == Null; \\ D[ob_{opt}], & else; \end{cases} \tag{5}$$

After the aforementioned steps, images with suitable objects can generate a scrambling factor. Through extensive experiments, we found that a 15% filtering factor significantly alters the image's features to enhance security and privacy while maintaining sufficient image quality for practical use. Moreover, our research revealed that the class probability of certain objects might not decrease, and could even increase, after being attacked. In such cases, setting the value of P to 50% provides an optimal balance. The 50% setting accounts for the uncertainty in class probabilities and ensures the system's robustness and reliability against various potential attacks.

3.2.4 Building Data Architecture

After completing the preceding three steps, we can construct a four-tiered data architecture. This architecture is pre-generated and stored, which means the sender doesn't have to process all images again before hiding the secret information. This significantly reduces the time needed for the information hiding process. The first tier comprises sequence keys that are shared among all receivers. The second tier encompasses scrambling factors derived from the images themselves. The third tier involves scrambled sequences (SS) obtained by applying the scrambling factors to the sequence keys. Each fourth tier contains a collection of images, from which the same scrambling factor can be extracted if they belong to the same list.

It is crucial to emphasize that images with similar suitable objects will be associated with the same scrambling factor and subsequently grouped into the same image list. This organization facilitates efficient handling and retrieval. Fig. 2 illustrates the outlined data structure.

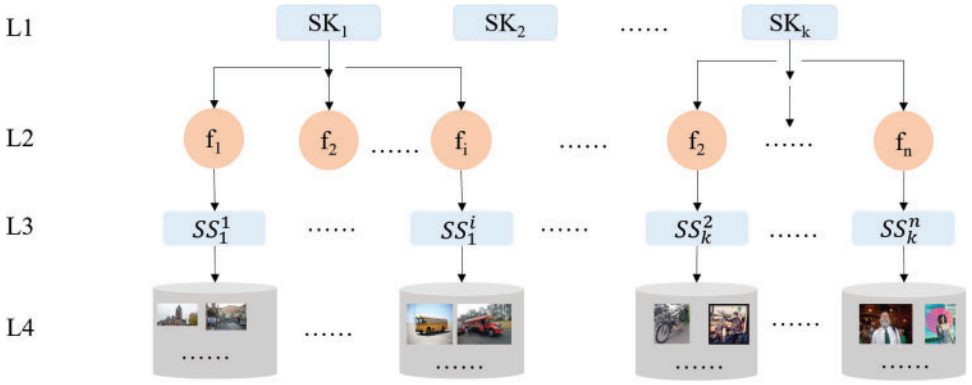

Figure 2: The data architecture of image database

3.3 Information Hiding

Before selecting stego images, the secret information SM needs to be converted into a binary stream M whose length is denoted as L. The length of the sequence key also needs to be considered during the matching process, which is noted t. Then match the image in the data architecture mentioned above, which can be divided into the following steps:

Step 1: For SM, select its first n bits $MG = mg_1||mg_2||\ldots||mg_n$. Due to the impossibility of matching extensive information within brief sequences, the value of n in the first matching is computed by Eq. (6) to steer clear of such futile matching endeavors.

$$n_{first} = min\,(t, L) \tag{6}$$

Step 2: Match all substrings with length n in the third level of the data architecture, $Match\,(MG, SS) == 0$ indicates the matching is failed, then set n to $n-1$, repeat this step. Otherwise, a stego image si_m could be obtained.

$$\begin{cases} n = n-1, & Match\,(MG,\ SS) == 0; \\ SI = SI||si_m, & else; \end{cases} \tag{7}$$

$$Match\,(MG,\ SS) = max_{i=1}^{L-n+1}\delta\,(MG,\ SS[i\colon i+n-1]) \tag{8}$$

Step 3: In addition, the location information $key_m = \{k_f,\ k_l\}$ (including the left position index and length of the substring) should be record.

$$k_f = argmax\,(Match\,(MG,\ SS)) \tag{9}$$

$$k_l = n \tag{10}$$

Step 4: Regard the rest of information as a new piece of information, and set L to $L - n$, repeat the previous steps until all the secret information is matched successfully, and get the stego images $SI = si_1||si_2||\ldots||si_m$.

Step 5: Since the position and length of substrings are also the key factors to recover secret information accurately, the $Key = key_1||key_2||\ldots||key_m$ should be transmitted to the receiver. In light of the item of efficiency, we use AES encryption algorithm to encrypt *Key* before the transfer.

This algorithm design ensures the security of secret information while maximizing the hiding capacity of the stego image. Furthermore, we can pass *Key* and stego images to the receiver respectively in different time periods and channels.

3.4 Information Extraction

Some stego images may be attacked during transmission and the suitable objects fails to be detected, so that the receiver cannot restore the information accurately. For avoiding a missing object affecting the extraction of other location information, the same number of '0' need to be added to the missing position. The *Key* can provide us with the lengths of information hidden in stego images. This operation is shown in Fig. 3.

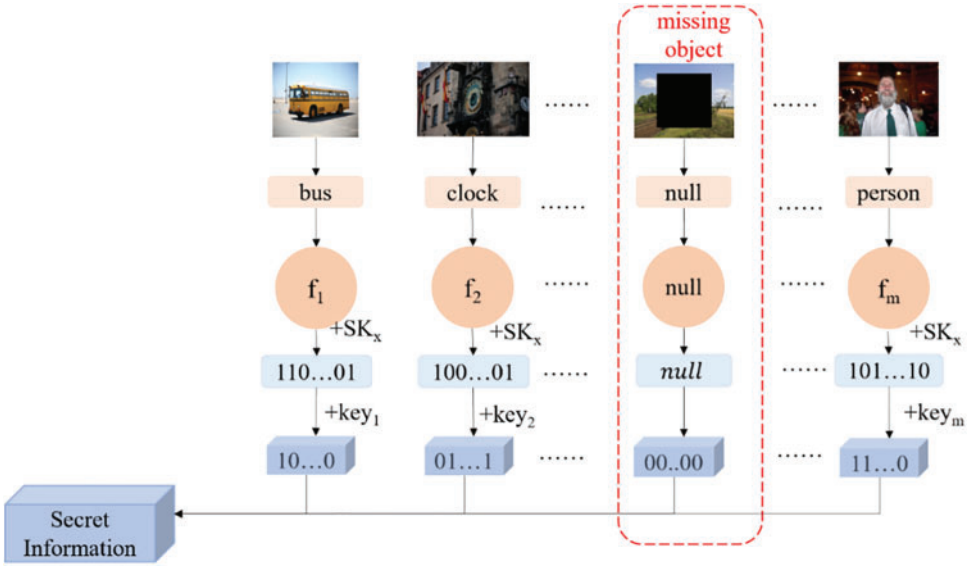

Figure 3: Process chart for information extraction

The steps of information extraction are as follows:

Step 1: Firstly, get object $OB_j = ob_1||ob_2||\ldots||ob_n$ after using YOLO-v5 to detect the jth stego image si_j, with $1 \leqq j \leqq$ m.

Step 2: Secondly, the objects with small bounding box area less than A are filtered out from OB_j, then The object with the highest classification probability is chosen as ob_{opt}.

$$OB_{filtered} = \{ob_i \in OB_j | A_i > A\} \tag{11}$$

$$ob_{opt} = \underset{ob_i \in OB_{filtered}}{\arg\max} \{P_i | P_i > P\} \tag{12}$$

Step 3: Match ob_{opt} in the mapping dictionary D to get a scrambling factor f_j. Then f_j is used to scramble the sequence key SK_x own by receiver to generate the scrambled sequence SS_x^j.

$$f_j = D\left[ob_{opt}\right] \tag{13}$$

$$SS_x^j = Scrambling\left(SK_x, f_j\right) \tag{14}$$

Step 4: Extract the hidden secret information mg_j of the image from SS_x through key_j.

$$mg_j = SS_x^j\left[key_j[0], key_j[0] + key_j[1] - 1\right] \tag{15}$$

Step 5: Finally, the all information fragments extracted from stego images are spliced together to recover the binary stream representation of secret information $M = mg_1||mg_2||\ldots||mg_m$, then M is converted to secret information SM.

4 Experiments and Analysis

Experimental environment: Intel(R) Xeon(R) Silver 4310 CPU @ 2.10 GHz, 30.00 GB RAM and one Nvidia GeForce RTX A4000 GPU. All experiments are completed in Pycharm and MATLAB R2021a.

Data set: MS COCO 2017 includes 118,287 training images, 5000 validation images, and 40,670 test images. This data set contains natural pictures and common object pictures in life and and is known for its complex background and relatively large number of objects, making it ideal for training YOLO-v5. And all comparative experiments use this dataset in this section.

4.1 Efficiency

There are various operations involved in the entire secret information transmission process, such as feature extraction, image matching, etc., so the efficiency of this method is also a crucial point to be taken into consideration. In this section, we will discuss the efficiency of our approach from three perspectives: (1) preprocessing efficiency; (2) image matching efficiency; (3) information extraction efficiency.

4.1.1 Preprocessing Efficiency

Due to the linear relationship between the time consumption of both object detection and establishing the data structure with the number of images, the time complexity for preprocessing can be expressed as $O(a)$, wherein a denotes the quantity of inputted images.

We randomly select 100,..., 1300 images to form a database, and then evaluate the time of object detection and the construction of data architecture, respectively. The test results are presented in Table 3.

Table 3: Preprocessing efficiency

Number	Time of detection (s)	Time of data architecture construction (s)
100	8.89	0.68
400	15.21	1.27
700	21.89	1.47
1000	29.47	1.60
1300	33.58	1.66

As can be observed from Table 1, YOLO-v5 offers a remarkable detection speed, with a remarkably low time consumption for data architecture construction.

4.1.2 Image Matching Efficiency

We can assess the efficiency by analyzing two aspects: the comparison of secret information of various lengths, and the comparison of sequences key of various lengths. The time complexity of secret message hiding depends on the substring matching during the iteration process. In the worst-case scenario, each substring must be compared against the scrambled sequence, incurring a linear time complexity. Consequently, the overall time complexity can be expressed as $O(nt)$. And the results of this analysis are presented in Table 4.

Table 4: Image matching efficiency (second)

	Key (bits)			
Message (bits)	100	400	800	1000
2000	0.901	5.279	17.954	26.613
4000	1.280	11.229	41.883	64.707
6000	2.500	16.813	60.575	86.016

As is shown in Table 3, the time consumed in searching for stego images is not small, owing to the fact that a tremendous amount of substrings take part in the process of matching secret data.

4.1.3 Information Extraction Efficiency

In this subsection, we adopt the same parameters as in the preceding subsection to evaluate the efficiency of information extraction. Similar to the information hiding phase, the efficiency of information extraction is also constrained by the length of the sequence key and the secret message. Likewise, its time complexity is evaluated as $O(nt)$. The results are illustrated in Table 5.

Information extraction is a time-intensive process primarily because YOLO-v5 needs to be restarted for each image detection, leading to significant redundancy in time consumption. Therefore, it consumes more time compared to information hiding.

Table 5: Information extraction efficiency (second)

	Key (bits)			
Message (bits)	100	400	800	1000
2000	40.704	31.850	31.340	30.369
4000	76.201	65.619	59.318	55.991
6000	116.514	105.366	88.970	87.327

4.2 Hiding Capacity

4.2.1 Capacity Test of Our Method

It is worth noting that a longer sequence key and more scramble factors can allow more substrings to be involved in the matching process, thus increasing the probability of long substrings being successfully matched. As the steganography capacity depends on the length of the sequence key and the number of scrambling factors, we test the capacity by comparing both sequence keys with different lengths and image databases with different sizes.

In this test, we randomly selected 100,..., 5000 images from the COCO dataset to form 6 image databases. For each database, we conducted tests by setting the length of sequence keys to 100,..., 50,000 bits. The results are presented in Table 6 and Fig. 4. We can observe that when the number of images surpasses a certain value, its impact on the capacity is insignificant. Additionally, within the same size database, we found that the steganography capacity increases with the augmentation of the length of the sequence key, yet the increment is significantly diminishing. Taking into account the retrieval efficiency, we recommend setting the length of the sequence key to 10,000 where the steganography capacity reaches around 19 bits.

Table 6: Hiding capacity of our scheme

	Length (bits)									
Images	100	400	800	1000	2000	5000	8000	10,000	15,000	20,000
100	11.688	14.754	14.876	15.517	16.364	17.822	18.182	18.947	19.149	19.565
500	12.676	14.876	15.652	16.071	17.143	18.557	19.149	19.565	20.0	20.455
1000	12.676	14.634	15.929	16.216	17.143	18.557	19.355	19.355	20.225	20.690
1500	12.857	15.0	16.216	16.364	17.308	18.75	19.565	19.565	20.225	20.455
2000	12.950	15.0	16.216	16.822	17.647	18.75	19.565	19.565	20.455	20.690
5000	13.043	15.0	16.364	16.822	17.876	18.75	19.565	19.565	20.455	20.930

4.2.2 Comparison of Capacity

In this part, our method is compared with a selection of other methods that rely on mapping rules. Unfortunately, some models could not be reproduced, so the data used for comparison was taken directly from the original paper or from papers cited therein. The comparison results are given in the Table 7. The steganography capacities in [13,16,34] are (8∼15) $\times$ n, 6 $\times$ n, and 5 $\times$ n, respectively.

In [16], the value of n is determined by the number of objects included in each cover image, while in [13], n depended on the number of objects (less than 4) with the largest areas, and in [34], n hinge on the number of object areas of 5–25 kb in an image. However, most of the images only have 1∼3 detectable objects, that is to say, the value of n is 1∼3 in general. Even if images containing multiple objects are purposely collected in order to assemble the database, these images tend to lack robustness. A cover image can be partitioned into nine blocks with the aid of methods [6,7], thereby yielding an 8-bit sequence. Moreover, methods [9,10] embed the secret message in the transform domain, having a steganography capacity of 1∼15 bits per image. Nevertheless, our method still has a slight advantage over these methods. However, the capacity of coverless steganography is typically lower than that of traditional steganography due to the limitations of the mapping rules that result in the collision-prone nature of image features.

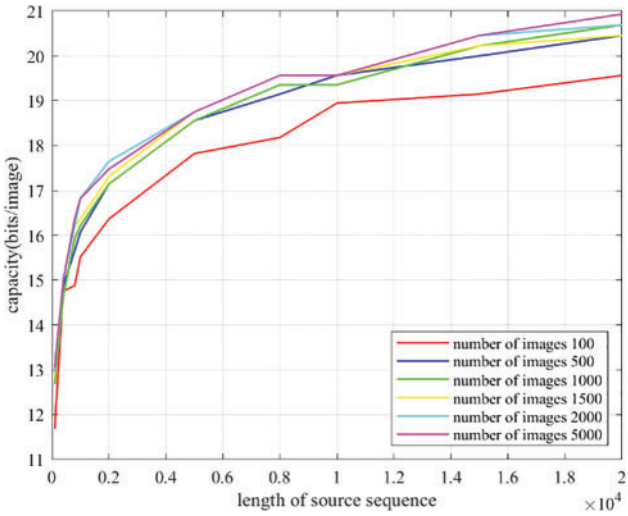

Figure 4: Capacity test under source sequences with different lengths

Table 7: The capacity of schemes

Ref.	Capacity (bits/image)
Ours	19
[11]	8∼25
[17]	1∼15
[16]	(8∼15) × n
[13]	6 × n
[34]	5 × n
[6]	8
[7]	8
[9]	1∼15
[10]	1∼15

4.3 Database Size

In order to carry out image matching on every conceivable binary stream of confidential information, it is essential to have a comprehensive database. However, the current coverless steganography is limited by the length of the hash, and the reason why the length of the hash cannot be overly long is because of the difficulty in constructing a comprehensive database. Therefore, it is worth exploring how to effectively reduce the size of the image database.

4.3.1 Database Size Test of Our Method

In the dictionary we have set up, each object is associated with a distinct scrambling factor, so the amount of detected entities that satisfy the concealed demand decides the amount of scrambling factors. In this section, we investigate how databases of different sizes influence the number of scrambling factors. To do this, we conduct experiments with various databases and evaluate the number of scrambling factors for each. According to the preceding section, the length of the sequence key is set to 10,000. It is evident from Table 6 that the size of the database exerts a greater influence on the steganography capacity when its value is within 1500. Therefore, it is advisable to choose a suitable size of the database within this range. We randomly drew 50,..., 1200 images from the COCO dataset to build the dataset, respectively. After conducting a series of experiments, the average number of scrambling factors generated by each dataset was determined, and the results were listed in Table 8.

Table 8: Database size test (the length of sequence is 10,000)

Number of images	Number of factors	Capacity (bits/image)
50	22	17.947
100	34	18.680
200	50	19.151
400	55	19.273
600	62	19.440
800	66	19.566
1000	67	19.608
1200	75	19.620

From Fig. 5, it is clear that the number of scrambling factors increases significantly when the database size increases from 50 to 200. However, when the size of the database is increased from 200 to 1200, the change in the number of scrambling factors is not very remarkable. This implies that 200 cover images are adequate to generate a large amount of scrambling factors, satisfying the requirements of steganography.

4.3.2 Comparison of Database Size

As should be expected, we compare our method with other models based on mapping rules. Existing methods, after all, directly map the features of cover images into binary sequences. Consequently, the steganography capacity of such methods is contingent upon the sequence length, and a larger sequence length means a larger database. Due to the different capacities utilized in these methods, it is impossible to make a direct comparison between their image database sizes. All methods are supposed to be expanded to 19 bits based on their original methods, and then the database size value can be

attained via calculation. Considering that we could not get a sufficient amount of images to finish up the experiment, we had to make a manual estimation of the theoretical minimum.

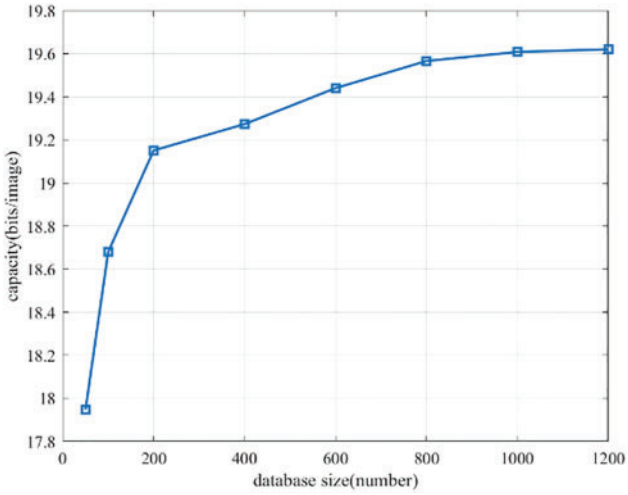

Figure 5: The relationship between image database size and steganography capacity

In [6,7], their models map features of each cover image to a binary sequence with a fixed length. Consequently, the length of the binary sequence determines the minimum number of cover images that the image databases need. Taking v as the length, the required number of covers theoretically is 2^v. Therefore, the steganography capacities of their models are $2^{19} = 524{,}288$. In [9,10], each cover image is partitioned into l sub-blocks, and for each sub-block, a sequence of length k can be generated. Considering factors such as robustness, the maximum value of l is set to 256 in its experiment, so the image can be mapped to $16 \times 16 = 256$ sequences, $2^{19}/256 = 2048$ images are needed at least. Methods [13,34] proposed a mapping rule between objects and bit strings. However, regardless of the number of objects in the cover image, it can only be mapped to one sequence in the end, thus requiring a minimum of $2^{19} = 524{,}288$ images for these methods. In [16], each image can be mapped to different sequences according to different object areas. However, there are only about 1∼3 objects that are suitable for representing secret information in an image. As a result, the minimal amount of images required is calculated to be $2^{19}/3 = 174{,}763$. In [11], each image can be mapped to up to 64 different sequences, so a minimum of $219/64 = 8129$ images need to be reserved. The results are shown in the Table 9. Of course, having considered the collisions between features of images, the sizes of the database are much larger than these values indicated.

Table 9: Database size required for different methods (for 19-bit capacity)

Ref.	Descriptions	Number of database images
Ours	It generates a scrambling factor to hide information	**200**
[11]	It generates a binary sequence based on CS-DCTR feature	≥8129
[15]	It generates a binary sequence based on object areas	≥174,763
[12]	It generates a sequence with fix length based on the number of objects	≥524,288

(Continued)

Table 9 (continued)

Ref.	Descriptions	Number of database images
[34]	ditto	≥524,288
[6]	It generates a sequence with fix length based on average pixel	≥524,288
[7]	It generates a sequence with fix length based on SIFT feature	≥524,288
[9]	It generates l binary sequence based on the DCT coefficients of the sub-blocks	≥2048
[10]	It generates l binary sequence based on the DWT coefficients of the sub-blocks	≥2048

From the comparison, we can find that other methods require an enormous amount of effort and time to construct an image database, whereas our method only necessitates around 200 random images to build the same. Consequently, our method has a great predominance in database management by comparison.

4.4 Robustness

During the transmission of stego images, malicious attackers may attempt to destroy them, thus will give rise to the hidden information of the images may not be accurately extracted. Therefore, robustness is also an important indicator to measure the quality of a steganography model.

All the pixels of the stego image may affect the extraction of information in vast majority of methods, so the rate of secret information extracted correctly is used to measure robustness. The calculation method for:

$$R = \frac{SM_c}{SM_o} \times 100\% \tag{16}$$

where SM_c is the length of the correctly extracted secret information, and SM_o is the length of the original information.

In this section, 4 common geometric attacks and 7 common noise attacks are used to test the robustness of the methods. And the performances of each attack under different parameters are supposed to be tested. All attack types and their associated parameters are shown in the Tables 10 and 11. The Figs. 6 and 7 illustrate the effects of these attacks.

Table 10: The specific parameters of geometric attack

Attack	Parameters
Centered cropping	Ratios: 5%, 10%, 20%
Edge cropping	Ratios: 5%, 10%, 20%
Translation	(36, 20), (40, 25), (80, 50)
Rotation	Rotation angles: 5, 10, 30

Table 11: The specific parameters of noise attacks

Attack	Parameters
Gaussian noise	The mean u: 0, the variances q: 0.001
Salt and pepper noise	The mean u: 0, the variances q: 0.001
Speckle noise	The mean u: 0, the variances q: 0.01
Mean filtering	The window size: 3 × 3
Median filtering	The window size: 3 × 3
Gaussian low-pass filtering	The window size: 3 × 3
Scaling	The scaling ratios: 3

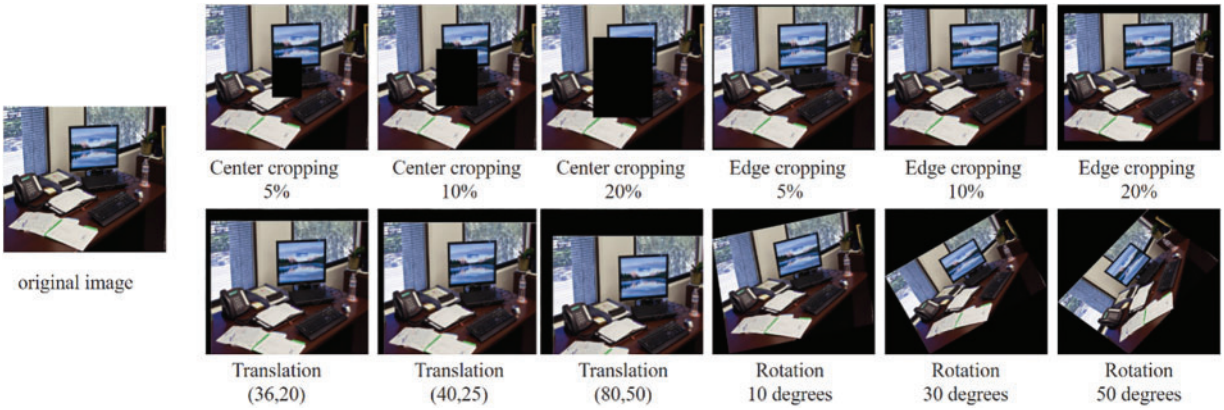

Figure 6: The geometric attack schemes

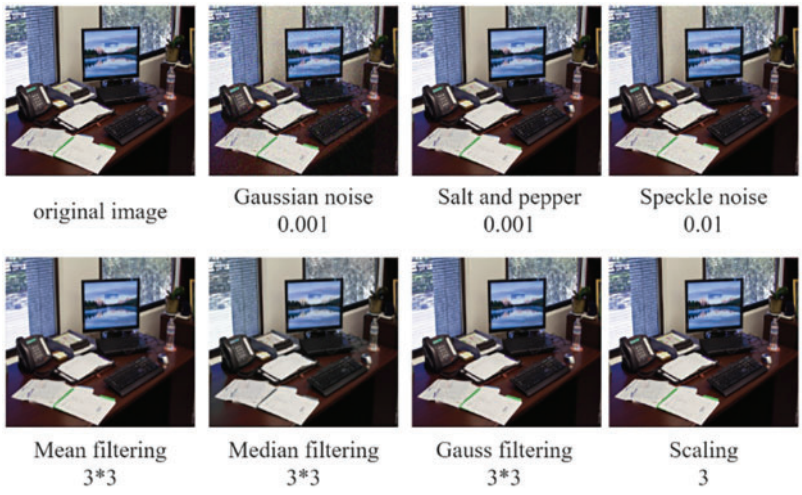

Figure 7: The noise attack schemes

Next, we will test and analyze the result of the methods under geometric attack and noise attack separately. For those models that could not be reproduced, we will take the data from the related papers

directly. And it ensures that all data was derived from experiments conducted using the same COCO dataset.

4.4.1 Robustness Comparison on Geometric Attacks

Traditional coverless steganography has poor robustness against geometric attacks, as the features used to map sequences are prone to be damaged or lost after the stego images have been attacked. Particularly when it comes to rotation attacks, its robustness is less than 10%. Nevertheless, there is a more optimistic outcome in most deep learning-based coverless steganography. This is because when constructing mappings, the objects used may still be detected after being attacked. It is noteworthy that the success of this type of model is contingent on the precision of the object detection model and the details of the mapping rules. Table 12 shows that the YOLO-v5 and mapping algorithms mentioned in our method have a good promotion in robustness. Facing to cropping attacks, in particular, it has an ascending advantage over other deep learning-based methods.

Table 12: The robustness of geometric attacks

Attack	Parameters	Ours	[11]	[13]	[34]	[6]	[7]	[9]	[10]
Center cropping	5%	**84.4%**	40.0%	51.2%	27.6%	47.4%	42.8%	48.4%	47.6%
	10%	**76.0%**	29.1%	46.2%	23.2%	29.4%	22.6%	30.2%	27.0%
	20%	**57.3%**	9.7%	34.8%	16.4%	22.4%	6.0%	22.6%	15.2%
Edge cropping	5%	**94.2%**	41.3%	87.6%	59.6%	58.2%	31.2%	59.4%	64.2%
	10%	**95.6%**	22.8%	86.2%	57.8%	38.8%	13.0%	39.8%	45.8%
	20%	**90.2%**	10.3%	82.0%	54.0%	23.2%	6.8%	21.6%	23.8%
Rotation	10	**86.7%**	5.0%	78.4%	50.8%	8.0%	2.6%	8.0%	8.6%
	30	61.3%	1.2%	**63.2%**	40.2%	1.4%	1.8%	1.4%	0.8%
	50	**50.7%**	0.6%	46.4%	29.8%	1.6%	1.2%	1.8%	0.8%
Translation	(36, 20)	**89.8%**	12.6%	83.2%	54.8%	20.4%	4.6%	20.2%	17.2%
	(40, 25)	**87.6%**	6.9%	83.6%	54.8%	16.4%	3.8%	16.6%	13.0%
	(80, 50)	**84.4%**	0.9%	77.0%	50.2%	6.0%	2.0%	5.2%	4.8%

4.4.2 Robustness Comparison on Noise Attacks

Traditional coverless steganography is highly resilient to noise attacks due to the selection of features that are less vulnerable to noise. On the contrary, the robustness of coverless steganography based on deep learning is slightly weaker by reason of the limit to the accuracy of existing object detection models. But as can be observed from Table 13, the performance is still impressive, with our method achieving an average of 87.4%, which is marginally better than other deep learning models.

Table 13: The robustness of noise attacks

Attack	Parameters	Ours	[11]	[13]	[34]	[6]	[7]	[9]	[10]
Gaussian noise	0.001	67.1%	94.7%	71.8%	43.2%	95.8%	65.6%	95.4%	**98.0%**
Salt and pepper noise	0.001	89.8%	98.0%	86.0%	59.2%	99.0%	90.8%	99.2%	**99.6%**

(Continued)

Table 13 (continued)

Attack	Parameters	Ours	[11]	[13]	[34]	[6]	[7]	[9]	[10]
Speckle noise	0.01	89.3%	96.4%	83.6%	56.4%	96.6%	74.4%	96.2%	**98.0%**
Median filtering	3 × 3	89.7%	97.1%	87.2%	56.6%	99.6%	88.4%	99.4%	**100%**
Mean filtering	3 × 3	90.2%	96.2%	86.0%	57.0%	**98.8%**	73.6%	95.8%	97.8%
Gaussian filtering	3 × 3	91.6%	95.8%	89.6%	61.2%	99.8%	92.6%	**100%**	**100%**
Scaling	3	94.2%	98.6%	91.2%	67.8%	99.6%	95.2%	**100%**	**100%**

4.5 Security

Our method effectively conceals secret information by forming a bond between objects, scrambling elements, and a sequence key. The statistical properties of the image remain unchanged, meaning that existing steganalysis techniques cannot detect stego images that contain the secret information. Thus it can be seen that our method has an impressive level of resistance to steganalysis.

Even if the stego images are suspected by the attacker and the *Key* is also cracked, the attacker would still be unable to extract the secret information, since the mapping dictionary and sequence key are agreed upon in advance and kept strictly confidential by both the sender and receiver. Despite the fact that one receiver inadvertently leaked the mapping dictionary and its sequence key, the security of the other receivers was not compromised, since each receiver has its own unique sequence key. This demonstrates the robustness of our method in terms of safety.

5 Conclusions

Our proposed method involves dynamic mapping-based coverless steganography. Unlike existing models, our scheme avoids direct image-sequence links and extracts a scrambling factor from the cover images. When the key is scrambled, a new sequence is created, with all substrings containing hidden information. The selection of stego images maximizes capacity within sequence length limits, reducing the load on the database. Each image can be mapped to multiple sequences, simplifying database management. Our method is effectively resistant to steganalysis, thereby ensuring the security of the algorithm. Experiments show robustness to geometric and noise attacks, although noise resistance is slightly less effective compared to traditional methods. The capacity is high but falls below traditional benchmarks. Our future work aims to significantly enhance the hidden ability of the method.

Acknowledgement: The authors would like to express their gratitude to all the researchers and reviewers who contributed to enhancing the quality of the idea, concept, and the paper overall.

Funding Statement: The authors received no specific funding for this study.

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Jiajun Liu, Lina Tan; data collection: Yi Li, Peng Chen; analysis and interpretation of results: Jiajun Liu, Zhili Zhou, Weijin Jiang; draft manuscript preparation: Lina Tan, Yi Li, Peng Chen. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are available from the corresponding author, Lina Tan, upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] C. H. Yang, C. Y. Weng, S. J. Wang, and H. M. Sun, "Adaptive data hiding in edge areas of images with spatial LSB domain systems," *IEEE Trans. Inform. Forensic. Secur.*, vol. 3, no. 3, pp. 488–497, 2008. doi: 10.1109/TIFS.2008.926097.

[2] R. T. Mckeon, "Strange fourier steganography in movies," in *2007 IEEE Int. Conf. Electro/Inform. Technol.*, Chicago, IL, USA, 2007, pp. 178–182. doi: 10.1109/EIT.2007.4374540.

[3] M. Y. Valandar, P. Ayubi, and M. J. Barani, "A new transform domain steganography based on modified logistic chaotic map for color images," *J. Inform. Secur. Appl.*, vol. 34, pp. 142–151, 2017. doi: 10.1016/j.jisa.2017.04.004.

[4] I. J. Cox, J. Kilian, T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for images, audio and video," in *Proc. 3rd IEEE Int. Conf. Image Process.*, Lausanne, Switzerland, 1996, vol. 3, pp. 243–246. doi: 10.1109/ICIP.1996.560429.

[5] W. H. Lin, S. J. Horng, T. W. Kao, P. Fan, C. L. Lee and Y. Pan, "An efficient watermarking method based on significant difference of wavelet coefficient quantization," *IEEE Trans. Multimed.*, vol. 10, no. 5, pp. 746–757, 2008. doi: 10.1109/TMM.2008.922795.

[6] Z. Zhou, H. Sun, R. Harit, X. Chen, and X. Sun, "Coverless image steganography without embedding," in *Cloud Comput. Secur.: First Int. Conf., ICCCS 2015*, Nanjing, China, Springer, Aug. 13–15, 2015. doi: 10.1007/978-3-319-27051-7_11.

[7] S. Zheng, L. Wang, B. Ling, and D. Hu, "Coverless information hiding based on robust image hashing," in *Intell. Comput. Methodol.: 13th Int. Conf., ICIC 2017*, Liverpool, UK, Springer, Aug. 7–10, 2017. doi: 10.1007/978-3-319-63315-2_47.

[8] L. Zou, J. Sun, G. Min, W. Wan, and B. B. Gupta, "A novel coverless information hiding method based on the average pixel value of the sub-images," *Multimed. Tools Appl.*, vol. 78, pp. 7965–7980, 2019. doi: 10.1007/s11042-018-6444-0.

[9] X. Zhang, F. Peng, and M. Long, "Robust coverless image steganography based on DCT and LDA topic classification," *IEEE Trans. Multimed.*, vol. 20, no. 12, pp. 3223–3238, Dec. 2018. doi: 10.1109/TMM.2018.2838334.

[10] Q. Liu, X. Xiang, J. Qin, Y. Tan, and Y. Luo, "Coverless steganography based on image retrieval of densenet features and dwt sequence mapping," *Knowl.-Based Syst.*, vol. 192, 2019, Art. no. 105375. doi: 10.1016/j.knosys.2019.105375.

[11] L. L. Tan, J. J. Liu, Y. Zhou, and R. R. Chen, "Coverless steganography based on low similarity feature selection in DCT domain," *Radioengineering*, vol. 32, no. 4, pp. 603–615, 2023. doi: 10.13164/re.2023.0603.

[12] F. Jiang, L. Dong, K. Wang, K. Yang, and C. Pan, "Distributed resource scheduling for large-scale MEC systems: A multiagent ensemble deep reinforcement learning with imitation acceleration," *IEEE Internet Things J.*, vol. 9, no. 9, pp. 6597–6610, May 1, 2022. doi: 10.1109/JIOT.2021.3113872.

[13] Y. Luo, J. Qin, X. Xiang, and Y. Tan, "Coverless image steganography based on multi-object recognition," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 7, pp. 2779–2791, Jul. 2021. doi: 10.1109/TCSVT.2020.3033945.

[14] Q. Liu, X. Xiang, J. Qin, Y. Tan, and Y. Qiu, "Coverless image steganography based on densenet feature mapping," *EURASIP J. Image Video Process.*, vol. 2020, no. 1, 2020, Art. no. 39. doi: 10.1186/s13640-020-00521-7.

[15] G. Huang, Z. Liu, V. Laurens, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 4700–4708. doi: 10.1109/CVPR.2017.243.

[16] Y. J. Luo, J. Qin, X. Xiang, Y. Tan, and N. N. Xiong, "Coverless image steganography based on image segmentation," *Comput. Mater. Contin.*, vol. 64, no. 2, pp. 1281–1295, 2020. doi: 10.32604/cmc.2020.010867.
[17] Q. Liu, X. Xiang, J. Qin, Y. Tan, and Q. Zhang, "A robust coverless steganography scheme using camouflage image," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 6, pp. 4038–4051, Jun. 2022. doi: 10.1109/TCSVT.2021.3108772.
[18] Y. Tan, X. Xiang, J. Qin, and Y. Tan, "Robust coverless image steganography based on human pose estimation," *Knowl.-Based Syst.*, vol. 296, 2024, Art. no. 111873. doi: 10.1016/j.knosys.2024.111873.
[19] I. Goodfellow *et al.*, "Generative adversarial nets," *Neural Inform. Process. Syst.*, vol. 27, pp. 2672–2680, 2014.
[20] M. M. Liu, M. Q. Zhang, J. Liu, Y. N. Zhang, and Y. Ke, "Coverless information hiding based on generative adversarial networks," 2017, *arXiv:1712.06951*.
[21] S. Zhang, S. Su, L. Li, Q. Zhou, and C. C. Chang, "An image style transfer network using multilevel noise encoding and its application in coverless steganography," *Symmetry*, vol. 11, no. 9, 2019, Art. no. 1152. doi: 10.3390/sym11091152.
[22] X. Chen, Z. Zhang, A. Qiu, Z. Xia, and N. Xiong, "Novel coverless steganography method based on image selection and stargan," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 1, pp. 219–230, Jan.–Feb. 1, 2022. doi: 10.1109/TNSE.2020.3041529.
[23] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004. doi: 10.1023/B:VISI.0000029664.99615.94.
[24] Y. Choi, M. Choi, M. Kim, J. W. Ha, and J. Choo, "StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 8789–8797. doi: 10.1109/CVPR.2018.00916.
[25] R. Xue and Y. Wang, "Message drives image: A coverless image steganography framework using multi-domain image translation," in *2021 Int. Joint Conf. Neural Netw. (IJCNN)*, Shenzhen, China, 2021, pp. 1–9. doi: 10.1109/IJCNN52387.2021.9534043.
[26] F. Peng, G. Chen, and M. Long, "A robust coverless steganography based on generative adversarial networks and gradient descent approximation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 9, pp. 5817–5829, Sep. 2022. doi: 10.1109/TCSVT.2022.3161419.
[27] W. Wen, H. Huang, S. Qi, Y. Zhang, and Y. Fang, "Joint coverless steganography and image transformation for covert communication of secret messages," *IEEE Trans. Netw. Sci. Eng.*, vol. 11, no. 3, pp. 2951–2962, May–Jun. 2024. doi: 10.1109/TNSE.2024.3354941.
[28] H. A. Rehman *et al.*, "Leveraging coverless image steganography to hide secret information by generating anime characters using GAN," *Expert. Syst. Appl.*, vol. 248, 2024, Art. no. 123420. doi: 10.1016/j.eswa.2024.123420.
[29] D. R. I. M. Setiadi, S. Rustad, P. N. Andono, and G. F. Shidik, "Digital image steganography survey and investigation (goal, assessment, method, development, and dataset)," *Signal Process*, vol. 206, 2023, Art. no. 108908. doi: 10.1016/j.sigpro.2022.108908.
[30] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun ACM*, vol. 60, no. 6, pp. 84–90, 2017. doi: 10.1145/3065386.
[31] A. N. Gorban, E. M. Mirkes, and I. Y. Tyukin, "How deep should be the depth of convolutional neural networks: A backyard dog case study," *Cognit. Comput.*, vol. 12, no. 2, pp. 388–397, 2020. doi: 10.1007/s12559-019-09667-7.
[32] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye and D. Ren, "Distance-IoU loss: Faster and better learning for bounding box regression," *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 7, pp. 12993–13000, Apr. 2020. doi: 10.1609/aaai.v34i07.6999.
[33] M. Zhang and L. Yin, "Solar cell surface defect detection based on improved YOLO v5," *IEEE Access*, vol. 10, pp. 80804–80815, 2022. doi: 10.1109/ACCESS.2022.3195901.
[34] Z. Zhou, Y. Cao, and M. Wang, "Faster-RCNN based robust coverless information hiding system in cloud environment," *IEEE Access*, vol. 7, pp. 179891–179897, 2019. doi: 10.1109/ACCESS.2019.2955990.