Data Hiding with Deep Learning: A Survey Unifying Digital Watermarking and Steganography

OLIVIA BYRNES, The University of Adelaide, Australia
WENDY LA, The University of Adelaide, Australia
HU WANG, The University of Adelaide, Australia
CONGBO MA, The University of Adelaide, Australia
MINHUI XUE, The University of Adelaide, Australia
QI WU, The University of Adelaide, Australia

Data hiding is the process of embedding information into a noise-tolerant signal such as a piece of audio, video, or image. Digital watermarking is a form of data hiding where identifying data is robustly embedded so that it can resist tampering and be used to identify the original owners of the media. Steganography, another form of data hiding, embeds data for the purpose of secure and secret communication. This survey summarises recent developments in deep learning techniques for data hiding for the purposes of watermarking and steganography, categorising them based on model architectures and noise injection methods. The objective functions, evaluation metrics, and datasets used for training these data hiding models are comprehensively summarised. Finally, we propose and discuss possible future directions for research into deep data hiding techniques.

CCS Concepts: • **Security and privacy**; • **Computing methodologies** → **Computer vision**;

Additional Key Words and Phrases: Data Hiding, Digital Watermarking, Steganography, Deep Neural Networks

1 INTRODUCTION

Data hiding is the process of hiding some form of information within another media. This could encompass everything from encoding a secret message into an existing piece of text, to embedding audio files into a digital image. As digital assets become more diverse and ubiquitous, the importance and variance in data hiding applications will only grow [47]. The data hiding process has become crucial in the modern era with the increasing prevalence of digital communication and multimedia data. Digital services require secure communication across all mediums, and digital intellectual property (IP) ought to be protected from theft and misuse.

Formally, data hiding can be separated into three categories: watermarking, steganography, and cryptography [72]. This survey concentrates on the former two — watermarking and steganography.

Digital watermarking employs data hiding techniques in order to embed some form of identification (ID) into a piece of media that communicates the owners of the intellectual property (IP). This way, if an adversary attempts to copy or modify the original piece of media, the ID can be extracted and the owners can thus be identified. The primary application of digital watermarking is for the authentication of digital assets, however, the process also has uses for licences and identification [30], digital forensics [9], and data protection in smart cities [22, 58]. Digital watermarking is not only useful for marking images and documents, but also for real-time audios and videos [33, 40], languages [1], as well as chip and hardware protection in electronics [63].

Steganography is similar to watermarking in that data is embedded into a piece of media. However, instead of being a form of ID denoting the creator of the artefacts, the data is a secret message. This secret message should be transmitted without being detected, intercepted, or decoded. Steganography differs from cryptography in that its main goal is to keep the format of the cover media readable and not distorted after hiding data within it [37]. The public should still be able to see what the cover media originally was without being able to detect the embedded messages.

2021. 0004-5411/2021/7-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

Steganography is applied in several industries including the medical, military, and multimedia fields, and is used wherever there is a need for secret communication with security purposes [43].

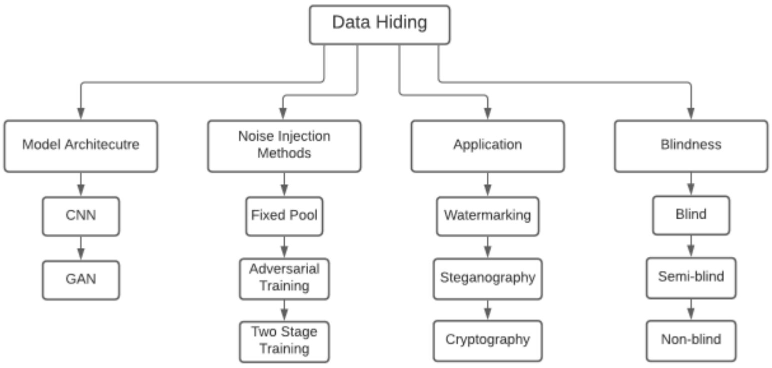

Fig. 1. A hierarchical diagram showing different methods for classifying deep learning-based data hiding techniques. Model architectures can be separated into CNN-based and GAN-based structures. Attack training refers to the method for organising and generating attacks during model training. Blindness refers to the functionality of the data hiding method, further explained in Section 2.

In the past, data hiding has been performed using specialised algorithms, which are classified based on the domain in which they operate — based on spatial or frequency. Spatial domain techniques directly embed data into the cover media by manipulating bit streams or pixel values. They are computationally simple compared to other techniques, and hence are more susceptible to removal and distortion from adversaries. Frequency domain techniques rely on the manipulation of frequency coefficients in the signal medium. These techniques achieve a higher degree of robustness to attacks, but are more computationally complex [47]. The basic idea is that high frequency areas of a signal are more suited for data embedding. Frequencies can be manipulated and partitioned into high/low areas in a number of different ways, after which the secret data is embedded into the lower frequency areas to optimise embedding quality. Spatial and frequency domain techniques can also be combined for a more robust strategy [47]. The combination of spatial and frequency-based methods is the basis of adaptive steganography methods. They can be applied when the senders and receivers of the data are aware of the attacker's detection mechanism, and so will modify their methods to prevent those attackers from corrupting or interfering with the message [13].

The drawback of these traditional algorithms is that their applications are narrow, and the creators of these algorithms require expert knowledge of the embedding process. Particular techniques are useful for certain limited tasks, and the growing sophistication of watermark removal and degradation attacks means that the effectiveness of these algorithms may be compromised in the near future [23, 44]. New advancements in the field of deep learning span many industries due to the strong representation abilities of deep neural networks. In the field of data hiding, deep learning models provide adaptable, generalised frameworks that can be used for a variety of applications in watermarking and steganography. Currently, most works in this area concentrate on image-based data hiding, and these are the works that will be compared in this survey. These machine learning models are able to learn advanced embedding patterns that are able to resist a much wider range of attacks with far more effectiveness than traditional watermarking or steganography algorithms [44]. With further research [1, 59, 85, 96], there is potential and highly critical to develop generalised frameworks for data hiding that can work with a range of cover media types to robustly embed data and provide highly secure content authentication and communication services. The advantage of the deep learning approach is that networks can be retrained to become resistant to new types of attacks, or to emphasise particular goals such as payload capacity or imperceptibility without creating specialised algorithms for each new application [96]. An additional advantage of deep data hiding techniques can enhance the security of the embedded messages. The high non-linearity of deep neural models makes it virtually impossible for an adversary to retrieve the embedded information [44]. Compared to traditional methods, deep learning-based methods are not only more secure and adaptable to different applications, but they also offer enhanced robustness to adversarial attacks and distortions. They are also able to achieve more imperceptible forms of data embedding.

The data hiding process, consisting of message embedding and extraction, maps intuitively onto the encoder-decoder network architecture, wherein the learning model is partitioned into two networks. In these models, an encoder network is trained to embed input messages to images. The images are then subjected to some forms of attack through distortion layers, and the decoder network must then extract the original message from the distorted image. These distortions can include blurring, cropping, compression, etc. The objective of the network training is to minimise an objective function, which accounts for the differences between the cover image and encoded image, as well as the differences between the embedded and extracted input message. The first papers exploring the capabilities of neural network technology for data hiding were released in 2017, and were based on convolutional neural networks [6, 44]. In recent years, GAN-based approaches have gained traction, popularised by the HiDDeN model [96], which was the first end-to-end trainable model for digital watermarking and steganography. These deep learning models employ different message embedding strategies in order to improve robustness, such as using adversarial examples, attention masks, and channel coding. The continued development of deep learning-based data hiding models will greatly improve the effectiveness and security of digital IP protection, and secure secret communication.

Since this is a relatively new area of research, current surveys on data hiding primarily concentrate on traditional algorithms. There are existing works examining deep learning-based techniques for steganography and cryptography [38, 45], but there is a lack of works examining deep watermarking techniques. There is an existing survey looking at deep learning-based watermarking and steganography [89]; however, a comprehensive survey regarding deep data hiding models unifying digital watermarking and steganography is still lacking. To the best of our knowledge, ours is the first survey to examine deep learning techniques for deep learning-based digital watermarking and steganography that includes the largest range of recent works. As this research area continues to expand, it is important to summarise and review the current methods. The aim of this survey is to systematically categorise and discuss existing deep learning models for data hiding, separated based on applications in either watermarking or steganography, as well as presenting future directions that research may take. The key contributions of the survey are listed as follows:

- This survey systematically categorises and compares deep learning-based models for data hiding in either watermarking or steganography based on network architecture and noise injection methods.
- We comprehensively discuss and compare different objective strategies, evaluation metrics, and training datasets used in current state-of-the-art deep data hiding techniques.
- We also presents a wide range of future directions for deep learning-based data hiding research.

Paper Collation. In this survey, we have collected, analysed and discussed over 30 papers based on the search results of Google Scholar[1] and DBLP[2] with the keywords *data hiding, digital watermarking, steganography, deep neural networks, and generative adversarial networks (GANs).* Those papers were selected from a plethora of top-tier Security and Privacy, Computer Vision and Machine Learning conferences and journals.

[1]https://scholar.google.com/
[2]https://dblp.org/

Organization of the Survey. In the following sections, our survey will cover recent advanced deep learning based data hiding methods from two forms: digital watermarking and steganography. Section 2 gives the problem formulation of data hiding. Section 3 highlights the architecture of data hiding and offers a comprehensive review of deep learning based data hiding techniques. This survey also summarises noise injection techniques in Section 4, objective functions in Section 5, evaluation metrics in Section 6, and existing datasets in Section 7. Finally, open questions and future work for deep learning based data hiding task are discussed in Section 8. Section 9 concludes the paper.

2 PROBLEM FORMULATION OF DATA HIDING

When we evaluate the effectiveness of data hiding techniques, there are many factors that should be considered. The three most important are **capacity**, how much information can be embedded into the cover media, **imperceptibility**, how easy the data is to detect, and **robustness**, how resistant the data is to attacks. Here, attacks refer to any alterations made to the cover media with the intent to degrade or remove the embedded data. There is an implicit trade-off between these three aforementioned characteristics. For instance, if there is a high payload capacity, then the message will be easier to detect, resulting in a lower level of imperceptibility. Similarly, improving robustness against attacks can potentially decrease both payload capacity and imperceptibility, since there is added redundancy to the encoded image that allows it to resist distortions.

In digital watermarking, robustness is generally favoured over secrecy because the ability to resist attacks and distortions is more important than the watermark's imperceptibility. Conversely, in steganography, imperceptibility is favoured since the highest priority is that the message remains a secret. This relationship is illustrated in Figure 2. Due to the adaptable nature of deep learning-based approaches, the trade-off between these metrics can be explicitly controlled by the user, and the key properties of robustness and imperceptibility underpin the objective of the deep learning system.

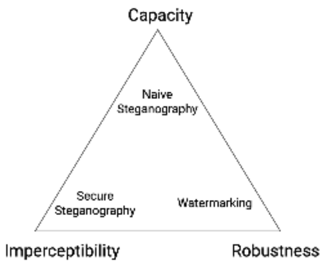

Fig. 2. A figure showing the trade-off between the three primary data hiding properties; robustness, imperceptibility, and capacity, as well as which data hiding applications favour each property over the others.

2.1 Digital Watermarking and Steganography Terminology

The basic watermarking process consists of an encoding and decoding process. The encoder E receives the cover media C and message to be hidden M, and outputs an the encoded media C' such that: $E(C, M) = C'$. Then the decoder receives the encoded media as input and extracts the message M', such that: $D(C') = M'$. In a robust implementation, M and M' should be as similar as possible in an effective strategy. Similarly, maximising the imperceptibility property is done by minimising the difference between C and C'. Types of data hiding can be classified based on a variety of properties as follows [47]:

- **Blindness:** related to how much information is required to extract the original watermark from the encoded media. **Blind** techniques do not require the original cover media or original

watermark to extract the watermark, and are the most practically useful. **Semi-blind** techniques require only the watermark and not the original cover media to extract the watermark from the encoded media. **Non-blind** techniques require both the cover media and watermark in order to extract the watermark.
- **Fragility:** related to how the watermark reacts to attacks and distortions applied to it. **Fragile** watermarks are designed to show all attacks applied to it so that, when extracted, it is possible to verify which attacks have been applied to the media. This is useful when verifying the integrity of the media. **Semi-fragile** watermarks are not robust against intentional distortions such as warping and noise filtering that attempt to degrade the watermark, but are robust against content-preserving distortions such as compression and enhancement. Therefore, it can be used to trace any illegal distortions made to the media.
- **Visibility:** whether the watermark is visible to the human eye.
- **Invertibility:** whether the watermark can be removed from the cover media once embedded. Invertible watermarks can be removed and non-invertible watermarks cannot.
- **Robustness**: the ability of the watermark to remain unchanged when attacks are applied to it. In practice, fragile techniques are often used in conjunction with robust techniques, so that one watermark will remain unaltered and the other can be used to trace the attacks applied to the media.
- **Security**: determines how difficult it is for an adversarial party to extract the data from the cover image.

3 DEEP LEARNING-BASED DATA HIDING TECHNIQUES

Deep learning-based data hiding models utilise the encoder-decoder network structure to train models to imperceptibly and robustly hide information. They present an advantage over traditional data hiding algorithms because they can be retrained to become resistant to a range of attacks, and be applied to different end-use scenarios. Deep learning methods negate the need for expert knowledge when crafting data hiding algorithms, and improve security due to the black-box nature of deep learning models.

The following section discusses deep learning-based data hiding techniques separated into techniques focused on watermarking and steganography. The detection and removal mechanisms are then discussed at the end of this section. The classification of deep learning-based data hiding techniques detailed in this section is outlined in Figure 3. It should be noted that CNNs incorporating adversarial training are different to GAN-based methods. Adversarial training in this instance refers to the use of trained CNNs for noise injection during the attack simulation stage, while GAN-based methods incorporate a discriminator to scrutinise encoded and cover images to improve embedding imperceptibility [60].

3.1 Introduction to Deep Learning-based Data Hiding Architectures

Currently, the majority of new watermarking models use an encoder-decoder architecture based on Convolutional Neural Networks (CNNs). A simple diagram showing the deep learning-based data hiding process can be found in Figure 4.

In these models, the encoder embeds data in a piece of cover media; this encoded media is subjected to attack simulation, and then the data is extracted by the decoder network. Through the iterative learning process, the embedding strategy becomes more resistant to the attacks applied during simulation, and the extraction process improves the integrity of the extracted data. The advantage of this technique over previous traditional algorithms is that they require no expert knowledge to program, and can simply be retrained for different applications and attack types instead of needing to be designed from scratch. The system exists as a black-box with high non-linearity where the intricacies of the embedding system are unknown and impossible to ascertain.

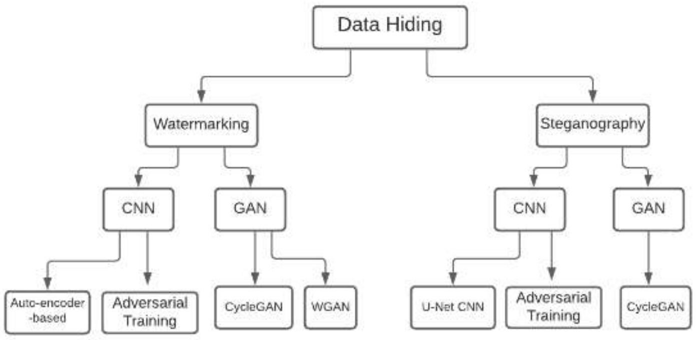

Fig. 3. A hierarchical diagram showing the classification of deep learning-based data hiding models presented in this survey. 'Adversarial training' refers to attack simulation during training, which includes noise-based attacks generated by a trained CNN.

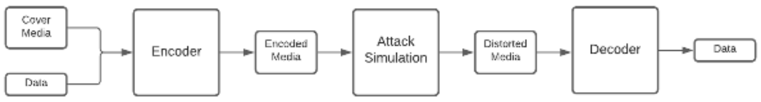

Fig. 4. A diagram showing a general encoder-decoder architecture for digital watermarking.

This makes deep learning-based methods highly secure, as well as adaptable to different end-use scenarios.

Some variations of the simple CNN encoder-decoder approach shown above include convolutional auto-encoders, and CNNs with adversarial training components. The U-Net CNN architecture is common in steganography applications due to image segmentation abilities.

Many models adopt the Generative Adversarial Network (GAN) structure [24]. The GAN framework consists of a generative model and a discriminative model. In deep data hiding, the discriminator network is given a mixture of encoded and unaltered images and must classify them as such. Throughout the learning process, the generative model improves in its data embedding capabilities, producing highly imperceptible examples, while the discriminative model improves at identifying encoded images. The end point of training is reached when the discriminator can only identify legitimately encoded images 50% of the time – it is making random guesses. The use of discriminative networks can greatly increase data imperceptibility, and is therefore useful for steganography as well as watermarking applications. A simple diagram of a GAN-based deep data hiding model can be found in Figure 5.

There are also further variations of the GAN framework, including Wasserstein GANs (WGANs) and CycleGANs. The CycleGAN architecture is useful for image-to-image translation, and includes two generative and two discriminative models. The primary benefit of CycleGANs is that the model can be trained without paired examples. Instead, the first generator generates images from domain A, and the second from domain B, where each generator takes an image from the other domain as input for the translation. Then, discriminator A takes as input both images from domain A and output images from generator A, and determines whether they are real or fake (and vice versa for discriminator B). The resulting architecture is highly useful for translating between images. A simple diagram of a CycleGAN framework used for data hiding can be found in Figure 6.

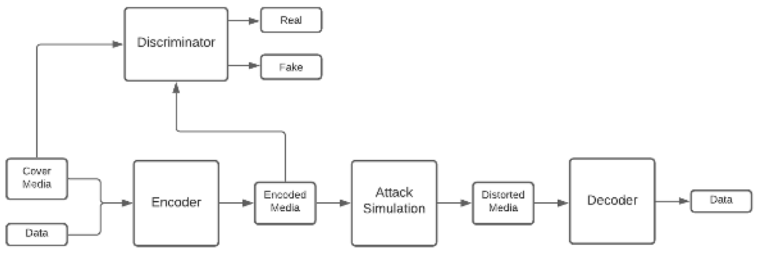

Fig. 5. A diagram showing the discriminator component added to the general encoder-decoder framework shown previously. The cover and encoded image are input to the discriminator, which then learns to distinguish between watermarked and non-watermarked instances.

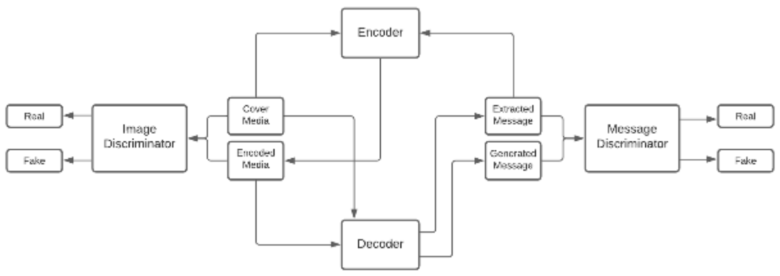

Fig. 6. A diagram showing the CycleGAN architecture adopted in [87]. The image discriminator receives encoded and unaltered images and must decide whether it has been encoded with a secret image, similar to the previous implementations. However, the decoding generative model receives both cover images and encoded images as input and attempts to extract a watermark. The 'generated message' is a watermark generated from an unaltered cover image. The message discriminator must decide whether the input messages are legitimate watermarks, or simply random generated images from the unaltered cover media.

3.2 Deep Learning-based Watermarking Techniques

In this section, current deep learning models for digital watermarking are categorised based on their network architecture design. The most important features of deep watermarking models is robustness, however, the adaptability of the framework is also important. Having controls to influence the trade-off between robustness and imperceptibility is hugely beneficial, since emphasising imperceptibility opens opportunities for applications in steganography.

3.2.1 Encoder-decoder Framework. Due to the encoding and decoding tasks central to the data hiding process, the encoder-decoder deep learning framework is well suited for data hiding models. The encoder and decoder networks incorporate CNNs, which are used in a variety of applications such as detection, recognition, and classification due to their unique capabilities in representing data with limited numbers of parameters. The layers in the CNN learn non-linear, complex feature sets, representing the inputs and outputs to the network using weight sharing mechanisms. The following deep watermarking models adopt the encoder-decoder framework without including a discriminator, which characterises GAN-based architectures [4, 44, 49, 60, 64, 95]. A simple diagram of the encoder-decoder deep watermarking structure can be found in Figure 4.

Auto-encoder based Model. The encoder-decoder architecture is a general framework and auto-encoder is a special case of the encoder-decoder structure. However, auto-encoder usually used in unsupervised-learning scenarios by reconstructing the inputs. The potential of the CNN-based encoder-decoder frameworks for digital image watermarking was first explored in [44], which uses two traditional Convolutional Auto-Encoders for watermark embedding and extraction. Auto-encoder based CNNs were chosen based on their uses in feature extraction and denoising in visual tasks, such as facial recognition and generation, and reconstructing handwritten digits. The intuition was that the auto-encoder CNN would be able to represent the watermark input in a form that was highly imperceptible when encoded within the cover image. This early CNN-based method applies two deep auto-encoders to rearrange the cover image pixels at the bit level to create a watermarked image. The technique was found to outperform the most robust traditional frequency domain methods in terms of both robustness and imperceptibility. Although promising for the future of deep watermarking, this technique is non-blind, and therefore not practically useful.

Robust and blind digital watermarking results can be achieved using a relatively shallow network, as was shown by WMNet [64]. The watermarking process is separated into three stages: watermark embedding, attack simulation, where the CNN adaptively captures the robust features of various attacks, and updating, where the model's weights are updated in order to minimise the loss function and thereby correctly extract the watermark message. Embedding is achieved by increasingly changing an image block to represent a watermark bit. The model is trained to extract watermark bits from the image blocks after attack simulations have been applied.

The back-propagation embedding technique utilised in WMNet [64] used only a single detector network, which was found to cause performance degradation if the gradient computation in the back propagation operation was affected by batch normalisation (BN). This deficiency was improved by adding an auto-encoder network as well as visual masking to allow flexible control of watermark visibility and robustness. The auto-encoder network was added to the encoder, and subsequently shorted time taken for both embedding and detection at the encoder because the feed-forward operation is generally much faster than back-propagation. These improvements were published in a follow-up paper [65].

Robustness Controls and Input Preproccessing. Subsequent works after the aformentioned early techniques in [44, 64] focused on generalising the watermarking process for multiple applications. Mechanisms such as robustness controls to influence the robustness/imperceptibility trade-off were introduced to gear models toward both watermarking and steganography applications, and mechanisms such as host and watermark adaptability were developed to pre-process inputs.

A blind and robust watermarking technique was achieved using the CNN-based system [95]. The aim of this model is to generalise the watermarking process by training a deep neural network to learn the general rules of watermark embedding and extraction so that it can be used for a range of applications and combat unexpected distortions. The network structure is characterised by an invariance layer that functions to tolerate distortions not seen during network training. This layer uses a regularisation term to achieve sparse neuron activation, which enhances watermark robustness and computational efficiency. The layer also includes a redundancy parameter that can be adjusted to increase levels of redundancy in the resulting image, giving the model a higher tolerance of errors and increasing robustness. The primary aim of these features is to generalise watermarking rules without succumbing to overfitting. Overfitting was tested by generating synthetic images as extreme examples not seen during training. These included blank images of intense red, green, and blue – the most difficult to watermark – and the model still showed good results under these conditions. The model was compared two auto-encoder CNN methods [44, 64], and was found to achieve greater robustness due to the new features it adopted.

ReDMark [4] uses two Full Convolutional Neural Networks (FCNs) for embedding and extraction along with a differentiable attack layer to simulate different distortions, creating an end-to-end training scheme. ReDMark is capable of learning many embedding patterns in different transform domains and can be trained for specific attacks, or against a range of attacks. The model also includes a diffusion mechanism based on circular convolutional layers, allowing watermark data to be diffused across a wide area of an image rather than being confined to one image block. This improves robustness against heavy attacks, because if one image block is cropped out or corrupted, the watermark can still be recovered. The trade-off between robustness and imperceptibility can be controlled via a strength factor that can influence the pattern strength of the embedding network.

The watermarking model developed by Lee et al. [49] uses a simple CNN for both embedding and extraction, without using any resolution-dependent layers. This allows for host image resolution adaptability – meaning that images of any resolution can be used as input to the system to be watermarked. There is an image pre-processing network that can adapt images of any resolution for the watermarking process. There is also watermark pre-processing, meaning the system can handle user-defined watermark data. This is achieved by using random binary data as the watermark that is updated at each iteration of training. The model also adopts a strength scaling factor, which allows for controllability of the trade-off between robustness and imperceptibility. The method showed comparable, if not better, performance compared to ReDMark [4], and two generative adversarial-based models [55, 96].

Adversarial Training. A further improvement to the CNN-based encoder-decoder framework was made by adopting trained CNNs for attack simulation. While many other works use a fixed pool of attacks or a differentiable attack layer, using a trained CNN to generate attacks can greatly improve robustness, and introduces an adversarial component to model training. The focus of the Distortion Agnostic (DA) model [60] was to directly improve the HiDDeN model [96], primarily through adding robustness to the watermarking system in situations where the model is trained on a combination of distortions rather than one predetermined type. Instead of explicitly modelling different distortions during training from a fixed pool, the distortions are generated via adversarial training by a trained CNN. This technique was found to perform better in terms of robustness than HiDDeN [96] when distortions not seen during training were applied to images. The DA framework also incorporates channel coding, a means of detecting and correcting errors during a signal transmission, to add an additional layer of robustness to images by injecting extra redundancy. The watermark message is initially fed through the channel encoder to add redundancy before being input to the encoder model. Similarly, prior to extraction the redundant watermark message is input to a channel decoder to retrieve the final message.

3.2.2 Generative Adversarial Networks. The second primary approach for deep watermarking uses GANs, building upon the aformentioned techniques. Current models that adopt the GAN framework include [55, 83, 87, 90, 96]. Many of the following models also use CNNs within their network architecture, but their use of the generative and discriminative components of the GAN framework set them apart from the aforementioned implementations. A simple diagram of the GAN framework used for data hiding can be found in Figure 5.

The first end-to-end trainable framework for data hiding was a model called HiDDeN [96], which uses an adversarial discriminator to improve performance. It was a highly influential paper that has informed the development of deep watermarking models since its release. The model consists of an encoder network, trained to embed an encoded bit string in a cover image whilst minimising perceptual perturbations, a decoder network, which receives the encoded image and attempts to extract the information, and an adversary network, which predicts whether or not an image has been encoded.

HiDDeN [96] uses a novel embedding strategy based on adversarial examples. When neural networks classify image examples to a particular target class, invisible perturbations in the image can fool the network into misclassifying that example [25]. These perturbations have been shown to remain preserved when exposed to a variety of image transformations [48]. Adversarial examples are ordinarily a deficiency in neural networks, since they reduce classification accuracy. However, since meaningful information can be extracted from imperceptible image perturbations, it was theorised that in a similar fashion meaningful information could be encoded in adversarial distortions and used as a watermark embedding strategy. This embedding technique, paired with the GAN framework, is able to achieve a higher payload capacity (measured in bits per pixel) than other common data hiding mechanisms such as Highly Undetectable Steganography (HUGO) [27], Wavelet Obtained Weights (WOW) [31], and S-UNIWARD [32]. One drawback of HiDDeN [96] was that loss between encoded and decoded message was minimised when trained only on a specific kind of attack compared to a combination of different attack types. This shows that the model is best when trained specifically to combat one type of attack, but not as effective when trained on a variety.

This shortcoming was improved in the model ROMark [83], which builds upon the framework from HiDDeN [96] by using a min-max formulation for robust optimisation. This was done by addressing two main goals; first, to obtain the worst-case watermarked images with the largest decoding error, and second, to optimise the model's parameters when dealing with the worst-case scenario so that decoding loss is minimised. The idea of this technique is to minimise decoding loss across a range of attacks, rather than training the model to resist specialised attacks, creating a more versatile and adaptable framework. Due to the optimisation for worst-case distortions in ROMark [83] , it performed better when trained on a combination of attacks, particularly on those that had not been seen during training. ROMark [83] was also more robust in some specialised attack categories, though HiDDeN [96] had higher accuracy in these categories.

Additional improvements were made to the framework from HiDDeN [96] by Hamamoto et al. [28]. This work uses a neural network for attack simulation rather than a single differentiable noise layer. There is a rotation layer followed by an additive noise layer, allowing the model learn robustness against geometric rotation attacks. It also features a noise strength factor to control the robustness/imperceptibility trade-off. It was tested against HiDDeN [96] and found to achieve greater image quality after watermark embedding, as well as greater robustness against JPEG compression. This model was the first to be trained to meet the Information Hiding Criteria (IHC) for robustness while simultaneously resisting geometric rotation attacks. It is suggested for future work to combine the architecture with a Scale Invariant Feature Transform (SIFT) detector [57], which is able to detect image features that are robust for embedding to withstand geometric attacks in traditional watermarking algorithms.

A novel embedding strategy using Inverse Gradient Attention (IGA) [90] was adopted recently. Similar to Attention-based Data Hiding [87], the focus is on identifying robust pixels for data hiding by using an attention mask. In the IGA method, the attention mask indicates the gradient values of the input cover image, which shows the robustness of each pixel for message reconstruction. It builds on the idea of adversarial examples first used for digital watermarking in HiDDeN [96], and uses the attention mechanism to locate the worst-case pixels for perturbation. By identifying robust pixel regions, this method further improves the payload capacity and robustness of watermarked images in tests against the following papers [60, 91, 96].

A novel two-stage separable deep learning (TSDL) framework for watermarking is introduced by Liu et al. [55], which addresses the problems with one-stage end-to-end training (OET) such as slow convergence leading to image quality degradation, and having to simulate noise attacks using a differentiable layer. Instead of relying on differentiable approximations, the TDSL framework can use true non-differentiable noise attacks such as JPEG compression during training. This is because, in OET models, the encoder and decoder are trained using a differentiable noise layer, which means the noise must support back propagation. The TDSL framework consists of noise free end-to-end adversary training (FEAT), which is used to train an encoder that autonomously encodes watermarks with redundancy and without reference to any noise. Noise aware decoder only training (ADOT) is used to train the decoder so that it is robust and able to extract the watermark from any type of noise attack. The model was tested against HiDDeN [96] and ReDMark [4], and found to achieve superior robustness for all but one attack category (crop). However, the two-stage training method is also robust against black-box noise attacks that are encapsulated in image processing software, which have not been tested in previous works.

Wasserstein GAN. A popular variation on the traditional GAN technique is the Wassertsein GAN (WGAN) [5]. This technique improves the model stability during training, as well as decreasing sensitivity of the training process to model architecture and hyperparameter configurations. The WGAN framework also provides a loss function that correlates with the quality of generated images. This is particularly useful for watermarking and steganography in the image domain, since image quality must be effectively optimised. Instead of the discriminator component of the network, WGANs include a critic. Rather than predicting the probability that a given image is real or fake, as is the discriminator's goal, the critic outputs a score denoting the 'realness' of the input image. In a data hiding scenario, the encoder's aim is to maximise the score given by the critic for real instances – which corresponds to an encoded image. Current papers adopting the WGAN framework for their watermarking models include [67, 68, 81, 91].

Zhang et al. [91] introduced SteganoGAN. Three variants of encoder architecture are explored, each with different connectivity patterns. The basic variant applies two convolution blocks, where the encoded image is the output of the second block. Residual connections have been shown to improve model stability [29], and in the residual variant, the cover image is added to the encoder outputs so that it learns to produce a residual image. The third variant uses a method inspired by DenseNet [34], in which there are additional feature connections between convolutional blocks that allow the feature maps generated by earlier blocks to be concatenated to those generated by subsequent blocks.

Using adversarial training, this model achieves a relative payload of 4.4 BPP, 10 times higher than competing deep learning methods. Although both works [91, 96] have mechanisms in place for handling arbitrarily-sized cover images as input, the higher payload capabilities of [91] means it can support a greater range of watermark data. The paper also proposes a new metric, Reed Solomon Bits Per Pixel (RS-BPP), to measure the payload capacity of deep learning-based data hiding techniques so that results can be compared with traditional data hiding methods. Although the primary focus for SteganoGAN is steganography, the high payload capacity and low detection rate of SteganoGAN produced images can also be applied to watermarking.

Two further works by the same authors also use the WGAN framework. Plata et al. introduced a new embedding technique where the watermark is spread over the spatial domain of the image [67]. The watermark message is converted into a sequence of tuples, where the first element of each is converted to a binary representation. The spatial message is created by randomly assigning binary converted tuples to sections of the message, including redundant data for increased robustness. The paper also introduces a new technique for differentiable noise approximation of non-differentiable distortions which allows simulation of subsampling attacks. The attack training pool is expanded from previous works to include subsampling and resizing attacks, and this wide range of attacks used during training increases general robustness. However, the spatial spread embedding technique reduces the embedding capacity, so is only useful for applications where capacity is not a priority. Furthermore, the training framework proposed requires half as much time as prior methods [4, 60, 96]. The authors expand upon this work in a follow-up paper [68], which introduces the double discriminator-detector architecture. The discriminator is placed after the noise layer, and receives both noised cover images and noised encoded images, and thus the discriminator learns to distinguished watermarked and non-watermarked images with attacks already applied. In practical contexts, this is useful because it reduces the likelihood of false accusations being made of IP theft. If the image has already been attacked and must be proven to contain a watermark, this training technique is useful. Crucially, it does not degrade the overall robustness of the encoded images.

A technique for encoded image quality improvement is introduced by Wang et al. [81] based on texture analysis. The cover image texture features are analysed by a grey co-occurence matrix which divides the image into complex and flat regions. The paper utilises the StegaStamp network [78] for embedding the watermark in the flat texture regions. This reduces the degree of image modification and improves the quality, and hence imperceptibility, of encoded images. The network in StegaStamp [78] produces higher quality images from low contrast examples; therefore the contrast value is used to calculate the texture complexity of the image.

CycleGAN. A further variation on the original GAN framework is CycleGAN [97]. The architecture is suitable for image-to-image translation tasks, and includes two generative and two discriminative models. A diagram can be found in Figure 6. As of writing, the only watermarking model that uses the CycleGAN framework is [87].

Attention-based Data Hiding (ADBH) [87] introduced an attention mechanism that helps the generative encoder pinpoint areas on the cover image best suited for data embedding. The attention model is an input processing technique that allows the network to focus on specific aspects of a complex input one at a time. The attention model generates an attention mask, which represents the attention sensitiveness of each pixel in the cover image. The value is regularised to a probability denoting whether altering the corresponding pixel will lead to obvious perceptual differences in the image that will be picked up by the discriminator. This improves the embedding process of the encoder network.

Rather than hiding a binary watermark message, this technique hides a secret image in the cover image. In the adopted CycleGAN framework, there is a target image generative model and a secret image generative model. The former generates watermarked images which are then fed to the cover image discriminative model, and builds on the PatchGAN model to operate on one image patch at a time [39]. Similarly, the secret image generative model serves as input for the secret image discriminative model. This architecture uses adversarial learning for both the embedding and extraction processes, ensuring that the distributions between both the cover and encoded images, and the encoded and extracted watermarks, are indistinguishable. The extractor network also operates on unmarked cover images, and inconsistent loss is used to make sure that, when this happens, there should be no correlation between the extracted data and the original watermark.

The three central contributions of this paper – the attention model, the use of cycle discriminative models, and the extra inconsistent loss objective – were all tested in isolation and found to improve the performance of the model overall in terms of both image quality and robustness.

3.2.3 Discussion. From the state-of-the-art deep learning methods discussed above, it seems that the GAN framework is the most promising in terms of robustness and secrecy optimisation due to the inclusion of adversarial loss in the objective calculations. To illustrate this, in HiDDeN [96], tests were conducted without including the adversary network and found to include perceptible alterations, whereas including the adversary greatly improved performance to produce an invisible watermarking technique. In tests comparing robustness, GAN-based models performed well, but were improved by techniques to target robust pixels for watermark embedding, such as the attention mechanisms used in Attention-based Data Hiding [87] and the IGA method [90].

It is also important for models to be robust against a range of attacks. Papers such as those by Hamamoto et al. [28] and Plata et al. [67] incorporate geometric rotation techniques, and subsampling and resizing attacks respectively. Having a wider range of attack types during training increases general robustness. Additionally, using true non-differentiable noise, as shown in the two stage training technique developed by Liu et al. [55], provides better results for JPEG compression than models that use differentiable approximations. Using a trained CNN to generate attacks, Distortion Agnostic Watermarking [60] is also a promising approach for diversifying the attacks encountered during training.

To produce an adaptable, generalised framework, it is important to include features such as host and watermark resolution adaptability, as well as robustness controls to influence the robustness/imperceptibility trade-off. Robustness controls make these models suitable for both watermarking and stenography, which relies more heavily on imperceptibility. The ideal digital watermarking model would include pre-processing networks for any watermark data or cover image to be input,

Table 1. Summary table of deep learning-based digital watermarking methods. The following abbreviations are used: AE (auto-encoder), RB (residual block), AP (average pooling), ASP (adaptive spatial pooling), AL (adversarial loss), CB (convolutional block), FC (fully connected), IG (inverse gradient), GP (global pooling), MP (max pooling), CC (circular convolutional).

Arch	Model	Domain	Embedding Network	Extractor Network	RA	RC	Remarks
AE CNN	[44]	frequency	AE CNN	AE CNN			First CNN for watermarking. Blind technique
	[65]	spatial	AE and RB	RB		✓	Autoencoder + visual mask
CNN	[60]	spatial	channel coding and CNN	channel coding and CNN			Channel coding and CNN for attacks
	[4]	frequency	CC layer and DCT layer	DCT layer	✓	✓	Diffusion mechanism
	[95]	spatial	CNN	CNN	✓	✓	Invariance layer and image fusion
	[49]	spatial	CNN and AP	CNN	✓	✓	Robustness against geometric attacks
GAN	[96]	spatial	AL	GP and CNN			Adversarial examples for embedding
	[90]	frequency	IG attention mask and CNN	CNN			IG mask improves capacity and robustness
	[83]	spatial	AL	FC layer		✓	Min-max formulation for robust optimisation
	[55]	spatial	AL	FC layer	✓	✓	Two stage training
	[28]	spatial	CNN	CNN + FC layers		✓	Robust against rotation and JPEG compression
WGAN	[91]	spatial	CB and ASP	CNN			High payload capacity
	[67]	spatial	CNN	CNN + AP			spatial spread embedding technique
	[68]	spatial	CNN	CNN + AP			double discriminator-detector
	[81]	spatial	CNN and MP	CNN and MP		✓	Uses texture analysis
CycleGAN	[87]	spatial	attention mask and CNN	CNN		✓	Uses images as watermark

as the framework was developed by Lee et al. [49], while also taking advantage of an adversarial discriminator or critic. The techniques used in [83] to improve robustness by obtaining and optimising parameters for ROMark worst-case examples is also a promising technique for improving robustness.

If a future model could combine these features; attention mechanisms to improve embedding, robustness against geometric attacks, host and watermark resolution controls, robustness controls, and handling worst-case distortions, it could result in a highly robust and adaptable framework. However, the added overhead of all these added features could be an issue in practice.

Table 1 shows a summary of the deep watermarking models reviewed in the above section. It shows whether the embedding strategy operates in the spatial or frequency domain, the nature of the watermark embedding and extracting networks, whether the technique supports host resolution adaptability (so that any cover image resolution can be used) **(RA)**, and whether it includes controls for influencing the trade-off between imperceptibility and robustness **(RC)**. The remarks column describes any important information or novel contributions of the paper, and the App (application) column to denote either watermarking (W) or steganography (S). Although many frameworks can be used for both applications, we denote the main focus presented in each paper.

3.3 Deep Learning-based Steganography Techniques

This section classifies deep steganography techniques based on model architecture. Most techniques adopt the encoder-decoder structure shown in Figure 4, and are based on convolutional neural networks (CNNs). Other approaches incorporate an adversarial component, using the Generative Adversarial Network (GAN) structure (see Figure 5). A variation of this structure is the CycleGAN framework (see Figure 6). Many steganography implementations differ from watermarking implementations in their use of the U-Net structure for image segmentation [73].

3.3.1 Encoder-decoder Framework. CNN-based encoder-decoder structures have been adopted in the following papers [6, 15, 20, 62, 76, 78, 84, 88]. A simple diagram of the encoder-decoder deep watermarking structure can be found in Figure 4. A paper from Google research [6], published in 2017, presented a deep steganography technique for hiding images inside other images. The structure contains three components in total. The first being the *prep network*, which serves two purposes; to adjust the size of a smaller secret image to fit into the cover image, and to transform the colour-based pixels to recognisable features to be encoded into the cover image. The second layer of the encoder is called the *hiding network*. As the name suggests, this layer creates the final stego-image. The final layer is the *reveal network* which is used to decode the output from the second layer. The experiments in the work [6] were mainly conducted to show that it was possible to completely encode a large amount of data with limited visual disturbances in the cover media. However, such a technique lacked robustness, security, and was not of high quality. It was possible for attackers to recover both the cover and secret image with a trained network.

Another image encoding technique called StegNet [84] was released in 2018. StegNet used structures from both auto-encoders and GANs to setup the encoding network. The cover image and the secret image were concatenated by a channel prior to the CNN encoding structure. Variance loss was included in loss calculations for the encoder and decoder. It was found that including the variance loss helped the neural network distribute the loss throughout the image rather than having concentrated areas of perceptual loss, improving the overall imperceptibility of embedding. The presented technique was highly robust against statistical analysis and when used against StegExpose [10], a commonly used steganalysis tool, it was also resistant against those attacks. Although robust, there were still some limitations to this method. Secret images and cover images must match in size and noise is still somewhat prominent in smoother regions.

Comparing the method of Baluja et al. [6] and StegNet [84], there has been a vast improvement in image hiding. The inclusion of components adapted from GANs and auto-encoders in StegNet [84] increased the robustness of the stego-images and was therefore more resistant to StegExpose [10]. Though StegNet [84] is quite robust, it is still lacking in some areas such as quality, image size restrictions and noise.

A faster R-CNN (region based CNN) method was introduced in [62]. Firstly, the cover image is passed through a region proposal network, which makes the selection for feature extraction faster. Softmax loss is used to box these regions and then specific existing steganographic algorithms are selected and assigned to the boxed regions. Since [62] uses a technique of selecting different steganography algorithms, using a combination of HUGO [27], S-UNIWARD [32] and WOW [31] algorithms, it is able to achieve highly imperceptible embedding. Being being able to select effective areas on the cover image also allows Meng et al. [62] to maintain a high level of robustness.

The adaptation the fusion technique [62] allows for minimal distortion in the extracted stego-image. When looking at StegNet [84], there is concern for noise in smoother areas of the cover photo. Although [84] has been shown to be robust, it could be further improved in this area with the box selection the fusion method [62] has. This does lead to a capacity dilemma if there was a method combining both StegNet [84] and the fusion method [62], since the latter would naturally be using less cover image area and therefore have a smaller capacity compared to StegNet [84].

A unique approach for steganography is shown by Sharma et al. in [76]. In this paper, both the encoder and decoder consist of two layers. In the first layer, the *prep layer*, smaller images are increased in size to correctly fit the cover image, there is a reconstruction of colour-based pixels to create more useful features, and the pixels are scrambled and then permutated. The second layer, the *hiding layer*, produces the stego-image with the output of first later and the cover image inputted. The decoder consists of the *reveal layer* and the *decrypt layer*. The *reveal layer* removes the cover image and the *decrypt layer* decrypts the output of the *reveal layer*. The advantage of this technique is that the first layer and its encryption method are similar to cryptography practises, allowing for a more secure embedded stego-image. Even when the cover media is known to the attacker, it is far more secure and difficult for the attacker to decode the secret image. This technique can also be applied to audio.

A reversible image hiding method was introduced by Chang et al. [15]. The structure relies on a concept called long short term memory. To encode, the cover image goes through a neural network in order to get a prediction, called the reference image. By subtracting the cover image from the reference image, the cover residuals (prediction errors) are calculated. Using histogram shifting (HS) on the cover residuals then produces stego residuals, along with an overflow map that is later used for the decoder. The stego-image is then created by adding the stego residual to the reference image. Where there is a pixel intensity flow, the overflow map is pre-calculated to flag these pixels. The decoder is essentially the reverse of the encoder, where the stego-image goes through a neural network to get a reference image, and the rest follows in reverse. This technique creates high quality, high capacity images that contain minimal noise. Its invertible feature to recover the cover image is unique to many other CNN based steganography.

Tang et al. [79] produced a steganography technique that uses adversarial embedding, called ADV-EMB. The network is able to hide a stego-message while being able to fool a CNN based steganalyser. The encoding network consists of a distortion minimalisation framework which adjusts and minimises the costs of the image according to features (gradients back propagation) from the CNN steganalyser. The focus of ADV-EMB [79] is to prevent steganalysers from being able to detect the stego-image. This shows in their results, with a high security rate and also increased imperceptibility. They are also able to train the system to counter unknown steganalysers by using a local well-performing CNN steganalyser as the target analyser, allowing for diverse applications. Although ADV-EMB [79] is able to decrease the effectiveness of adversary-aware steganalysers, it has a weak pixel domain. However, an increase in payload capacity would increase the detection rate of steganalysers.

U-Net CNN. U-Net CNNs are used to facilitate more nuanced feature mapping through image segmentation. This is useful in image steganography applications because a cover image can be broken up into distinct segments based on a certain property (for example, [20] uses a heatmap to score suitable embedding areas).

A U-Net CNN technique for reversible steganography was developed by Ni et al. [20]. This technique is capable of directly encoding the secret image into the cover image by concatenating the secret image into a six channel tensor. Its decoder is formed from six convolution layers, each of which is followed by a batch normalisation and a ReLu activation layer. The embedding technique is not easily effected by excessively high or low frequency areas. It is also able to produce a high-quality image with a capacity that is in general better than other cover-selection and cover-synthesis based steganography techniques. Even with its high capacity capabilities, like other steganographic technqiues that do not focus on robustness, too high of an embedding rate will increase the distortion rate more dramatically compared to robustness based steganography. StegaStamp [78] has a different implementation to most of the papers surveyed so far. It was designed to embed hyperlinks into images that can then be scanned either on a screen or physically. The encoder of this uses the same idea as Ni et al. in [20], with its U-Net structure. The technique uses four channels with a 400 x 400 pixel input, utilising a dense CNN structure for the decoder. The method is able to produce good quality images for messages up to 100 bits with a 95% accuracy rate for recovery. Experiments showed that embedding was robust against different printing machines, screens, and cameras used to display the encoded image. Although robust, distortions can still be perceptible in large low frequency regions of the image.

Universal Deep Hiding (UDH) is a model proposed for uses in digital watermarking, steganography, and light field messaging [88]. The encoder uses the simplified U-Net from Cycle-GAN [97], and a dense CNN for the decoder. The encoder hides the image in a cover-agnostic manner, meaning that it is not dependent on the cover image. UDH is an effective method due to the high frequency discrepancy between the encoded image and cover image. This discrepancy makes embedding robust in low frequency cover images. UDH is also less sensitive to pixel intensity shift on the cover image. The UDH method was compared with cover dependent deep hiding (DDH). Since the encoding of the secret image was independent on the cover image, there was not a method to adapt the encoding mechanism according to the cover image. The cover image may have some smoother areas which may not be ideal to embed data into, but with UDH there was no method of finding out this type of information. UDH is also unable to work well with severe uniform random noise.

CNN with Adversarial Training. The inclusion of adversarial attack networks during training can be used to help improve against stegalaysis and promote robustness. Adversarial training helps the system distinguish small perturbations that an untrained steganography method may bypass. This is an important feature to have in steganography, since its main focus is to protect message security. Chen et al. developed a model that incorporates a trained CNN-based attack network that generates distortions [16], similar to the technique used in the watermarking framework used by Xiyang et al. in Distortion Agnostic Watermarking [60]. The encoding structure is based off of a simple model that hides a secret grey-scale image into channel B (blue) of a coloured cover image, where both must have the same resolution. From this basic model, the paper was able to add two other enhanced models, a secure model and a secure robust model. The secure model inserts a steganalysis network into the basic model where its goal is to increase security against steganalysis. The secure and robust model uses the secure model and inserts an attack network to increase the robustness of the system. The separation of each model allows the framework by Chen et al. [16] to be used in several different scenarios, allowing the user to adjust to their needs. A downside is that users are unable to send RGB pictures and are limited to just grey-scale images.

The secure model was shown to have the best invisibility against all models and was still the best when compared to the following methods [6, 71]. The secure and robust model did not perform as well as the secure model but was still able to improve. The basic model and the secure model showed increased visual results this comparison, with the secure model having the best visual results. The visual results of the secure and robust model was similar to ISGAN [93].

3.3.2 Generative Adversarial Networks based models. Generative Adversarial Networks (GANs) are used extensively in deep steganography. Various new structures have allowed the simple GAN structure to be improved, increasing the effectiveness of steganography. It has brought up interesting techniques such as coverless steganography, and the ability to generate cover images. GAN-based architectures have been used in the following papers [14, 50, 61, 70, 93]. Refer to Figure 5 for a simple diagram of a GAN structure used for steganography.

ISGAN is a steganography technique that uses the GAN structure [93]. Similarly to the secure technique developed by Chen et al. [16], ISGAN is only able to send a grey-scale secret image. The encoder firsts converts the cover image into the YCrCb colour space where only the Y channel is used to hide the secret image since it holds luminance but no colour information. This colour space conversion does not affect back propagation. The encoder also uses an inception module, which helps to fuse feature maps with different receptive field sizes. To aid the speed of training, the technique also adds a residual module and batch normalisation. A CNN is used to decode with batch normalisation added after every convolutional layer excluding the final layer. The results of this model showed that it was able to achieve a high level of robustness, with low detectability when scrutinised using steganalysis tools.

In comparison to [71], ISGAN residuals were less obvious, showing that the extracted secret image of ISGAN is much closer to the original than [71]. ISGAN is also able to achieve higher levels of invisibility since ISGAN uses a grey scale image. Thus, there is a trade-off between the complexity of the image (i.e. RGB values) and the imperceptibility of embedded information.

Two separate designs are introduced by Li et al. in [50], which studies embedding data into texture images. The first model separates the texture image generation and secret image embedding processes. The texture image is generated using a deep convolutional generative neural network. The output of this is then used as the input for the concealing network for image hiding. The second model integrates the concealing network with the deep convolutional generative network. This second network should be able to generate the texture image while simultaneously embedding another image. The first model is easier to train and can be used in more diverse applications than the second model. Detection rates from steganalysis tools are almost 0 in both cases. These models provide high security, but with a few limitations. Firstly, the cover images generated are only textures and other subjects are not considered. Colour distortions also occur in the second model when the cover and secret images differ too much.

Coverless steganography is possible because of the features of GAN. The general encoding idea of the proposed coverless method [70] was that at first convolutional blocks were used to process the cover image to get a tensor, *a*. The secret message was then concatenated to *a* and processed through another convolutional block to get *b* which was the same size as *a*. The paper details two different models, one called basic model and the dense model. The basic model uses the aforementioned encoding scheme, whereas the dense model includes a *skip connection* to increase the embedding rate. The decoder for both models uses Reed Solomon algorithms on the tensor produced from the stego-image. The aim with this model was to improve the capacity and quality that other coverless steganography has not been able to achieve. Encoded image quality and payload capacity of the stego-images were improved when compared to [91]. The basic model introduced by Quin et al. [70] was able to perform significantly better in these aspects while the dense model was able only to match SteganoGAN [91]. The decoding network used in the coverless method [70] was also more accurate than the one proposed for SteganoGAN [91]. But there is a difference in the media encoded, where the coverless method [70] encodes a string of binary message while SteganoGAN [91] is able to encode an image.

Many steganography techniques have not been invertible and leave the cover image distorted after removing the secret piece of media. With the method proposed by Chang et al. [14], the model was able to achieve invertible stegnaography. The method used is based on the Regular-Singular (RS) method. RS realises lossless data embedding through invertible noise hiding. There are three discriminate blocks used in this technique – regular, singular and unusable – in order to get the RS map. The adversarial learning component serves to capture the regularity of natural images. The model uses conditional GAN to synthesise, where the generator uses a U-Net structure and a Markovian discriminator is used. The use of GAN in conjunction with the RS method greatly improved upon the results of previous RS based models.

Currently, the paper [20] uses a basic GAN structure and could be further improved or diversified with the adaption of other GAN structures. This could lead to further improvements in cover media recovery in terms of quality. Overall, the adversarial learning adopted in GAN-based models leads to greater robustness in steganography applications.

CycleGAN. CycleGAN is a variation of the GAN architecture for image to image translation. A diagram of a simple CycleGAN structure for data hiding is shown in Figure 6. The CycleGAN framerwork was adopted for steganography in S-CycleGAN [61]. Within the model, there were three discriminators, two of which used the same function as the original CycleGAN framework. The third was an increased steganalysis module used to distinguish the stego-image from the generated images. The training cycle consists of three stages. First is the translation of the image from the X-domain to the style of the Y-domain. Second is the use of a LSB matching algorithm to embed the secret message into the output of the first stage. The third stage is where the stego-image is reconstructed to the input image of the first generator to the second generator. The full objective function included adversarial loss for all discriminators, as well as cycle consistency loss for the generative models. The advantage of S-CycleGAN is its ability to produce high quality images which are also robust against steganalysis. Overall S-CycleGAN [61] showed results that were more resistant to detection, increasing the invisibility of embedded stego-images.

When S-CycleGAN [61] is compared against SGAN [80], the quality of the image is much higher, with 2.6x the Inception Score and 7x the Frechet Inception Distance of SGAN. In general, the results of S-CycleGAN [61] showed that it was much more robust than SGAN, and the combined use of the original CycleGAN framework and traditional steganography algorithm S-UNIWARD [32].

Table 2. Summary table of deep learning-based steganography methods.

Arch	Model	Embedding Network	Extractor Network	Remarks
CNN	[6]	CNN	CNN	Embeds coloured image in coloured cover image
	[84]	CNN	CNN	Concatenating channel for embedding
	[62]	R-CNN and CNN	N/A	Uses multiple techniques based on selected regions
	[76]	CNN	CNN	Embeds by scrambling and permutating pixels
	[15]	LSTM w/ CNN and HS	LSTM w/ convolutional layers and HS	Uses HS for reversibility
	[79]	CNN and adversarial emb	N/A	Trains stego-images with steganalysers
CNN w/ Adv. Training	[16]	CNN with ReLU	CNN with ReLU	Several models for different priorities
U-Net CNN	[20]	U-Net CNN w/ BN and ReLU	CNN w/ BN and ReLU	can withstand high and low frequencies
	[78]	CNN U-Net	CNN	Embeds hyperlinks in images
GAN	[88]	U-Net from Cycle-GAN	CNN	Improvement on [96] for diverse use
	[93]	CNN	CNN	Hides data in Y channel of cover image
	[50]	GAN and CNN	N/A	Generates a textured cover image
	[70]	CNN	CNN	Improvement on [91] in capacity and security
	[14]	U-Net CNN	CNN	Invertible
CycleGAN	[61]	CNN	CNN	Increased imperceptibility

3.4 Data Hiding Detection and Removal Mechanisms

As data hiding techniques improve, so do strategies for detecting and/or removing said data. Detection mechanisms are of particular important in steganography, as the field of steganalysis exists to detect and extract secret messages. Protecting against data detection in watermarking is less important than in steganography, as long as said data cannot be altered or removed from the cover media. This is because the primary purpose of watermarking is to identify the owners of the media – the hidden data is an ID, not a secret message that should be kept secure. Therefore, the focus of adversaries in watermarking is on the removal or degradation of the watermark without altering the original cover media. It is important to understand the goals and methods of adversaries in data hiding depending on the application, since these scenarios should be mitigated by the data hiding strategy.

3.4.1 Steganlysis. Steganlysis is the process of detecting hidden steganographic messages from an adversary's perspective. These analysis techniques are able to detect flaws in the embedding process and recognise if a piece of media has been encoded. There are two primary classes of steganalysis techniques: signature steganalysis and statistical steganalysis [43].

Signature steganalysis is comprised of two types called specific signature steganalysis and universal signature steganalysis. In specific signature steganalysis, the adversary is aware of the embedding method used, whereas universal signature steganalysis does not require this knowledge. Therefore, universal signature steganalysis can be used to detect several types of steganographic techniques [43, 66].

The development of steganalysis, along with new steganography techniques, is crucial since it shows how robust new embedding techniques are to ever-improving detection technologies.

3.4.2 Watermark Removal Strategies. Watermark removal techniques can be separated into three categories [69]. The first, blind watermark removal, is the technique that deep learning methods can defend against through varied attack simulation strategies. In blind removal techniques, the adversary has no knowledge of the watermarking process and attempts to degrade the watermark through attacks such as compression and geometric distortions. In key estimation attacks, the adversary has some knowledge of the watermarking scheme, and is then able to estimate the secret key used for embedding. Similarly, in tampering attacks the adversary has perfect knowledge of the watermarking scheme, resulting to a complete breakdown of the watermarking system where the secret key is explicitly obtained. In deep learning-based watermarking, the system is a black box even to its creators, hence the watermarking scheme cannot be uncovered by adversaries. Deep learning-based strategies only need to protect against the first kind of attacks; blind attacks. However, as watermark removal techniques improve through deep learning methods of their own, this is likely to change [23].

Table 3. Comparing deep learning-based watermarking models based on robustness, measured using BER. The 'Results Origin' column shows the paper where the BER result is taken from.

Model	Results Origin	Architecture	Cover Image Dimensions	Watermark Bits	BER (%)
HiDDeN [96]	[90]	GAN	128 x 128	30	20.12
IGA [90]	[90]	GAN	128 x 128	30	17.88
DA [60]	[90]	GAN	128 x 128	30	20.48
ReDMark [4]	[4]	GAN	128 x 128	30	18.18
Rotation [28]	[28]	GAN	64 x 64	8	8.0
DNN [95]	[95]	CNN	128 x 128	32	20.12
Double Detector-Discriminator [68]	[68]	GAN	256 x 256	32	5.53
Spatial-Spread [67]	[68]	GAN	256 x 256	32	8.38
Two-Stage [55]	[55]	GAN	128 x 128	30	8.3

Table 4. Comparing deep data hiding models based on encoded image quality, measured using PSNR. The 'Results Origin' column shows the paper where the PSNR result is taken from.

Model	Results Origin	Architecture	Cover Image Dimensions	Watermark Bits	PSNR (dB)
Two-Stage [55]	[55]	GAN	128 x 128	30	33.51
ABDH [87]	[87]	CycleGAN	512 x 512	256	31.79
ISGAN [93]	[87]	GAN	512 x 512	256	24.08
DA [60]	[60]	GAN	128 x 128	30	33.7
HiDDeN [96]	[60]	GAN	128 x 128	30	32.3
DNN [95]	[95]	CNN	128 x 128	32	39.93
WMNET [49]	[95]	CNN	128 x 128	32	38.01
ROMark [83]	[83]	GAN	128 x 128	30	27.80
IGA [90]	[90]	GAN	128 x 128	30	32.80
SteganoGAN [91]	[90]	GAN	128 x 128	30	30.10

3.5 Results

This section shows the results achieved by various deep data hiding models in terms of both robustness and imperceptibility. There is no standard dataset universally used for testing deep data hiding models, but COCO [53] is the most popular. Therefore, results achieved using this dataset were compared. Many papers record both Bit Error Rate (BER) and Peak Signal-to-Noise Ratio (PSNR), though some only record one. It should also be noted that tests were performed using different cover image and watermark dimensions, as noted in the tables, and certain papers simulated a different fixed pool of attacks during testing. Therefore, the results are not directly comparable, but merely give a rough indication of relative performance.

Table 3 compares deep data hiding models based on Bit Error Rate (BER), a measure of robustness. All models were tested using the COCO [53] dataset. The BER for all models is not directly comparable due to the different resolution of cover images and size of watermark payload. For example, a watermark embedded in a higher resolution image can retain integrity without sacrificing encoded image quality, therefore improving robustness. Conversely, a smaller watermark payload can be embedded with less of a degradation in cover image quality, also improving robustness.

Table 4 compares deep data hiding models based on Peak Signal-to-Noise Ratio (PSNR), a measure of encoded image quality. All models were tested using the COCO [53] dataset.

4 NOISE INJECTION TECHNIQUES

Aside from model architecture, deep learning-based data hiding techniques can also be classified based on noise injection methods. This refers to the attack simulation strategies employed during training, which corresponds to increased robustness in the final model. A number of techniques have been employed to increase robustness through attack simulation, the most common of which is using a pre-defined set of attacks simulated by a differentiable noise layer and subsequent geometric layers. Other approaches include using a trained CNN to generate novel noise patterns [16, 60], and adopting a two-stage training process to increase the complexity of simulated attacks [55]. The following section discusses different noise injection techniques, including their advantages and shortcomings.

4.1 Identity

Some earlier deep learning-based data hiding techniques [44] had no attack simulation and were simply a proof-of-concept for the technique. Subsequent models generally include a control training scenario where no noises are added (such as the 'Identity' layer in HiDDeN [96]. Unsurprisingly, techniques where no noise are added during training result in much lower rates of robustness. For example, for HiDDeN [96], the model trained exclusively with the Identity layer completely failed in successfully extracted watermarks when exposed to JPEG and Crop distortions (50% success rate), and performed significantly worse when exposed to all other attack types.

4.2 Pre-defined Attack Pool

The most common technique for noise injection is to use a fixed pool of pre-defined attack types. Then, during training, the model can be exposed to one, multiple, or the entire range of attack types. Each attack includes an intensity factor that can be adjusted in order to vary the strength of the attack during training. In an ideal scenario, the model would be just as effective with all attacks types when trained against the entire range as when trained with one specific attack. In most scenarios, the model uses a differentiable noise layer, meaning that all noise-based attacks must support back-propagation to cohesively work with the neural network [96]. Since non-differentiable noise attacks such as JPEG compression are common in real attack scenarios, they are incorporated into training using differentiable approximations [96]. The accuracy of these approximations are crucial to creating models that are robust against non-differentiable noise, therefore efforts have been made to revise and improve these approximations [67].

Further improvements to the pre-defined attack pool include a greater range of attacks. For example, a rotation layer was added to the framework by Hamamoto et al. [28], and subsampling and resizing attacks were incorporated by Plata et al. [67]. Various techniques have been introduced to improve the performance of models on a combined range of attacks, since early techniques using this method [96] showed improved performance when trained with specific attacks rather than a range. For instance, ROMarK [83] adopts a technique wherein distortions that generate the largest decoding error are generated and used to optimise the model's parameters during training.

4.3 Adversarial Training

Another technique for noise injection does not use a fixed set of pre-defined attacks, but rather uses a separately trained CNN to generate novel noise-based distortions [16, 60]. The Distortion Agnostic model [60] generates distortions based on adversarial examples, using a CNN to generate a diverse set of image distortions. The Distortion Agnostic model was tested against HiDDeN and exposed to a number of distortions not seen during training, such as saturation, hue, and resizing attacks, and found to be more effective. Although the Distortion Agnostic model could not surpass HiDDeN's effectiveness against a specific attack when trained only with that type, but this is not a practical training setup for a end use scenario. Using a trained CNN for attack generation can greatly improve robustness since distortions are generated in an adversarial training scenario. This means that distortions are generated explicitly to foil the decoder network throughout training.

4.4 Two-stage Training for Noise Injection

A novel training method developed by Liu et al. [55] improves noise injection techniques in two ways. Firstly, a training method dubbed Two-Stage Separable Deep Learning (TSDL) is used. This second stage of training is adopted for the decoder so that can work with any type of noise attacks, improving practicality. Because of this novel training scenario, this model can be trained using true non-differentiable distortion, rather than differentiable approximations. Aside from this distinction, a fixed pool of pre-defined attacks are still used during training. The use of true non-differentiable noise during training improved robustness against such attacks, including GIF and JPEG compression, but it is unclear from this work whether the increased overhead of the TDSL framework us a suitable trade-off for this improve performance, given the relative accuracy of differentiable approximations.

5 OBJECTIVE FUNCTIONS

Objective functions are used when training machine learning models to optimise their performance. The function measures the difference between the actual and predicted data points. By optimising the loss function, the model iteratively learns to reduce prediction errors. The loss functions used for the discussed data hiding models fall into the regression category, which deal with predicting values over a continuous range, rather than a set number of categories.

The aim of data hiding models is to optimise both robustness and imperceptibility, finding a balance between these two properties depending on the application needs. The most common architecture for deep data hiding models is the encoder-decoder architecture where the model is partitioned into two separate networks for encoding and decoding. A common method for evaluating loss is to have two separate equations, one for the encoder and one for the decoder, and compute total loss across the system as a weighted sum. Architectures that include an adversarial network also include adversarial loss in this sum. When optimising imperceptibility, the loss between the initial cover image and final encoded image must be minimised. When optimising robustness, the loss between the initial secret message or watermark information and the final extracted message must be minimised. This is the bases for the encoder-decoder loss method. This section will go through some common objective functions utilised in deep learning models for data hiding, as well as novel strategies that do not follow the encoder-decoder loss framework.

5.1 Mean Squared Error (MSE)

A function used to compute the difference between two sets of data. It squares the difference between data points in order to highlight errors further away from the regression line. In image-based data hiding, it can be used to compare the cover image to the encoded image by taking the square of the difference between all pixels in each image and dividing this by the number of pixels. The equation's sensitivity to outliers makes it useful when pinpointing obvious perceptual differences between two images. For example, a cluster of pixels that are obviously different from the original is perceptually obvious, but a change distributed over all image pixels is harder to spot. The equation is also used to compute the difference between the original and encoded watermark in some applications. For two images X and Y of dimensions W × H, the mean square error is given by the following equation:

$$MSE(X,Y) = \frac{1}{WH}\sum_{i=1}^{W}\sum_{j=1}^{H}(X_{i,j} - Y_{i,j})^2. \tag{1}$$

MSE is used in these works [28, 49, 50, 67, 68, 70, 91, 93] to calculate loss at the encoder, while many works [4, 16, 20, 55, 60, 76, 78, 81, 84, 96] use it to compute both encoder and decoder loss. MSE is used in conjunction with SSIM 17 to calculate perceptual difference at the encoder [50, 93]. The papers [78, 81] also use LPIPS perceptual loss [94] to evaluate loss at the encoder, which is discussed in Section 6.2. The Distortion Agnostic Model [60] incorporates additional adversarial loss into its Euclidean Distance functions. Loss at the encoder incorporates GAN loss with spectral normalisation to further control the quality of the encoded image. It is similar to the adversarial loss function used in [96] in Equation (6), but is incorporated into the encoder loss function rather than being weighted separately in the overall weighted sum.

5.2 Mean Absolute Error (MAE)

Similar to MSE, MAE is used to compute the difference between two sets of data, but instead takes the absolute value of the difference between data points. Therefore, it does not highlight errors further from the regression line or give any special significance to outliers. It is used to compute loss at the decoder in [49]. Since the secret message or watermark consists of binary values, it is more suitable to use MAE, which is suited to discrete values, to compute the difference to ensure balanced training of both the encoder and decoder networks. Additionally, it is used to calculate encoder and decoder loss in [14, 15, 84, 88]. For two binary watermarks M and M' of dimensions X x Y, the mean square error is given by the following equation:

$$MAE(M, M') = \frac{1}{XY} \sum_{i=1}^{X} \sum_{j=1}^{Y} |M(i,j) - M'(i,j)| \tag{2}$$

5.3 Cross Entropy Loss

This function is used to measure the difference between two probability distributions for a given variable, in this case the initial and extracted binary watermark message. The equation takes in $p(x)$, the true distribution, and $q(x)$, the estimated distribution. In digital watermarking this corresponds to the original watermark and the extracted watermark respectively, where $q(x)$ is the output of the decoder network representing the probability of watermark bits. Cross entropy loss takes the log of the predicted probability, therefore, the model will be 'punished' for a making a high probability prediction further from the true distribution. Cross entropy loss is given by the following equation:

$$H(p,q) = -\sum_{x} p(x) log(q(x)). \tag{3}$$

This technique is used for measuring loss at the decoder in [4, 28, 78, 81, 91], and used to compute adversarial loss in [14, 79]

5.4 Mean-Variance Loss

Variance loss is used to calculate how loss varies across an entire distribution. Using variance loss in conjunction with mean calculations leads to loss being distributed throughout an image rather than concentrated in certain areas. When calculating loss at the encoder, it is important that there are no concentrated areas of difference, since this will be easily perceptible. The paper [84] adopts variance loss and MAE loss for both encoder and decoder loss calculations.

A similar technique is applied by [67, 68], but for a different reason. Variance loss is used to calculate loss at the decoder rather than MSE since the redundant data used to create the watermark message means the original watermark and extracted watermark need not be identical. Rather than extract this redundant data, as was done in [49], the loss function was modified to a joint mean and variance function. For original message M and extracted message M', the mean and variance are given by:

$$L_m = (M, M') = \frac{b^2}{HW(n+k)} ||M - M'||_1, \tag{4}$$

$$L_v = (M, M') = \frac{b^2}{HW} \sum_{h=0}^{H_b} \sum_{w=0}^{W_b} Var(|M - M'|), \tag{5}$$

where b is the number of times every 'slice' of M is replicated across the image in the spatial spread embedding technique [67], and k relates to the number of tuples in the sequence created in this technique.

5.5 Adversarial Loss

For deep learning models that use a discriminator network for adversarial learning, an additional loss function is added to the overall optimisation that accounts for the discriminator's ability to detect an encoded image. Adversarial loss can be combined with loss at the encoder, since the encoder is the generative network, to improve the imperceptibility of the encoded watermark. The discriminator takes in an image X that is either encoded or unaltered, and predicts $A(I)\epsilon[0, 1]$, denoting the probability that X has been encoded with a watermark. The discriminator optimises the function for adversarial loss from its predictions given by the following equation:

$$l_a(X, Y) = log(1 - A(I_c) + log(A(Y)), \tag{6}$$

where X is the cover image, Y is the encoded image. This equation is combined with the loss function for encoder loss in [28, 83, 96].

5.6 KL Divergence

KL divergence is used to quantify the difference between probability distributions for a given random variable, for instance, the difference between an actual and observed probability distribution. It is often used in Generative Adversarial Networks to approximate the target probability distribution. This equation was used in [70] to calculate adversarial loss, where it is used in conjunction with JS divergence. This method for calculating adversarial loss can be represented in the following way:

$$KL(P||Q) = \sum_{c\epsilon C} P(c)log\frac{P(c)}{Q(c)}, \tag{7}$$

$$JS(P||Q) = \frac{1}{2}KL(P||\frac{P+Q}{2}) + \frac{1}{2}KL(Q||\frac{P+Q}{2}), \tag{8}$$

where c is the cover image, P is the probability distribution function for all cover images, and Q is the probability distribution function for the generated steganographic images.

5.7 Cycle Consistency Loss

This function is often used for CycleGAN implementations. The function performs unpaired image-to-image translation and is useful for problems where forward and backwards consistency between mapping functions is important. In both [61, 87], there are two generative and two discriminative models used, therefore the framework needs to learn the bijective mapping relationship between two image collections, where the first contains the original cover images and the second contains the secret images that serve as watermarks. The generative model needs to learn to transform from one source image domain to another, and this mapping relationship cannot ensure that the extracted watermark is the same as the original. Cycle consistency loss is detailed in [61] as an equation for the transformation of an X-domain style image to a Y-domain style image, where the generator models for both domains satisfy backward cycle consistency. Cycle adversarial loss is abstracted as the following:

$$D_M(M, M') = G_S(G_e(X, M, A)), \tag{9}$$

where D_S represents the cycle discriminative network judging the extracted watermarks, M is the original secret watermark, M' is the extracted watermark, G_S is the generative model that generates secret images to use as watermarks, X is the cover image, and A is the attention mask.

5.8 Wasserstein Loss

The following models [67, 68, 78, 81, 91] use a WGAN framework, therefore using a critic network rather than a discriminator to train the encoder. Rather than classifying an example as 'real' or 'fake', the critic outputs a number. The aim is to maximise this number for 'real' instances, which in this case are images that have been encoded with a message. Wasserstein loss is given by the following equation:

$$l_w = C(X) - C(E(X, M)), \tag{10}$$

where $C(X)$ is the critic output for the initial cover image, and $C(E(X, M))$ represents the critic output for the encoded image E consisting of the cover image and watermark message M.

5.9 Novel Approaches

Certain approaches to performance optimisation do not follow the above framework for optimising at the encoder and decoder, along with a third possible adversarial component.

Image Fusion. The technique developed in [95] differs from many other deep learning techniques for digital watermarking by approaching the process as an *image fusion* task, aiming to maximise correlation between the feature spaces of images.

The previous techniques focus on preserving certain parts of the cover image in the resulting encoded image. Different optimisations are applied to control embedding and extraction, and weights are used to control watermarking strength. Conversely, [95] takes 2 input spaces, C denoting the cover image, and W denoting the watermark. W is mapped onto one of its latent feature spaces W_f. Watermark embedding is performed by the function $\{W_f, C\} -> M$, where M denotes the fusion feature space of the watermark and cover image, creating an intermediate latent space. In this system, loss is computed using the *correlation* function given by:

$$\psi(m_i, w_f^i) = \frac{1}{2}(||g(f_1(w_f^i)), g(f_1(m_i))||_1 + ||g(f_2(w_f^i)), g(f_2(m_i))||_1), \tag{11}$$

where m_i and w_f^i are examples of the spaces M and W respectively. g denotes the Gram matrix of all possible inner products and f represents the convolutional blocks in the encoder network. A similar image fusion approach developed independently is utilised in [62]

Distortion Minimisation Framework. The steganography model is detailed in [79], steganography is formulated as an optimisation problem with a payload constraint:

$$min_S D(X, Y), \psi(Y) = k, \tag{12}$$

where X is the cover image, Y is the encoded stego-image, and $D(X, Y)$ is a function measuring the distortion caused by modifying X to create Y. $\psi(S)$ represents the data extracted from Y and k is the number of bits that make up the data. Different additive distortion functions may be used for D that measure the cost of increasing each pixel value in the cover image by 1. Large cost values are assigned to pixels more likely to cause perpetual differences in the final stego-image, and will therefore have a low probability of being modified during embedding.

Inconsistent Loss. The CycleGAN model is introduced in [87], there is an additional loss function called inconsistent loss that ensures that a secret watermark image can only be extracted from an encoded image. Since the second generative model G_S that extracts the watermarks also receives un-encoded cover images as input, any data it extracts should be completely different from the real secret watermark image. This is given by the following equation:

$$max_{G_S}(G_S, M') = max_{G_S}|G_S(X) - G_S(Y)|, \tag{13}$$

where M' is the extracted watermark image, X is the original cover image, and Y is the encoded 'target' image. [87] combines this with adversarial loss and cycle adversarial loss to evaluate the model's overall performance.

6 EVALUATION METRICS

Evaluation metrics for data hiding techniques can be divided into two primary categories; those evaluating robustness and those evaluating encoded image quality. Additionally, there are metrics for measuring information capacity.

6.1 Robustness

The follows metrics are used to evaluate the robustness of the embedded data. After attacks are applied to the encoded image and the data is extracted, these metrics are applied to compute the difference between the original message data and the data extracted from the image to judge how well that data survived distortions applied to it.

6.1.1 Bit Error Rate (BER):. is the number of bit errors divided by the total number of bits transferred over a certain time-frame. It therefore gives the percentage of erroneous bits during the transmission. It is used to compare the extracted message and the initial message, each converted into binary. The BER between the original message X and extracted message Y is given by the following equation:

$$BER(X, Y) = 100 \frac{\text{no. of bits in error}}{\text{total no. of bits transmitted}} \tag{14}$$

6.1.2 Normalised Correlation (NC):. is a measure of the similarity between two signals as a relative displacement function. Normalised correlation gives a values in the range [0,1] with a higher value denoting a higher similarity between images. The normalised correlation between the original message X and extracted message Y is given by the following equation:

$$NC(X, Y) = \frac{1}{WH} \sum_{i=0}^{W-1} \sum_{j=0}^{H-1} \delta(X_{i,j}, Y_{i,j}), \tag{15}$$

where

$$\delta(X, Y) = \begin{cases} 1, & \text{if } X = Y \\ 0, & \text{otherwise} \end{cases}$$

6.2 Quality

The follows metrics are used to evaluate the quality of the encoded image. It is linked to the property of imperceptibility since embedded data should not produce any detectable perturbations in the image.

6.2.1 Peak Signal to Noise Ratio (PSNR):. computes the difference between two images by taking the ratio between the maximum power of signal and the power of corrupting noise that affects the signal. In data hiding, it is used to evaluate the difference between the cover image and encoded image, showing the effectiveness of the embedding technique. PSNR is expressed in decibels (dB) on a logarithmic scale, with a value of above 30dB generally indicating that the image difference is not visible to the human eye. Therefore, PSNR can be used to quantify data imperceptibility. PSNR is defined by taking the Mean Square Error (MSE) of two images (cover image I_c and watermarked image I_w) using the following equation.

$$PSNR(I_c, I_w) = 10log_{10}\left(\frac{255^2}{MSE(I_c, I_w)}\right) \tag{16}$$

6.2.2 Structural Similarity Index (SSIM):. a perception-based model that measures image degradation as perceived changes in structural information. It considers variance and covariance, which measure the dynamic range of pixels in an image. Unlike MSE and PSNR, which measure absolute error, SSIM considers the dependencies between spatially close pixels, incorporating luminance and contrast masking. SSIM is given in the range [-1.0,1.0], with 1 indicating two identical images. The SSIM of two images is denoted as follows, where the mean of the two images is given by μ_x and μ_y, and their variances are given by σ_x^2 and σ_y^2.

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + k_1)(2\sigma_x\sigma_y + k_2)}{(\mu_x^2 + \mu_y^2 + k_1)(\sigma_x^2 + \sigma_y^2 + k_2)} \tag{17}$$

The constants k_1 and k_2 exist to stop a 0/0 calculation. In the default configuration, $k_1 = 0.01$ and $k_2 = 0.03$.

6.2.3 *Learned Perceptual Image Patch Similarity (LPIPS).*

: is another type of perceptual loss. The aim of LPIPS [94] is to measure human perceptual capabilities by going beyond the simpler SSIM 17 and PSNR 16 metrics, studying the deeper nuances of human perception. It is a model that evaluates the distance between image patches. The model uses a large set of distortions and real algorithm outputs. There is consideration of traditional distortions, noise patterns, filtering, spatial warping operations and CNN-based algorithm outputs. LPIPS is able to perform better due to a larger dataset used compared to other datasets similar to this kind, as well as being able to use the outputs of real algorithms. LPIPS has a scoring system where the lower the score, the more similar to the compared image while a higher score indicates a larger difference between the images.

6.2.4 *Frechet Inception Distance (FID):.*

is a model that measures the quality of a synthetic image. It was mainly developed to check the quality and the performance of images generated by GAN structures. FID uses the inception v3 model [77] where in the last layer of FID, it is used to get the computer-vision-specific features of an input image. They are calculated for the activations for all images, real and generated. These two distributions are then calculated using FID. The score compares the quality of syntetic images based on how well inception v3 classifies them as one of the 1,000 known objects. Essentially the generated images are not compared to real images, instead they are compared to other synthetic images that has already been compared to real images and scoring their similarity. A low FID score indicates a higher quality image while a high FID score indicates a lower quality image.

6.3 Capacity

Capacity is a measure of the amount of information that can be included in the data payload and subsequently embedded into the cover media.

6.3.1 *Bits Per Pixel (BPP).*

The number of bits is used to define the colour value of a pixel. A higher BPP value denotes a higher number of possible colour values for a pixel. Therefore, a higher BPP value for payload capacity shows that more identifying data can be embedded in the pixels of an image. BPP is used to calculate data payload capacity in [91].

6.3.2 *Reed Solomon Bits Per Pixel (RS-BPP).*

The paper [91] introduces a new metric RS-BPP based on Reed-Solomon codes. In deep data hiding models, measuring the number of bits that can be embedded per pixel depends on the model, the cover image, and the model itself, and is therefore non-trivial to measure just in BPP. Given an algorithm that returns an erroneous bit with probability p, the aim is to have a number of incorrect bits that is less than or equal to the number of bits that can be corrected. This can be represented by the equation:

$$pn \leq \frac{n-k}{2}, \tag{18}$$

where the $\frac{k}{n}$ ratio is the number of data bits that can be transmitted for each bit of the message. RS-BPP allows deep data hiding techniques such as [91] to be directly compared to traditional algorithms in terms of capacity.

7 DATASETS

This section shows a table showing the image databases used to train the discussed deep data hiding models.

Describable Textures Dataset (DTD): The DTD includes multiple kinds of labelled texture images, consisting of 5640 images in total. It was used to train [50].

Table 5. Summary table of datasets used to train deep learning-based data hiding models.

Name	Trained	Description
COCO [53]	[28, 50, 60, 62, 67, 68, 87, 90, 91, 96]	330 images of everyday scenes. Cluttered images useful for DH
DIV2K [3]	[70, 90, 91]	1000 low resolution images. Includes open scenery difficult for DH
CIFAR-10 [46]	[4, 95]	60k 32x32 images of singular objects
Pascal VOC [21]	[4, 93]	15k images of singular objects
BOSSBase [7]	[14, 15, 49, 64, 79]	9k 512x512 greyscale images in PGM format
ImageNet [75]	[6, 16, 20, 61, 78, 79, 84, 88, 93, 95]	14mil images organised based on WordNet hierarchy
MIRFLICKR [36]	[78, 81]	1mil Flickr images under Creative Commons license
Flickr30k [86]	[76]	31k images from Flickr under Creative Commons license
Labeled Faces in the Wild (LFW) [35]	[93]	13k images of faces from the web
Describable Textures Dataset (DTD) [19]	[50]	5k labelled texture images

COCO: COCO is a large-scale object detection, segmentation, and captioning dataset [53]. The dataset contains 330 thousand images of complex, everyday scenes including common objects. The goal of the dataset was to advance AI scene understanding by combining the more narrow goals of image classification, object localisation, and semantic segmentation. Since the dataset consists of relatively cluttered images picturing multiple objects, there are more surfaces and variation in textures in which to embed watermarking data, making it useful and popular for data hiding applications. COCO was used to train [28, 50, 60, 62, 67, 68, 87, 90, 91, 96] and to test [4, 55, 83, 95].

DIV2K: Div2K is a single-image super-resolution dataset of 1000 images of different scenes. The dataset consists of low resolution images with different types of degradations applied. Compared to COCO, it contains more images of open scenery, which can present a challenge for data embedding. Therefore, it is used to train [90, 91] in conjunction with [53] to ensure the model is trained on a variety of different types of images. On its own, it is used to train [70].

CIFAR-10: CIFAR-10 is a labelled subset of the 80-million Tiny Images Dataset [46], and contains 60 thousand small 32x32 images. They mostly picture single objects such as animals or vehicles, and are separated into 10 classes depending on the subject, though these are ignored for data hiding applications. CIFAR-10 was used to train [4, 95].

Pascal VOC: Similar to [46], the Pascal dataset [21] includes pictures of objects such as vehicles and animals rather than full scenes with multiple objects. This dataset has been widely used as a benchmark for object detection, semantic segmentation, and classification tasks. It was used to train [4, 93].

BOSSBase: This dataset contains 9074 512x512 greyscale images in PGM format [7]. It is widely used for training steganography algorithms. The dataset was used to train [14, 15, 49, 64, 79].

ImageNet: ImageNet contains 14 million images organized according to the WordNet hierarchy [75]. Images are classified into ‘synsets’ based on their subjects, all sorted and labelled accordingly. It was used to train [6, 16, 20, 61, 78, 79, 84, 88, 93, 95].

MIRFLICKR: The MIRFLICKR dataset [36] contains 1 million Flickr images under the Creative Commons license. It is used to train [78, 81].

Flickr30k: The Flickr30k dataset [86] contains 31,000 images collected from Flickr, together with 5 reference sentences provided by human annotators. It was used to train [76].

Labeled Faces in the Wild (LFW): The LFW dataset [35] public benchmark for face verification, containing more than 13,000 images of faces collected from the web. The dataset was used to train [93].

8 OPEN QUESTIONS AND FUTURE WORK

Deep learning for data hiding is a new and evolving research field, and there remain many different avenues to consider moving forward. This section will discuss some open questions that we believe warrant further research and consideration including expanding the applications of digital watermarking to other media domains, pursuing deep learning-based language watermarking, improving robustness against deep learning-based watermark removal attacks, watermarking machine learning models, combating the use of watermarking to launch backdoor attacks on machine learning models, and exploring the applications of watermarking for detecting and identifying synthetic media. Finally, future directions for steganography are discussed, including steganography used to spread malware.

8.1 Expanding Applications for Deep Learning Digital Watermarking Models

The deep watermarking models discussed in this survey were all focused on image watermarking, but there are many other important applications for watermarking other media. Although there are many traditional algorithms focused on watermarking video, audio, 3D models, and electronics, there is yet to be any deep learning models focusing on these areas. For example, are already numerous methods for watermarking 3D models [18], including new promising methods for watermarking vertices data [12]. There are also existing works concerning GANs generating 3D models [8, 98]. Furthermore, a recent paper from Google [85] details a promising deep learning technique of embedding watermark messages into simple 3D meshes which can then be decoded from 2D rendered images of the model from multiple angles and lighting configurations. It is noted in this work that robustness and capacity will need to be improved before practical application, particularly robustness to non-differentiable 3D attacks. More complex models and lighting arrangements could be explored with a better-quality renderer. A paper from Google research by Innfarn et al. [85] sets the precedent for watermarking 3D models using a deep learning, but more work is required before a generalised, practically-applicable framework for 3D watermarking can be developed.

Audio watermarking is in a similar situation. There already exist GAN-based frameworks for audio generation, and many traditional methods for audio watermarking [41, 54]. This suggests that current machine learning models may be capable of learning audio embedding techniques and applying these to audio databases. To our knowledge there are currently no works exploring deep learning frameworks for audio watermarking, although considering the current interest in this field, there are sure to be upcoming contributions.

Video watermarking is also a promising future direction. There are existing traditional algorithms for video watermarking [40], and deep learning techniques are also being considered, for example in [92]. This paper introduces RIVAGAN, a new architecture for robust video watermarking. The attention-based architecture is robust against common video processing attacks such as scaling, cropping, and compression. In the framework, a 32-bit watermark is embedded into a sequence of frames, and the watermark can be extracted from any individual or collection of frames. The framework uses an attention mechanism to identify robust areas for embedding, and produces watermarked footage that is nearly indistinguishable to human observers (approximately 52% detection accuracy). There is also DVMark [59] from Google Research, which employs a multiscale design where the watermark is distributed across multiple spatial-temporal scales. It was found to be more robust than traditional video watermarking algorithms (3D-DWT) and the deep learning-based image watermarking framework from HiDDeN [96] against video distortions, while retaining a high level of quality. A 3D-CNN that simulates video compression attacks was used during training to achieve high levels of robustness. The framework in DVMark [59] also includes a watermark detector that can analyse a long video and locate multiple short instances of copyrighted content. This could be useful for reliably identifying copyright infringement on online video platforms such as YouTube.

8.2 Text Watermarking for Combating Misinformation

Another promising application for deep watermarking is in watermarking text. As natural language generation technology improves, machine learning models are capable of generating highly fluent language that can fool human detectors [2]. There is growing concern that such models will be used to spread misinformation and 'fake news' online. The Adversarial Watermarking Transformer (AWT) developed by Sahar et al. [1] is the first end-to-end deep learning model for language watermarking. Language watermarking is inherently more complex than image, video, and audio watermarking because the language itself must be altered, which can case drastic syntactic and semantic changes. Previous techniques include synonym replacement and altering sentence structure, which rely on fixed rule-based techniques. The aim of such techniques is to achieve high effectiveness, secrecy, and robustness, while sustaining only subtle changes to the text, preserving correct structure and grammar, as well as language statistics. The deep learning model AWT [1] uses an attention mechanism and adversarial training to achieve robustness against attacks such as denoising, random changes, and re-watermarking. The model undergoes human evaluation and achieves better results than the state-of-the-art synonym substitution baseline. This technique only works effectively for long pieces of text such as news articles, whereas shorter pieces would require longer relative watermarks, thereby noticeably degrading the original text. For practical scenarios, it is suggested to combine this AWT technique with automated or human fact-checking to reduce the likelihood of false positives. Further research into deep learning-based models for language watermarking is highly important as text generating models continue to improve and become widely available to potentially malicious actors. This will help to identify misinformation and differentiate generated from genuine text.

8.3 Mitigating Watermark Removal Attacks

As technology for watermark embedding improves, so too does technology for attacking and removing those watermarks. Thus, the process can be regarded as an ever-evolving, adversarial game between content owners and attackers. Currently, deep learning-based techniques for watermark removal exist that are able to remove information robustly embedded using the top frequency domain algorithms [23]. This technique uses a simple CNN to perform denoising removal attacks, and is able to not only remove the watermark, but also recover the original cover images without significant quality degradation. It was tested on a dataset of watermarked images created using state-of-the-art DCT, DWT, and DFT traditional algorithms in a black-box setting. Although this technique is focused on traditional algorithms, as deep learning techniques for digital watermarking evolve, it is inevitable that adversarial techniques will continue to be developed that aim to remove these watermarks. Therefore, it is important to continue to improve the robustness of watermarking techniques so they can resist these emerging, deep learning-based removal methods.

Current deep watermarking strategies have been tested against a range of attacks including cropping, pixel dropout, compression, and blurring. However, in most models these attacks are encapsulated by a differetiable attack layer, as was implemented in [4, 83, 96], meaning that they must support back propagation, which is not representative of many real-world attack scenarios. However, it should be noted that these models can still simulate JPEG compression, which is non-differentiable, by using differentiable approximations. Promising techniques for improving the scope of attacks used during training include generating attacks using adversarial examples from a trained CNN [60], and the use of black-box noise [55], which are from algorithms encapsulated in the image processing software that are difficult to simulate. To achieve optimal robustness, it is important to train models on attacks generated from an adversarial network to improve results, rather than generating attacks from a fixed pool of differentiable attacks, as was done in earlier implementations.

8.4 Watermarking for Protecting Machine Learning Models

Another important application for digital watermarking is protecting machine learning models as intellectual property. Although the survey has discussed watermarking digital media such as images, audio, and video, machine learning models themselves require increasingly large amounts of computational resources and private training data to train and operate. Therefore, there is a growing need to protect machine learning models as intellectual property. Digital watermarking is just one of many tasks that was once done using traditional algorithms, but is now being offloaded to the effectiveness of machine learning strategies. As the happens, it is important to protect these models from theft and misuse, not only because of the resource expenditure by the owners in creating these models, but because their immense computational capabilities could be used for malicious activities.

There are many techniques currently being researched for watermarking machine learning models, most of which rely on embedded identifying information into training datasets [11]. However, as was learned through the concept of adversarial examples, even small perturbations in training instances can cause extreme degradation in the model's performance. Therefore, many watermarking strategies also sacrifice the model's classification accuracy.

A famous examples is DeepSigns [74], an end-to-end IP protection framework that inserts digital watermarks into deep learning models by embedding the watermark into the probability density function of the activation sets in different layers of the network. It is robust against a range of attacks, including model compression, fine-tuning, and watermark overwriting, which proved challenging for previous model watermarking techniques.

One recent and promising technique for watermarking training datasets is Entangled Watermarking Embeddings [42]. Instead of only learning features sampled from the task distribution, the defending model also samples data that encode watermarks when classifying data. Therefore, watermark data is entangled with legitimate training data, so an adversary attempting to remove the watermarks cannot do this without damaging the training data itself, and thereby sacrificing performance. The method uses Soft Nearest Neighbour Loss to increase entanglement, a new loss function. And the method is evaluated in the image and audio domains, showing that with this method an owner can claim with 95% confidence that model extraction has taken place to produce a similarly performing model by extracting prediction vectors. This technique notably shows robustness against adaptive adversaries, meaning the adversary has knowledge of the watermarking technique being used.

As machine learning models become more ubiquitous across a range of industries, it is important to implement digital watermarking strategies within the models themselves so that the owner's private information remains secure. However, just as was discussed in section 7.2, there will inevitably be technology developed to remove watermarks, even from machine learning models. For example, REFIT is a recent unified watermark removal framework based on fine-tuning. It does not require knowledge of the watermarks being removed, and is effective against a wide range of current watermarking techniques [17].

8.5 Watermarking for Launching Backdoor Attacks

Watermarks can be embedded into the training examples of machine learning models in order to cause inaccurate classifications when the model is deployed. Many third party cloud computing providers such as Google, Microsoft, Azure, and Amazon provide Machine Learning as a Service (MLaaS) to train machine learning models. A malicious party can embed watermarks into training data images and train the model to misclassify such examples, either randomly (random target attack), or mislabel as a different example (single target attack). As more third-party providers offer MLaaS, and machine learning models become more ubiquitous, backdoor attacks on neural networks are certain to become more common. Therefore, it is important to be able to detect watermarks embedded in training examples. As digital watermarking technologies become more advanced through deep learning as discussed in this survey, this will become more difficult. In this case, the improving watermark removal techniques discussed in Section 8.2 could be pursued for a benign rather than malicious purpose. Although existing works discuss visible watermarks embedded in training images [26, 56], there are growing efforts to construct invisible backdoor attacks that cannot be detected by a human moderator, or by automated backdoor detection techniques [51, 52]. Although the techniques presented in this survey are promising for digital IP protection purposes, they will inevitably be utilised for malicious purposes such as embedding undetectable, invisible backdoors in machine learning models.

8.6 Deepfake Detection and Identification

Another promising application for digital watermarking is in the identification and detection of synthetic media. As GAN technology for producing synthetic media such as images, audio, and video improve, deepfakes will become both more accurate and easier to produce. When deepfakes are produced, it is important to be able to identify the original source of the media that has been manipulated. Similarly, it is important to be able to identify a piece of media as synthetic in the first place, without malicious parties removing this identifying tag and presenting the synthetic media as genuine. A recent paper presents DeepTag, an end-to-end deep watermarking framework that includes a GAN simulator that applies common distortions to facial images [82]. The watermark can be recovered to identify the original unaltered facial image. In future, as regulations surrounding deepfakes arise, watermarking techniques that are robust to GAN-based distortions will become increasingly important. A connected application is embedding watermarks into synthetic media so that it can be easily identified as such.

8.7 Malware using Steganography

Digital steganography can be used maliciously to spread malware to victim technology. The ability to hide an executable file within an image or audio file gives attackers an easy attack vector to target unaware users. There have also been known cases of attackers using steganography to pass data through unsuspecting platforms. They can easily set a time for the receiver and either upload or update an image temporarily. At this set time the receiver can download and save the photo, decode the message, and then the attacker can restore the image to the original or delete the image. This could be almost impossible to detect when third parties are unaware of where the attack will take place, and would be even more unlikely to catch the act if the stego-image used was highly imperceptible. These high levels of imperceptibility can now be achieved easily though modern deep learning approaches.

Steganalysis could implemented into antivirus software that not only scans images but scans sites before entering. Though this would be a difficult task due to the large amount of power it would require, constantly scanning almost every website or item on the web page just *in case* there is malware present. There could be a filter that decides when steganalysis could be used such as situations when users decide to continue onto an already scanned suspicious website. Currently, the possibilities are limited, but the implementation of steganalysis in antivirus software may be essential in the future as steganography techniques both improve in performance and become more widely accessible to the public.

9 CONCLUSION

This survey presented an overview of deep learning techniques for data hiding, including both watermarking and steganography, comparing current state-of-the-art methods classified in terms of network architecture and model performance. The survey also compared the objective functions and evaluation metrics for model performance measurement, and finally looked at possible future directions for research in this field. This is a promising new field of research with the potential to revolutionise both the protection of digital IP and the security of communication across a range of industries. As techniques improve, and applications spread to other types of media, deep learning-based methods for data hiding will likely surpass the capabilities of any traditional algorithms in all media and greatly enhance digital information security.

REFERENCES

[1] Sahar Abdelnabi and Mario Fritz. 2021. Adversarial Watermarking Transformer: Towards Tracing Text Provenance with Data Hiding. (2021). arXiv:2009.03015 [cs.CR]

[2] David Ifeoluwa Adelani, Haotian Mai, Fuming Fang, Huy H. Nguyen, Junichi Yamagishi, and Isao Echizen. 2019. Generating Sentiment-Preserving Fake Online Reviews Using Neural Language Models and Their Human- and Machine-based Detection. (2019). arXiv:1907.09177 [cs.CL]

[3] E. Agustsson and R. Timofte. 2017. NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study. In 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). 1122–1131. https://doi.org/10.1109/CVPRW.2017.150

[4] Mahdi Ahmadi, Alireza Norouzi, S. M. Reza Soroushmehr, Nader Karimi, Kayvan Najarian, Shadrokh Samavi, and Ali Emami. 2020. ReDMark: Framework for Residual Diffusion Watermarking on Deep Networks. Expert Systems with Applications 146 (2020), 113157.

[5] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein GAN. PMLR (2017). arXiv:1701.07875 [stat.ML]

[6] Shumeet Baluja. 2017. Hiding images in plain sight: Deep steganography. In Proceedings of the 31st International Conference on Neural Information Processing Systems. 2066–2076.

[7] Patrick Bas, Tomáš Filler, and Tomáš Pevný. 2011. "Break Our Steganographic System": The Ins and Outs of Organizing BOSS. In Information Hiding, Tomáš Filler, Tomáš Pevný, Scott Craver, and Andrew Ker (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 59–70.

[8] Shailesh Bendale. 2020. A Study of Generative Adversarial Networks in 3D Modelling. International Research Journal of Engineering and Technology 6 (05 2020), 826–830. Issue 12.

[9] Sharbani Bhattacharya. 2014. Survey on Digital Watermarking- A Digital Forensics & Security Application. International Journal of Advanced Research in Computer Science and Software Engineering ISSN number:2277-128X 4 (11 2014), 11.

[10] Benedikt Boehm. 2014. Stegexpose-A tool for detecting LSB steganography. arXiv preprint arXiv:1410.6656 (2014).

[11] Franziska Boenisch. 2020. A Survey on Model Watermarking Neural Networks. arXiv:2009.12153 [cs.CR]

[12] Marco Botta, Davide Cavagnino, Marco Gribaudo, and Pietro Piazzolla. 2020. Fragile Watermarking of 3D Models in a Transformed Domain. Applied Sciences 10, 9 (2020). https://doi.org/10.3390/app10093244

[13] Rajarathnam Chandramouli, Grace Li, and Nasir Memon. 2002. Adaptive steganography. Proceedings of SPIE - The International Society for Optical Engineering (01 2002).

[14] Ching-Chun Chang. 2020. Adversarial Learning for Invertible Steganography. IEEE Access 8 (2020), 198425–198435.

[15] Ching-Chun Chang. 2021. Neural Reversible Steganography with Long Short-Term Memory. Security and Communication Networks 2021 (2021).

[16] Beijing Chen, Jiaxin Wang, Yingyue Chen, Zilong Jin, Hiuk Jae Shim, and Yun-Qing Shi. 2020. High-Capacity Robust Image Steganography via Adversarial Network. KSII Transactions on Internet & Information Systems 14, 1 (2020).

[17] Xinyun Chen, Wenxiao Wang, Chris Bender, Yiming Ding, Ruoxi Jia, Bo Li, and Dawn Song. 2019. REFIT: a Unified Watermark Removal Framework for Deep Learning Systems with Limited Data. CoRR abs/1911.07205 (2019). arXiv:1911.07205 http://arxiv.org/abs/1911.07205

[18] Chang-Min Chou and D.C Tseng. 2008. Technologies for 3D Model Watermarking: A Survey. International Journal of Engineering and Applied Sciences 2 (07 2008), 126–136.

[19] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. 2014. Describing Textures in the Wild. In Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR).

[20] Xintao Duan, Kai Jia, Baoxia Li, Daidou Guo, En Zhang, and Chuan Qin. 2019. Reversible image steganography scheme based on a U-Net structure. IEEE Access 7 (2019), 9314–9323.

[21] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. 2010. The Pascal Visual Object Classes (VOC) Challenge. International Journal of Computer Vision 88, 2 (June 2010), 303–338.

[22] Aidin Ferdowsi and Walid Saad. 2017. Deep Learning-Based Dynamic Watermarking for Secure Signal Authentication in the Internet of Things. CoRR abs/1711.01306 (2017). arXiv:1711.01306 http://arxiv.org/abs/1711.01306

[23] Linfeng Geng, Weiming Zhang, Haozhe Chen, Han Fang, and Nenghai Yu. 2020. Real-time attacks on robust watermarking tools in the wild by CNN. Journal of Real-Time Image Processing 17 (6 2020). https://doi.org/10.1007/s11554-020-00941-8

[24] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Networks. Advances in Neural Information Processing Systems 3, 11 (2014). arXiv:1406.2661 [stat.ML]

[25] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and Harnessing Adversarial Examples. arXiv:1412.6572 [stat.ML]

[26] Tianyu Gu, Kang Liu, Brendan Dolan-Gavitt, and Siddharth Garg. 2019. BadNets: Evaluating Backdooring Attacks on Deep Neural Networks. IEEE Access 7 (01 2019), 47230–47244. https://doi.org/10.1109/ACCESS.2019.2909068

[27] Gokhan Gul and Fatih Kurugollu. 2011. A New Methodology in Steganalysis: Breaking Highly Undetectable Steganograpy (HUGO). International Workshop on Information Hiding 6958 (2011), 71–84.

[28] Ippei HAMAMOTO and Masaki Kawamura. 2020. IEICE Transactions on Information and Systems E103.D (01 2020), 33–41. https://doi.org/10.1587/transinf.2019MUP0007

[29] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, Las Vegas, NV, USA, 770–778. https://doi.org/10.1109/CVPR.2016.90

[30] Alexander Herrigel, S. Voloshynovskiy, and Zenon Hrytskiv. 2000. An Optical/Digital Identification/Verification System based on Digital Watermarking Technology. Proceedings of SPIE - The International Society for Optical Engineering (03 2000). https://doi.org/10.1117/12.388440

[31] V Holub and J Fridrich. 2012. Designing steganographic distortion using directional filters. IEEE International Workshop on Information Forensics and Security (Dec. 2012), 234–239.

[32] V Holub, J Fridrich, and T Denemark. 2014. Universal distortion function for steganography in an arbitrary domain. EURASIP Journal on Information Security (2014).

[33] Guang Hua, Jiwu Huang, Yun Q. Shi, Jonathan Goh, and Vrizlynn L.L. Thing. 2016. Twenty years of digital audio watermarking—a comprehensive review. Signal Processing 128 (2016), 222–242. https://doi.org/10.1016/j.sigpro.2016.04.005

[34] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. 2016. Densely Connected Convolutional Networks. CoRR abs/1608.06993 (2016). arXiv:1608.06993 http://arxiv.org/abs/1608.06993

[35] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. 2007. Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. Technical Report 07-49. University of Massachusetts, Amherst.

[36] Mark J. Huiskes and Michael S. Lew. 2008. The MIR Flickr Retrieval Evaluation (MIR '08). Association for Computing Machinery, New York, NY, USA, 39–43. https://doi.org/10.1145/1460096.1460104

[37] Mehdi Hussain, Ainuddin Wahid Abdul Wahab, Yamani Idna Bin Idris, Anthony T.S. Ho, and Ki-Hyun Jung. 2018. Image steganography in spatial domain: A survey. Signal Processing: Image Communication 65 (2018), 46–66. https://doi.org/10.1016/j.image.2018.03.012

[38] Shunquan T Hussain I, Zeng J. 2020. Survey on Deep Convolutional Neural Networks for Image Steganography and Steganalysis. KSII Transactions on Internet and Information Systems 14 (2020), 1228–1248. Issue 3. https://doi.org/10.3837/tiis.2020.03.017

[39] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. 2018. Image-to-Image Translation with Conditional Adversarial Networks. arXiv:1611.07004 [cs.CV]

[40] Anita Jadhav and Megha Kolhekar. 2014. Digital Watermarking in Video for Copyright Protection. In Proceedings - International Conference on Electronic Systems, Signal Processing, and Computing Technologies, ICESC 2014. IEEE, Nagpur, India, 140–144. https://doi.org/10.1109/ICESC.2014.29

[41] Shulei Ji, Jing Luo, and Xinyu Yang. 2020. A Comprehensive Survey on Deep Music Generation: Multi-level Representations, Algorithms, Evaluations, and Future Directions. arXiv:2011.06801 [cs.SD]

[42] Hengrui Jia, Christopher A. Choquette-Choo, Varun Chandrasekaran, and Nicolas Papernot. 2021. Entangled Watermarks as a Defense against Model Extraction. arXiv:2002.12200 [cs.CR]

[43] 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 335 (2019), 299–326.

[44] Haribabu Kandi, Deepak Mishra, and Subrahmanyam R.K. Sai Gorthi. 2017. Exploring the learning capabilities of convolutional neural networks for robust image watermarking. Computers & Security 65 (2017), 247–268. https://doi.org/10.1016/j.cose.2016.11.016

[45] S Kartik, A Aggarwal, T Singhania, D Gupta, and A Khanna. 2019. Hiding Data in Images Using Cryptography and Deep Neural Network. Journal of Artificial Intelligence and Systems (2019), 143–162. https://doi.org/10.33969/AIS.2019.11009

[46] Alex Krizhevsky. 2012. Learning Multiple Layers of Features from Tiny Images. University of Toronto (05 2012).

[47] Chandan Kumar, Amit Kumar Singh, and Pardeep Kumar. 2018. A recent survey on image watermarking techniques and its application in e-governance. Multimedia Tools and Applications (Feb. 2018). https://doi.org/10.1007/s11042-017-5222-8

[48] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. arXiv:1607.02533 [cs.CV]

[49] Jae-Eun Lee, Young-Ho Seo, and Dong-Wook Kim. 2020. Convolutional Neural Network-Based Digital Image Watermarking Adaptive to the Resolution of Image and Watermark. Applied Sciences 10, 19 (2020). https://doi.org/10.3390/app10196854

[50] Chuanlong Li, Yumeng Jiang, and Marta Cheslyar. 2018. Embedding image through generated intermediate medium using deep convolutional generative adversarial network. Computers, Materials & Continua 56, 2 (2018), 313–324.

[51] Shaofeng Li, Hui Liu, Tian Dong, Benjamin Zi Hao Zhao, Minhui Xue, Haojin Zhu, and Jialiang Lu. 2021. Hidden backdoors in human-centric language models. ACM Conference on Computer and Communications Security (CCS) (2021).

[52] Shaofeng Li, Minhui Xue, Benjamin Zhao, Haojin Zhu, and Xinpeng Zhang. 2020. Invisible Backdoor Attacks on Deep Neural Networks via Steganography and Regularization. IEEE Transactions on Dependable and Secure Computing PP (09 2020), 1–1. https://doi.org/10.1109/TDSC.2020.3021407

[53] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. 2015. Microsoft COCO: Common Objects in Context. arXiv:1405.0312 [cs.CV]

[54] Jen-Yu Liu, Yu-Hua Chen, Yin-Cheng Yeh, and Yi-Hsuan Yang. 2020. Unconditional Audio Generation with Generative Adversarial Networks and Cycle Regularization. arXiv:2005.08526 [eess.AS]

[55] Yang Liu, Mengxi Guo, Jian Zhang, Yuesheng Zhu, and Xiaodong Xie. 2019. A Novel Two-Stage Separable Deep Learning Framework for Practical Blind Watermarking. In MM '19: Proceedings of the 27th ACM International Conference on Multimedia (Nice, France) (MM '19). Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3343031.3351025

[56] Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. 2017. Trojaning Attack on Neural Networks. The Network and Distributed System Security Symposium (NDSS) (2017).

[57] David Lowe. 2004. Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60 (11 2004), 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94

[58] Rongxing Lu, Guoyin Zhang, Liang Kou, Liguo Zhang, Chao Liu, Qingan Da, and Jianguo Sun. 2017. A New Digital Watermarking Method for Data Integrity Protection in the Perception Layer of IoT. Security and Communication Networks 2017 (10 2017), 12. https://doi.org/10.1155/2017/3126010

[59] Xiyang Luo, Yinxiao Li, Huiwen Chang, Ce Liu, Peyman Milanfar, and Feng Yang. 2021. DVMark: A Deep Multiscale Framework for Video Watermarking. (04 2021).

[60] Xiyang Luo, Ruohan Zhan, Huiwen Chang, Feng Yang, and Peyman Milanfar. 2020. Distortion Agnostic Deep Watermarking. Computer Vision Foundation (2020). arXiv:2001.04580 [cs.MM]

[61] Ruohan Meng, Qi Cui, Zhili Zhou, Zhangjie Fu, and Xingming Sun. 2019. A steganography algorithm based on CycleGAN for covert communication in the Internet of things. IEEE Access 7 (2019), 90574–90584.

[62] Ruohan Meng, Steven G Rice, Jin Wang, and Xingming Sun. 2018. A fusion steganographic algorithm based on faster R-CNN. Computers, Materials & Continua 55, 1 (2018), 1–16.

[63] Saraju P. Mohanty, A. Sengupta, P. Guturu, and E. Kougianos. 2017. Everything You Want to Know About Watermarking: From Paper Marks to Hardware Protection: From paper marks to hardware protection. IEEE Consumer Electronics Magazine 6 (2017), 83–91.

[64] Seung-Min Mun, Seung-Hun Nam, Haneol Jang, Dongkyu Kim, and Heung-Kyu Lee. 2019. Finding robust domain from attacks: A learning framework for blind watermarking. Neurocomputing 337 (2019), 191–202. https://doi.org/10.1016/j.neucom.2019.01.067

[65] Seung-Min Mun, Seung-Hun Nam, Haneol Jang, Dongkyu Kim, and Heung-Kyu Lee. 2019. Finding robust domain from attacks: A learning framework for blind watermarking. Neurocomputing 337 (2019), 191–202. https://doi.org/10.1016/j.neucom.2019.01.067

[66] Arooj Nissar and A.H. Mir. 2010. Classification of steganalysis techniques: A study. Digital Signal Processing 20, 6 (2010), 1758–1770. https://doi.org/10.1016/j.dsp.2010.02.003

[67] Marcin Plata and Piotr Syga. 2020. Robust Spatial-Spread Deep Neural Image Watermarking. In 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). 62–70. https://doi.org/10.1109/TrustCom50675.2020.00022

[68] Marcin Plata and Piotr Syga. 2020. Robust watermarking with double detector-discriminator approach. (06 2020).

[69] Luis Pérez-Freire, Pedro Comesaña, Juan Ramón Troncoso-Pastoriza, and Fernando Pérez-González. 2006. Watermarking Security: A Survey. Lecture Notes in Computer Science 4300 (01 2006), 41–72.

[70] Jiaohua Qin, Jing Wang, Yun Tan, Huajun Huang, Xuyu Xiang, and Zhibin He. 2020. Coverless image steganography based on generative adversarial network. Mathematics 8, 9 (2020), 1394.

[71] Rafia Rahim, Shahroz Nadeem, et al. 2018. End-to-end trained cnn encoder-decoder networks for image steganography. In Proceedings of the European Conference on Computer Vision (ECCV) Workshops. 0–0.

[72] Anees V Rasmi A, Arunkumar B. 2019. A Comprehensive Review of Digital Data Hiding Techniques. Pattern Recognition and Image Analysis 29 (2019). Issue 4. https://doi.org/10.1134/S105466181904014X

[73] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. CoRR abs/1505.04597 (2015). arXiv:1505.04597 http://arxiv.org/abs/1505.04597

[74] Bita Darvish Rouhani, Huili Chen, and Farinaz Koushanfar. 2018. DeepSigns: A Generic Watermarking Framework for IP Protection of Deep Learning Models. (2018). arXiv:1804.00750 [cs.CR]

[75] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. arXiv:1409.0575 [cs.CV]

[76] Kartik Sharma, Ashutosh Aggarwal, Tanay Singhania, Deepak Gupta, and Ashish Khanna. 2019. Hiding Data in Images Using Cryptography and Deep Neural Network. arXiv preprint arXiv:1912.10413 (2019).

[77] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going Deeper With Convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[78] Matthew Tancik, Ben Mildenhall, and Ren Ng. 2020. Stegastamp: Invisible hyperlinks in physical photographs. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2117–2126.

[79] Weixuan Tang, Bin Li, Shunquan Tan, Mauro Barni, and Jiwu Huang. 2019. CNN-based adversarial embedding for image steganography. IEEE Transactions on Information Forensics and Security 14, 8 (2019), 2074–2087.

[80] Denis Volkhonskiy, Ivan Nazarov, and Evgeny Burnaev. 2020. Steganographic generative adversarial networks. In Twelfth International Conference on Machine Vision (ICMV 2019), Vol. 11433. International Society for Optics and Photonics, 114333M.

[81] Kuangshi Wang, Li Li, Ting Luo, and Chin-Chen Chang. 2020. Deep Neural Network Watermarking Based on Texture Analysis. In Artificial Intelligence and Security, Xingming Sun, Jinwei Wang, and Elisa Bertino (Eds.). Springer Singapore, Singapore, 558–569.

[82] Run Wang, Felix Juefei-Xu, Qing Guo, Yihao Huang, Lei Ma, Yang Liu, and Lina Wang. 2020. DeepTag: Robust Image Tagging for DeepFake Provenance. (2020). arXiv:2009.09869 [cs.CR]

[83] Bingyang Wen and Sergul Aydore. 2019. ROMark: A Robust Watermarking System Using Adversarial Training. arXiv:1910.01221 [cs.CV]

[84] Pin Wu, Yang Yang, and Xiaoqiang Li. 2018. Stegnet: Mega image steganography capacity with deep convolutional network. Future Internet 10, 6 (2018), 54.

[85] Innfarn Yoo, Huiwen Chang, Xiyang Luo, O. Stava, Ce Liu, P. Milanfar, and Feng Yang. 2021. Deep 3D-to-2D Watermarking: Embedding Messages in 3D Meshes and Extracting Them from 2D Renderings. ArXiv abs/2104.13450 (2021).

[86] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. 2014. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. Transactions of the Association for Computational Linguistics 2 (02 2014), 67–78. https://doi.org/10.1162/tacl_a_00166 arXiv:https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00166/1566848/tacl_a_00166.pdf

[87] Chong Yu. 2020. Attention Based Data Hiding with Generative Adversarial Networks. Proceedings of the AAAI Conference on Artificial Intelligence 34, 01 (Apr. 2020), 1120–1128. https://doi.org/10.1609/aaai.v34i01.5463

[88] Chaoning Zhang, Philipp Benz, Adil Karjauv, Geng Sun, and In So Kweon. 2020. Udh: Universal deep hiding for steganography, watermarking, and light field messaging. Advances in Neural Information Processing Systems 33 (2020), 10223–10234.

[89] Chaoning Zhang, Chenguo Lin, Philipp Benz, Kejiang Chen, Weiming Zhang, and In So Kweon. 2021. A Brief Survey on Deep Learning Based Data Hiding, Steganography and Watermarking. arXiv:2103.01607 [cs.CR]

[90] Honglei Zhang, Hu Wang, Yuanzhouhan Cao, Chunhua Shen, and Yidong Li. 2021. Robust Watermarking Using Inverse Gradient Attention. (2021). arXiv:2011.10850 [cs.CV]

[91] Kevin Alex Zhang, Alfredo Cuesta-Infante, Lei Xu, and Kalyan Veeramachaneni. 2019. SteganoGAN: High Capacity Image Steganography with GANs. (2019). arXiv:1901.03892 [cs.CV]

[92] Kevin Alex Zhang, Lei Xu, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. 2019. Robust Invisible Video Watermarking with Attention. arXiv:1909.01285 [cs.MM]

[93] Ru Zhang, Shiqi Dong, and Jianyi Liu. 2019. Invisible steganography via generative adversarial networks. Multimedia tools and applications 78, 7 (2019), 8559–8575.

[94] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. arXiv:1801.03924 [cs.CV]

[95] Xin Zhong, Pei-Chi Huang, Spyridon Mastorakis, and Frank Y. Shih. 2020. An Automated and Robust Image Watermarking Scheme Based on Deep Neural Networks. IEEE Transactions on Multimedia (2020). https://doi.org/10.1109/tmm.2020.3006415

[96] Jiren Zhu, Russell Kaplan, Justin Johnson, and Li Fei-Fei. 2018. HiDDen: Hiding Data with Deep Networks. In Proceedings of the European Conference on Computer vision (ECCV 2018). 657–672.

[97] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. 2020. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. arXiv:1703.10593 [cs.CV]

[98] Cihan Öngün and Alptekin Temizel. 2019. Paired 3D Model Generation with Conditional Generative Adversarial Networks. arXiv:1808.03082 [cs.CV]