

VisCode: Embedding Information in Visualization Images using Encoder-Decoder Network

Peiying Zhang, Chenhui Li, and Changbo Wang

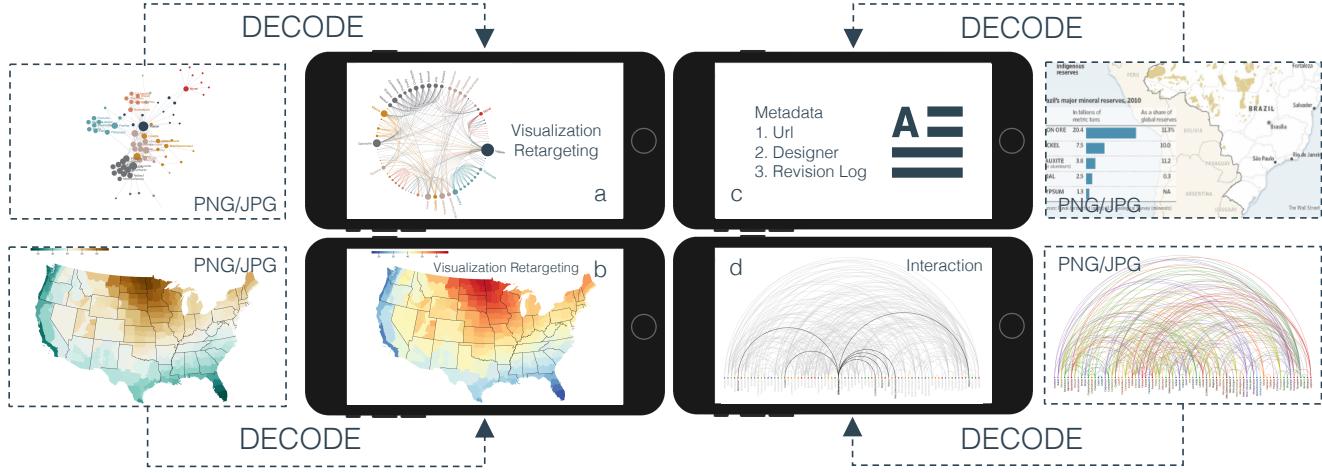


Fig. 1. VisCode can be used in a variety of practical applications. The input is a static visualization image. According to the encoding rules, we can retarget the original visual style after performing the decoding operation. Visualization retargeting includes representation (a) and theme color (b). VisCode can be also applied to metadata embedding as shown in (c) and large-scale information embedding. After decoding the source code into a static image, we can quickly build a visualization application and apply a common interaction such as selection as shown in (d).

Abstract—We present an approach called VisCode for embedding information into visualization images. This technology can implicitly embed data information specified by the user into a visualization while ensuring that the encoded visualization image is not distorted. The VisCode framework is based on a deep neural network. We propose to use visualization images and QR codes data as training data and design a robust deep encoder-decoder network. The designed model considers the salient features of visualization images to reduce the explicit visual loss caused by encoding. To further support large-scale encoding and decoding, we consider the characteristics of information visualization and propose a saliency-based QR code layout algorithm. We present a variety of practical applications of VisCode in the context of information visualization and conduct a comprehensive evaluation of the perceptual quality of encoding, decoding success rate, anti-attack capability, time performance, etc. The evaluation results demonstrate the effectiveness of VisCode.

Index Terms—Information visualization, information steganography, autocoding, saliency detection, visualization retargeting.

1 INTRODUCTION

For a long time, information visualizations were created by artists, designers or programmers, and most of them were disseminated in the form of bitmap images. Many researchers have performed many studies on specific aspects of the design of information representations, such as the visualization layout, the color theme, and more specific explorations. However, with the development of fast network technology, the spread of visualization images has encountered two problems. One is that the copyright problem for visualization has not been effectively solved. Most designers use explicit icons to present copyright information, but this method affects the aesthetics of a chart and can obstruct important information. Moreover, when a visualization is converted into a bitmap image, the data in the vi-

sualization are converted into pixels, and the original data are completely lost. Although several researchers, such as Poco et al. [40], have applied a pattern recognition approach to visualization images to retarget the color theme, the effectiveness of data recognition is still related to the form of visualization. As an alternative approach, information steganography [13] (a means of implicit information embedding) for visualization images has potential for copyright protection and source data preservation.

Information steganography has a very meaningful practical application in the field of visualization design. In many forms of information propagation, such as social networks, blogs, and magazines, bitmap images are used most frequently and do not require the support of an additional web server. After conversion into an image, the original code used to generate the image is lost, thus increasing the difficulty of modification and visualization redesign. Similar difficulties also arise when designers and programmers collaborate. However, if a designer could implicitly embed the key visualization code into an image during the visualization design period, then the dissemination and revision of the visualized information would not require two different types of media (image and text), which would help programmers integrate such visualizations into big data analysis systems. Another potential application is the visualization retargeting. Visualization retargeting is very useful for the creation of visualizations. On the one hand, it reduces the

• Peiying Zhang, Chenhui Li, and Changbo Wang are with School of Computer Science and Technology, East China Normal University. Chenhui Li is the corresponding author. E-mail: chli@cs.ecnu.edu.cn.

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

designer's workload. On the other hand, it broadens the artist's creative space. Fig. 1 shows four practical applications of our work.

Information steganography is different from digital watermarking, which is a technique for hiding information in a piece of media (usually an image). As described in the book of Cox et al. [13], steganography has a long history. Researchers have proposed a variety of methods of embedding information in images for diverse steganographic applications. As surveyed by Ghadirli et al. [17], the techniques used in image steganography include chaos-based methods, permutation-based methods, frequency-domain methods, hash-based methods, and evolutionary methods, among others. The continuous development of steganographic technology and the recent emergence of deep learning technology have enabled the extension of image steganography into broader areas of application. To our knowledge, however, few researchers have studied how to embed information in visualization images. Related research has focused only on the recognition of visualization images [20, 39]. Although many methods of steganography have been proposed for natural images, the features of data visualizations are quite distinct. First, most of these models are based on natural images and make use of rich and mature features, while visualization charts usually have clean backgrounds and clear visual elements, increasing the difficulty of encoding and decoding. Second, Bylinskii et al. [11] demonstrated that the visual importance of visualizations depends not only on the image context but also on higher-level factors (e.g., semantic categories), which is different from that of natural images.

We present a novel information embedding approach for visualization images, called VisCode. VisCode is an end-to-end autoencoding method that implicitly encodes information into a visualization image with little visual error. First, we propose to use QR codes as a form of media in which to store information to increase the success rate of decoding. We create a dataset consisting of visualization chart images and QR codes to be used for autoencoder training. Second, we outline a deep neural network architecture based on saliency detection that is designed to perform stable information steganography for visualization images. Information in the form of QR codes is adaptively encoded into the visually nonsignificant areas of a visualization image by means of a saliency-based layout algorithm to reduce the visual traces of this encoding. We also report the successful application of the VisCode method to three practical application scenarios: metadata embedding, source code embedding, and visualization retargeting. Based on the embedded information, the designer can easily manage and modify a graphical chart produced using this method, for example, to generate different versions of the visual design. Additionally, our VisCode system can convert static visualization images into interactive images to facilitate more accurate visualization comprehension. Furthermore, our tool supports visualization retargeting and improvement in accordance with user preferences, such as modification of the color palette style and the form of visualization.

Our experimental results and evaluations show that VisCode has great potential for application in information visualization. In summary, our contributions include three aspects:

- (1) We define the problem of information embedding in the context of information visualization. We verify the practical importance of this study in pioneering a new application domain.
- (2) We outline a deep learning framework that can be used to implement personalized steganography for information visualization images. This framework can achieve high-quality large-scale information hiding.
- (3) We present a set of evaluations to demonstrate the effectiveness of our proposed framework from multiple perspectives, including user friendliness, indices of encoding quality and decoding success, steganography defense, and time performance.

2 RELATED WORK

2.1 Information Steganography

Information steganography, the art of concealing secret data within other carrier data, has numerous applications (e.g., digital watermarking, covert transmission, and access control). The most popular form of steganography is to embed secret information by slightly perturbing carrier images, videos, 3D models, texts, or audio data while ensuring that these perturbations are imperceptible to the human visual system. Instead of processing popular forms of media such as videos and images, COBRA [21] encodes data into a

special type of colorful 2D barcode to improve the efficiency of transmission between a smartphone screen and camera. Yang et al. [57] outlined an effective steganography algorithm using a 3D geometric model as the carrier. Their algorithm hides information in the histogram of the vertex coordinates of the 3D model and can achieve robust performance. Jo et al. [27] utilized light signals from displays and cameras to transmit information. Xiao et al. [56] proposed an interesting method of directly embedding messages into the letters of text. The glyphs of fonts are perturbed to encode secret information, and this information is decoded by recognizing characters by means of an optical character recognition (OCR) library. However, this method has two limitations: different perturbation rules are needed when processing different font types, and sufficient resolution of the text document is necessary when decoding a message. Hota and Huang [4] adopt an image watermarking technique to embed the information into a scientific visualization. It seems that their work is similar to VisCode. However, the variety of data types, representations, and propagations of the information visualizations pose new challenges. The steganography work of information visualization is different from scientific visualization.

In summary, previous research on information steganography has investigated a variety of carriers, but there is less previous study on information steganography for information visualization. In the information visualization field, steganographic applications of personalized information visualization require a specially designed information steganography approach that is different from previous methods.

2.2 Image Steganography

Traditional Steganography Images are the most widely processed and exchanged information carriers in the steganographic domain. Early methods used spatial-domain techniques, such as bit plane complexity segmentation (BPCS) [28] or least-significant-bit (LSB) substitution [33]. These methods involve applying small variations to pixel bits, which are not visually obvious. However, they are easily detected by statistical analysis due to their fixed principles of operation. On the basis of digital signal processing theory, some approaches have been developed for embedding messages in the frequency domain. Almohammad et al. [3] hid information based on the discrete cosine transform (DCT). To avoid detection through statistical analysis, many steganography algorithms define special functions to optimize the localization of embeddings. Highly undetectable steganography (HUGO) [37] minimizes distortion by calculating a variable weight for each pixel based on high-dimensional image models. The Wavelet Obtained Weights (WOW) algorithm [24] measures the textural complexity of different image regions with directional filters.

Deep Steganography Recent studies have achieved impressive results by combining deep learning with steganography. Pibre et al. [38] demonstrated that the results obtained from deep neural networks (DNNs) surpass the conventional results based on handcrafted image features. Several methods utilize a DNN as a component in combination with traditional steganography techniques [25, 48]. More recently, end-to-end steganography networks have become popular. Inspired by autoencoders, Baluja et al. [5] learned feature representations to embed a secret image into a cover image. Hayes et al. [22] utilized adversarial training to generate steganography results. Some studies have also considered sources of corruption in digital transmission, such as JPEG compression [61] and light field messaging (LFM) [53]. Most of these models are based on natural images and make use of rich and mature features. We focus on encoding a large amount of data into images designed for visualization purposes while preserving their visual perceptual quality.

2.3 Retargeting of Information Visualization

Early work on information visualization focused on data representations. Tufte [49] introduced the basic principles of quantitative chart design, emphasizing that the purpose of chart design is to allow users to quickly obtain rich and accurate information and, consequently, decoration that is not related to the data is unnecessary. Tufte's research illustrates the importance of data and the user's understanding of charts in information visualization. In contrast to Tufte, Wilkinson [54] offered a method of describing charts at a higher level of abstraction. This abstract method involves categorizing charts based on the visual characteristics of the data and geometric figures to simplify the architecture of the drawing system. In recent years, scattering, network diagrams, regions, dynamics, metaphors, tools, and

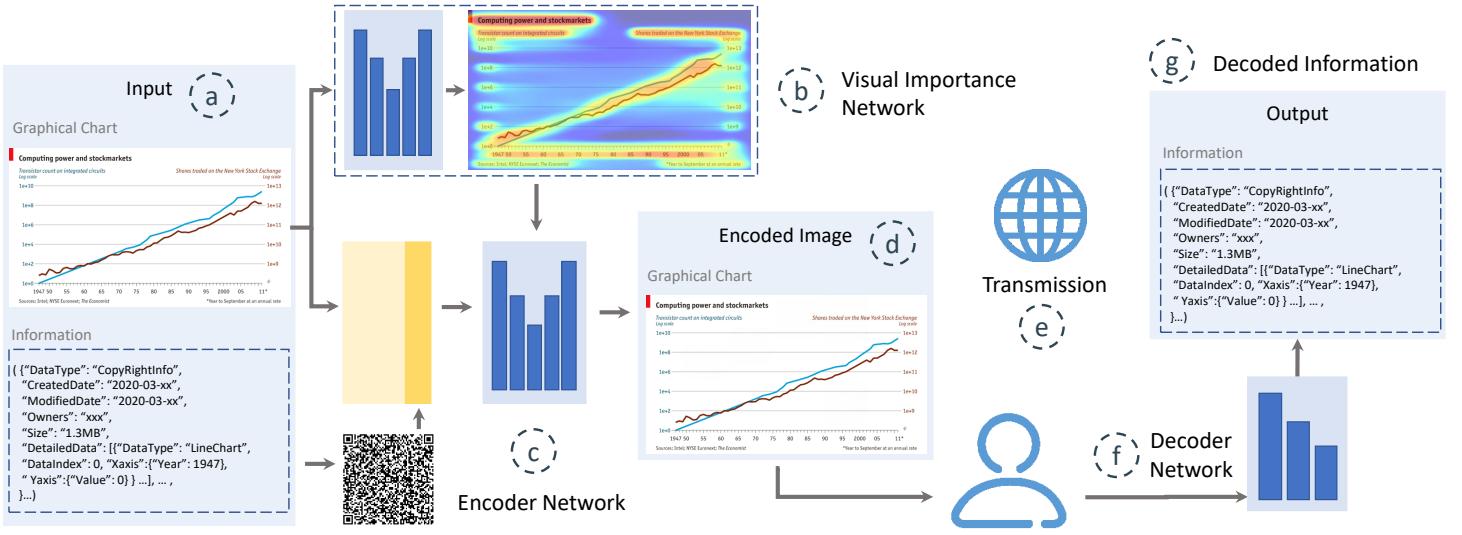


Fig. 2. The main components of our VisCode system. First, the visual importance network (b) processes the input graphical chart (a), which can facilitate salient feature detection of the visualization and output a visual importance map. Next, an encoder network (c) embeds secret information in the graphical chart (a). The carrier chart image and the QR code image are embedded into vectors by the feature extraction process. Then, these two vectors are concatenated (as the yellow rectangle shown) and fed into the auto-encoding stage. The encoded image (d) is then sent to the user (e), and the user can send it to others by digital transmission. When a user wishes to obtain the detailed information hidden in the chart, the encoded image can be uploaded to the decoder network (f). After data recovery and error correction, the user receives the decoded information (g).

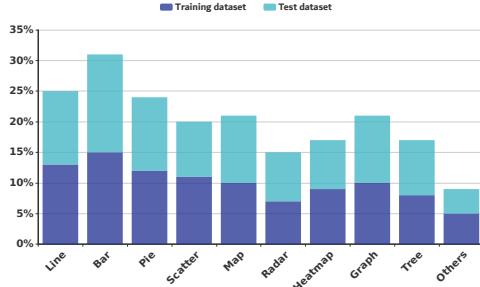


Fig. 3. Distribution of different categories in our visualization dataset.

other visualization techniques have been proposed, and the capabilities of information visualization are becoming increasingly close to ideal. With the development of big data and deep learning technologies, visualization retargeting approaches have been presented to aid in visualization redesign. Poco et al. [40] outlined an approach for extracting color mappings from static visualization images. They evaluated their technique on a variety of visualization types. Later, an end-to-end system was proposed by Poco and Heer [39] for extracting information from static chart images. Similar works include iVoLVER [32] and Chart Decoder [14], and Enamul and Maneesh [16] have similarly offered a deconstruction tool for extracting data and visual encodings from D3 charts. In addition to common usage, functionalities intended to support visually impaired users have also been considered; for example, Choi et al. [12] presented a DNN-based method of extracting the features of a chart to help visually impaired users comprehend charts.

To the best of our knowledge, less previous work has addressed steganography in the context of information visualization. In this paper, we implement visualization retargeting from a novel perspective using information steganography.

3 OVERVIEW

An essential goal of data visualization is to improve the cognition and perception of data. Compared with the direct transmission of a large amount of raw data, images are more easily and widely processed carriers for disseminating visual information on the Internet. However, after the original detailed information (e.g., data points, axis text, and color legends) is

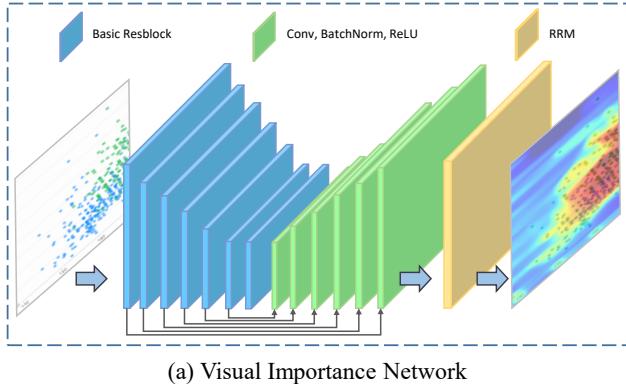
converted into pixel values, it is difficult to reconstruct the original information from the resulting images. To solve this problem, we attempt to encode additional information into the carrier visual designs with minimal perturbation.

In this paper, we propose VisCode, a framework for embedding and recovering hidden information in a visualization image using an encoder-decoder network. Fig. 2 shows the VisCode model, which has 3 main components:

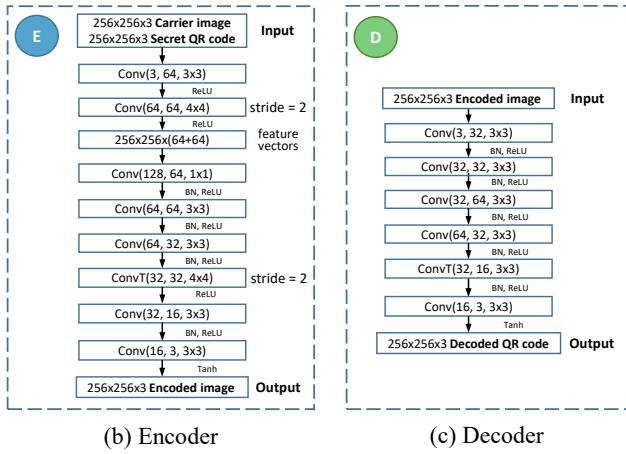
- **Visual Importance Network** The preparation stage includes a visual importance network and a text transformation model. Since the semantic features of data visualizations are quite different from those of natural images, we use a network to model the visual importance of the input data graph. The resulting importance map is applied as a constraint in the *encoder network*. Accounting for error correction, we convert the plain message to be embedded into a series of QR codes instead of binary data.
- **Encoder Network** The encoder network embeds the message within the carrier image and outputs the resulting coded image. In this network component, the carrier chart image and the QR code image are embedded into vectors by the feature extraction process. Then, these two vectors are concatenated and fed into an autoencoder. The resolution of the output coded image is the same as that of the input carrier image.
- **Decoder Network** The decoder network retrieves the information from the coded image. Users can share coded images with others through digital transmission. After such a coded image is decoded, the final information is obtained from the decoded QR code image with error correction.

3.1 Definition

Our VisCode model assumes that, given an information visualization image I_c (also called the carrier image) and a plain message T_s , the coded image I'_c output by the encoder network $Enc()$ should be perceptually similar to I_c according to human vision, while the recovered message T'_s output by the decoder network $Dec()$ should be as accurate as the input T_s . Considering error correction, we convert the input plain message T_s to the QR code image instead of binary data. That is $I_s = PrepQR(T_s)$, with $PrepQR()$ being the preparation of QR code from the text, I_s being the transformed QR code



(a) Visual Importance Network



(b) Encoder

(c) Decoder

Fig. 4. VisCode system contains three network models such as visual importance network, encoder network, and decoder network.

image (also called the secret image). The output of the decoder network is I_s' , and $T_s' = PostQR(I_s')$, where $PostQR()$ is the process of converting a recovered QR code image into text form with an error-correction coding scheme, and T_s' is the result text. Formally, we attempt to learn functions $Enc()$ and $Dec()$ such that:

$$\text{minimize} \|I_c - I_c'\| + \alpha \|I_s - I_s'\| \quad (1)$$

where $Enc(I_c, I_s) = I_c'$ and $Dec(I_c') = I_s'$ represent the two functions of interest. α is a scalar weight.

4 METHODS

4.1 Dataset

Information Visualizations The features of data visualization images are quite different from those of natural images. Hence, to build our data visualization dataset for training, we used several popular chart and visualization libraries, such as D3, Echarts, AntV, and Bokeh, to generate carrier visual designs in 10 categories (e.g., “Bar Chart” or “Scatter Chart”). We also selected a subset of images from MASSVIS [8], a dataset of visualizations collected from a broad variety of social and scientific domains. The resolutions of these images are expressed as $H \times W$, where $H, W \in [300, 3000]$. We selected 1500 images and split them into 1000 images composing the training set and 500 images composing the test set. Fig. 3 shows the distribution of this visualization dataset.

QR codes It is necessary to use an error-correction coding scheme when recovering secret information from the coded images. Shannons [45] foundational work on coding theory for noisy channels has inspired many later studies. Reed et al. [42] proposed a coding scheme based on a polynomial ring. Boneh et al. [6] utilized the Chinese remainder theorem to decode ReedSolomon codes in polynomial time. BCH codes [9] are error-correction codes for binary group data. However, encoding binary data directly into carrier visual designs may result in obvious perturbations of

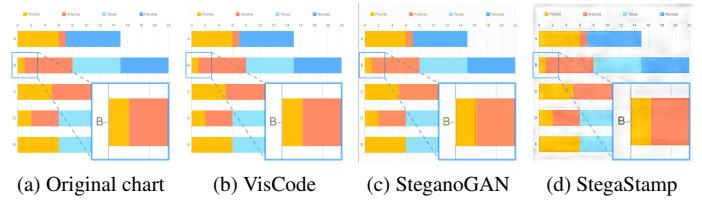


Fig. 5. Comparison of the encoded image quality with other methods.

the original appearance. Fig. 5 presents an illustrative example comparing SteganoGAN [58] and StegaStamp [47]. SteganoGAN [58] utilizes an adversarial training framework to embed arbitrary binary data into images based on ReedSolomon codes. StegaStamp [47] hides hyperlink bitstrings in photos using BCH codes. In Fig. 5, we can see that SteganoGAN may cause color variations in images with rich color content, while StegaStamp may generate visible noise when embedding information. Moreover, visualization charts usually have clean backgrounds and clear visual elements, which leads to a high decoding error rate in SteganoGAN. Due to these limitations, the two methods StegaStamp and SteganoGAN are not suitable for embedding information in data visualizations.

Barcodes are one of the most popular ways to transfer text information between computing devices. In our implementation, we convert arbitrary binary data provided by users into two-dimensional barcodes (QR codes) as our error correction coding scheme. QR codes [52] have essentially the same form as 2D images and therefore offer better visual effects than ReedSolomon codes or BCH codes. QR codes have many advantages, such as a high capacity, a small printout size, and good dirt and damage resistance. We refer the reader to the ISO/IEC 18004 specification [1] for more technical details of QR codes. For our QR code dataset, we chose “Binary” as the information type and randomly generated 1500 QR code images with various character lengths from 1 to 2900 and various error correction capability levels of ‘Low’ (‘L’), ‘Medium’ (‘M’), ‘Quality’ (‘Q’), and ‘High’ (‘H’). Raising this level improves error correction capability, but also increases the size of QR code when embedding the same amount of data.

4.2 Visual Importance Map

At a high level, the goal of the encoder network is to embed a large amount of information into a graphical chart while leaving the coded image perceptually identical to the original. A straightforward solution is to train a model to minimize the mean squared error (MSE) of the pixel difference between the original chart and the encoded chart [5]. However, this metric is unreliable for assessing the visual quality of a chart [50]. For example, the encoder may generate visible noise in important areas since the MSE metric weights all pixels equally. To solve this problem, we introduce a visual importance map as a perceptual constraint to preserve the visual quality of the carrier chart image when embedding information.

Although many methods of saliency prediction have been proposed for natural images, the features of data visualizations are quite distinct. The visual importance of visualizations depends not only on the image context but also on higher-level factors (e.g., semantic categories) [11]. A visual importance map assigns different scores to different regions of an image, which can help to preserve the visual quality of the important content in the encoding stage.

Inspired by Bylinskii et al. [11], we build a prediction model based on the MASSVIS dataset with eye movement annotations [7]. Instead of

Table 1. Module Configurations of QR codes

| Number of Characters | ECC Level | Resolution |
|----------------------|-----------|------------|
| 1 ~ 350 | 'High' | 100 × 100 |
| 351 ~ 1000 | 'Quality' | 200 × 200 |
| 1001 ~ 2000 | 'Medium' | 200 × 200 |
| 2001 ~ 2900 | 'Low' | 300 × 300 |

Table 2. Visual Importance Map Comparison

| Method | Dataset | $CC \uparrow$ | $RMSE \downarrow$ | $R^2 \uparrow$ |
|-----------------------|---------|--|-------------------|----------------|
| Bylinskii et al. [11] | | 0.686 | 0.165 | 0.369 |
| Ours | MASSVIS | \mathcal{L}_{bce} | 0.866 | 0.123 |
| | | \mathcal{L}_{ssim} | 0.866 | 0.126 |
| | | $\mathcal{L}_{bce} + \mathcal{L}_{ssim}$ | 0.868 | 0.117 |
| | | $\mathcal{L}_{bce} + \mathcal{L}_{ssim}$ | 0.868 | 0.117 |

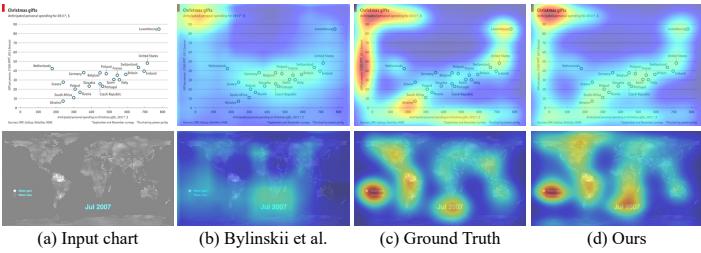


Fig. 6. Visual importance maps predicted by our network are more similar to the distribution of ground truth.

leveraging an original fully convolutional network (FCN) architecture, we fine-tune a state-of-the-art salient object detection network, BASNet [41].

As shown in Fig. 4(a), we use a network architecture the same as BASNet [41], based on U-Net [44] with a multiscale residual refinement module (RRM). Our visual importance network receives a three-channel RGB chart image with a resolution of $H \times W$ as input and outputs a one-channel $H \times W$ visual importance map. Similar to ResNet [23], we use basic ResBlocks to obtain feature maps with different resolutions. Subsequently, we upsample the feature maps from the previous stage and concatenate them. The last feature map is then sent to the RRM for refinement. We refer the reader to the description of BASNet [41] for more details.

The aim of the original BASNet is to detect salient objects, while our goal is to obtain a real-valued visual importance map describing the importance scores of all pixels. Our training process is based on the MASSVIS dataset [7]. Accordingly, to obtain more accurate high-level semantic context information and low-level details, we use a hybrid training loss:

$$\mathcal{L}_{prep} = \mathcal{L}_{bce} + \mathcal{L}_{ssim} \quad (2)$$

where \mathcal{L}_{bce} represents the BCEWithLogits loss [15] and \mathcal{L}_{ssim} denotes the structural similarity index (SSIM) loss [51].

Given the ground-truth importance map at each pixel p , $G_p \in [0, 1]$, over all pixels $p = 1, \dots, N$, the BCEWithLogits loss is defined as:

$$\mathcal{L}_{bce}(G, V) = -\frac{1}{N} \sum_{p=1}^N (G_p \log V_p + (1 - G_p) \log(1 - V_p)) \quad (3)$$

where V_p is the prediction of the visual importance network. The BCEWithLogits loss is defined in a pixelwise manner and facilitates the semantic segmentation of all pixels.

Let $x = \{x_p | p = 1, 2, \dots, N\}$ and $y = \{y_p | p = 1, 2, \dots, N\}$ be two corresponding image patches extracted from the ground-truth importance map G_p and the predicted result V_p , respectively. Let μ_x and μ_y be the means of x and y , let σ_x^2 and σ_y^2 be their variances, and let σ_{xy} be their covariance. The SSIM is defined as:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (4)$$

where $C_1 = 0.01^2$ and $C_2 = 0.03^2$ are scalar constants. The SSIM loss is computed as $\mathcal{L}_{ssim} = 1 - SSIM(x, y)$. This loss can help capture the structural information of the input chart image as the weights of the boundaries of visual elements increase.

The hybrid loss presented here helps us preserve the multiscale features of the visual importance map. Since we aim to get a visual importance map that is similar to the distribution of ground truth with smooth gradient for pixels, we discard IoU loss used in BASNet [41]. For evaluation, we use the

same metrics and test dataset as in the report by Bylinskii et al. [11]. Cross Correlation (CC) is a common metric used for saliency evaluation. Root-Mean-Square Error ($RMSE$) and the R^2 coefficient measure how correlated two maps are. We conduct an ablation study over different loss terms to demonstrate the effectiveness of the hybrid loss. The experimental results are shown in Table 2. Higher CC values, lower $RMSE$ values and higher R^2 values are better. As we can see, the hybrid loss outperforms others. Fig. 6 demonstrates that the visual importance maps predicted by our network are more similar to the ground truth distribution than those of Bylinskii et al. [11].

4.3 Encoder and Decoder

The encoder network is designed to embed information within a carrier chart image while minimizing perturbations in the encoded image. The decoder network recovers the embedded secret information from an encoded image. Inspired by DeepSteg [5], we use a network structure similar to that of an autoencoder. As shown in Fig. 4(b), the feature vectors of the input carrier chart image and the QR code image to be embedded are first extracted using 3×3 filters and 4×4 filters and the ReLU activation function [34]. Then, these two vectors are concatenated and fed to the next 3 convolutional layers, followed by batch normalization (BN) [26] and a ReLU activation function. After upsampling with 4×4 filters (stride = 2) and a ReLU function, the resolution of the output encoded image is the same as that of the input chart image. Then we use two convolution layers with 3×3 filters and a tanh function to obtain the final output encoded image.

The architecture of the decoder network (Fig. 4c) is simpler than that of the encoder network. The encoded chart image is passed through a series of convolutional layers with 3×3 filters to produce the decoded QR code image as the final output. After error correction, we can obtain the reconstructed information in the same form as the original input message.

Different from DeepSteg [5], we introduce the visual importance map to supervise the encoder network. The encoder loss is defined as:

$$\mathcal{L}_{Enc} = V \odot \mathcal{L}_{mse} = \frac{1}{N} \sum_{p=1}^N (V_p \odot (I_{cp} - I_{cp}'')^2) \quad (5)$$

Here, V_p is the predicted visual importance map, which can be regarded as a weight matrix; regions with higher visual importance scores will have higher weights. I_c is the input carrier chart image, and I_c'' is the output encoded image. N is the number of pixels.

The decoder network is supervised with a cross-entropy loss. Thus, we define the joint loss as:

$$\mathcal{L}_{Joint} = \mathcal{L}_{Enc} + \alpha \mathcal{L}_{Dec} = V \odot \mathcal{L}_{mse}(I_c, I_c') + \alpha \mathcal{L}_{mse}(I_s, I_s') \quad (6)$$

where I_s is the original input QR code image and I_s' is the reconstructed result of the decoder network. α is the weight of the decoder loss, which indicates a tradeoff between the visual quality of the encoded images and the decoding accuracy. Higher α results in higher decoding accuracy and decreased perceptual quality. α is set to 0.25 in our implementation.

During the training process, we train the encoder network and decoder network simultaneously. That is, the joint loss incorporates both the visual quality of the coded image and the reconstruction error. In this study, our *encoder-decoder* network needed a smaller structure and fewer parameters to fit the dataset since our visualization data available for training were limited compared with natural images. We trained our encoder and decoder networks using the aforementioned dataset consisting of data visualizations and QR codes. During training, the images were resized to 256×256 with antialiasing. We used the Adam optimizer [29] with an initial learning rate of $lr = 1e^{-3}$. We trained the encoder and decoder networks until the joint loss converged, without utilizing the validation set.

In the testing process, the encoder and decoder can be used separately. Fig. 8 shows an example of results obtained using the encoder. The coded image looks identical to the original chart image since the perturbations in the coded image are unobservable to human perception. We enhance the intensity of the residuals (Fig. 8c) between the input chart image (Fig. 8a) and the output coded image (Fig. 8b) by a factor of 10 times to more clearly show the differences. The results demonstrate that our VisCode model can embed information while preserving the perceptual quality of the original chart image.

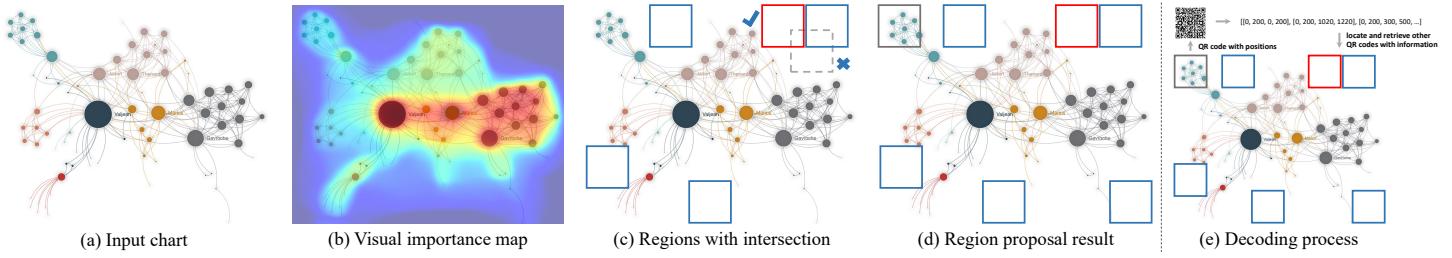


Fig. 7. Illustration of the embedding region proposal algorithm. We localize the optimal embedding regions based on the visual importance map and avoid overlaps of the proposal region boxes. The position QR code makes the decoding process more efficient and accurate.

4.4 Large-Scale Data Embedding

Our VisCode system is designed to take a chart image with a resolution of $H \times W$ and arbitrary binary data as input, making it suitable for various actual scenarios. Embedding a large amount of information in an image is a challenging problem because it is easy to generate obvious noise or cause deviations in color or texture. To overcome this challenge, Wu et al. [55] proposed learning the conditional probability distribution of the original image. However, this method may affect viewers' understanding of a chart since semantic information such as color maps and data points are particularly meaningful in data visualizations. Accordingly, we propose a novel approach for the large-scale embedding of information in regions to which the human visual system is insensitive.

Although arbitrary binary data can be accepted as input information, we focus on a text message to facilitate the explanation. Suppose that the user inputs text with a length of Len_T characters. We first split the text into several blocks:

$$Num_B = \lceil Len_T / \eta \rceil, Len_B = \begin{cases} Len_T / Num_B, \text{ other blocks} \\ Len_T \bmod Num_B, \text{ last block} \end{cases} \quad (7)$$

where Num_B denotes the number of blocks and Len_B denotes the length of each block. Considering that there are various module configurations available for QR codes, which may influence the effects of the *encoder-decoder* network, we conducted tests with various mapping relations between the length of the text and the module configurations of QR codes. Table 1 shows the results. We use these mapping relations as criteria instead of performing dynamic calculations, which reduces the search time. The parameter η can be specified in accordance with the user's preference. A higher η corresponds to more text in each QR code. We set this parameter to 800 by default.

Algorithm 1 Embedding region proposal algorithm.

```

Input:  $V$ : the visual importance map;  $Text$ : the user input text;  $\eta$ : the acceptable maximal number of characters in a single QR code;
Output:  $R_{o\_set} = \{R_{o_1}, R_{o_2}, \dots, R_{o_k}\}$ : a set of optimal embedding regions;
1:  $(W, H) = size(V)$ ,
2: Calculate  $Num_B$  and  $Len_B$  based on Equation 7
3:  $R_{o\_set} = \{\}$ ,  $cnt = 0$ 
4:  $kernel\_size = map(Len_B)$  based on Table 1
5:  $Full_{set} \leftarrow$  Apply average pooling on  $V$ 
6: sort( $Full_{set}$ ) in increasing order
7: for  $c\_region$  in  $Full_{set}$  do
8:   for  $s\_region$  in  $R_{o\_set}$  do
9:     if  $Intersection(c\_region, s\_region) > 0$  then
10:        $conflict = true$ 
11:     end if
12:   end for
13:   if  $conflict = false$  and  $cnt \leq Num_B$  then
14:      $R_{o\_set}.append(c\_region)$ ,  $cnt = cnt + 1$ 
15:   end if
16: end for
17: return  $R_{o\_set}$ 
```

After determining the resolution of each QR code, we attempt to localize the optimal regions of the carrier chart image in which to embed these QR

codes. A straightforward solution is to randomly select several regions, but this may create discontinuities near the boundaries between modified and unmodified regions that will look disharmonious. Another brute-force solution is to list all possible regions and evaluate the effects of the *encoder-decoder* network on each one. However, this may incur a substantial time cost due to the large search space. Instead, inspired by region proposal techniques from the field of object detection (e.g., selective search [18] and RPN [43]), we generate bounding boxes based on the visual importance map instead of the original chart image.

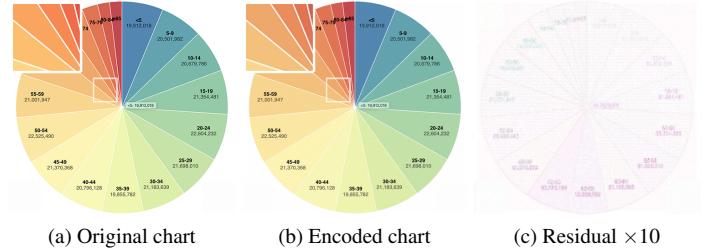


Fig. 8. Comparison of image quality before and after encoding.

The corresponding problem can be formulated as follows. Given a chart image with a resolution of $H \times W$ and a text with a character length of Len_T , we wish to find the Num_B optimal regions with the smallest values in the visual importance map. The size of each region box depends on the mapping relations shown in Table 1. We generate Num_B QR code images containing the Num_B blocks of text to be embedded. Given the visual importance map V_p and the size of each region box, we use average pooling to calculate a value for each region box:

$$R_v = \frac{1}{N_b} \sum_{p=1}^{N_b} V_p \quad (8)$$

where N_b is the kernel size of the filter, which we regard as the size of the region box. Then, we sort the region boxes in increasing order of their values to obtain the list of candidate regions. We preserve the top left region box $[(0, 0), (x_{rd}, y_{rd})]$ for embedding the position information of the other Num_B region boxes. A QR code containing the corresponding position information is also generated. To avoid overlap of the region proposal boxes, we adopt nonmaximum suppression (NMS) [35] based on their intersection values. Fig. 7 shows the key steps of the embedding region proposal algorithm. The detailed region proposal algorithm is presented in Algorithm 1. After determining the optimal regions, we crop the corresponding $Num_B + 1$ patches of the input chart image and feed them into the encoder network with the $Num_B + 1$ QR codes. In the decoding stage, we first extract the top left QR code with the position information from the coded chart image. Then, we locate other corresponding image patches and send them to the decoder network to retrieve the complete information. We utilize the position QR code instead of generating the visual importance map, which is more efficient and accurate.

5 APPLICATIONS

VisCode can be used for a number of applications, such as embedding metadata or source code in visualization design and visualization retargeting.

Experiments related to the envisioned applications were implemented on a PC with an Intel Core i7 CPU, an NVIDIA GeForce 2080 Ti GPU, and 32 GB of memory. The visualization images were generated using D3 [10] and Echarts [30]. The deep learning framework was implemented based on Pytorch [36].

5.1 Metadata Embedding of Visualization Design

Information visualization technology uses computer graphics and image processing techniques to convert data into graphics or images, allowing users to observe and understand data patterns more intuitively. Related to the processes of visualization design, display, and dissemination, there is usually considerable information that needs to be stored, such as the name of the designer, the design institution, the design time, the URL of the visualization web page, and historical revision information. Such related information for a visualization image is called metadata [19].

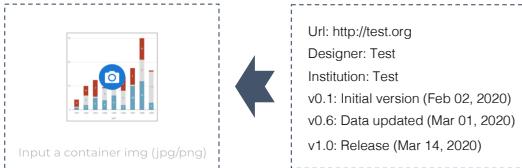


Fig. 9. Application interface for embedding information in a visualization image.

VisCode provides strong support for information steganography in visualization design. First, our information encoding method is implicit and does not affect the original visual design. Sect. 6 will demonstrate the efficiency of our encoding method. Second, the proposed approach has a fast encoding speed and a high decoding success rate. Therefore, it is possible to effectively encode and decode the author information of visual works, the source website, and other such information.

Fig. 9 shows the application interface for embedding information in a visualization image. The input to the VisCode system consists of a visualization image and the corresponding metadata. These metadata may include website links, author names, institutions, and revision logs. The metadata are hidden in the visualization image. Obtaining the metadata through the VisCode decoding process is simple. Using this application, the designer of a visual chart can hide common text information in the chart. On the one hand, the design work is implicitly protected, and on the other hand, there is no need to maintain additional modification log files corresponding to the image. The ability to hide URL links in a chart also provides a convenient way to allow users to view web-based or interactive visualizations. As shown in Fig. 9, after the author hides the URL information in the image, a user who has obtained the image can use the decoding module of VisCode to obtain the URL and access the real-time interactive visualization corresponding to the static image.

5.2 Source Code Embedding in a Visualization

Due to the limitations of network connections and Web servers, the presentation of dynamic web pages is not as flexible as that of static images. Therefore, the display of static images is a very important means of sharing information visualizations. However, static image display has many shortcomings compared with web-based visualization applications. First, a mature information visualization toolbox usually provides multiple interaction modes to help users further explore the contents of data, such as time period selection, data area selection, map level selection in map visualizations, and clicking and dragging operations. Fig. 10 presents several cases of source code embedding in visualizations. The interactive information label display shown in Fig. 10(a) is a frequently used function for information visualization tasks. Second, since static images are mostly stored in bitmap format, when the resolution of an image is not high, it cannot clearly present information, as shown in Fig. 10(b). Third, using the 3D form is a convenient way for the user to observe the patterns in different views. This is not possible with a static visualization image as shown in Fig. 10(c). Fourth, in a visualization involving temporal data, animation is commonly used to visualize data in different periods represented by time steps, as shown in Fig. 10(d). Hans Rosling's dynamic visualization of population and income throughout the world is a representative example [31].

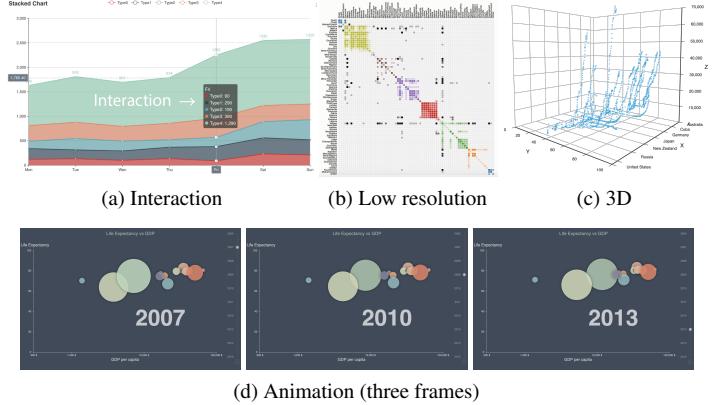


Fig. 10. Three application scenarios involving the embedding of source code in visualizations.

To extend the usage of VisCode, we present an application scenario that provides users with a robust source code embedding function. In this source code embedding scenario, we encode the source code data into a static visualization image. Because the VisCode framework supports very low loss of encoding quality and the encoding of large datasets, it can well support the encoding of source code in information visualizations. For the implementation of this application scenario, we first integrated common information visualization frameworks (such as D3 [10] and Echarts [30]) into the application and then built a functionality for encoding personalized visualization code into a specified image. When a user inputs an encoded image, the system decodes the source code from the image and dynamically generates and displays the visualization file, usually in the form of a web page.

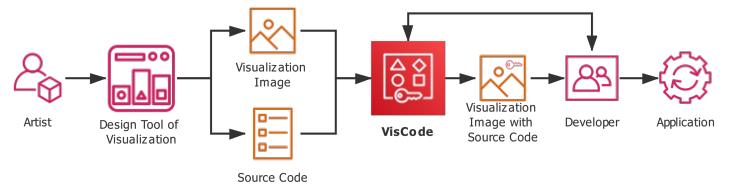


Fig. 11. A pipeline for collaboration between an artist and a developer using the VisCode system.

In addition, source code embedding through VisCode can facilitate collaboration between artists and developers working on visualization projects. In the production processes of large enterprises, artists usually use existing visualization design tools to create visualizations based on aesthetic considerations and then provide the renderings and source code generated by those design tools to developers. Because the interactions between these two roles are frequent and the visual design may undergo many revisions, it can be easy to confuse the design images and codes from different versions. The need to maintain or reconstruct the correspondence relationship between code files and images will increase the difficulty of file sharing. As an alternative approach, VisCode can effectively convert an image and its corresponding source code for a design into a single static image, thus reducing the occurrence of low-level errors. Fig. 11 shows a pipeline for collaboration between an artist and a developer using the VisCode system. The artist can encode the source code into a static visualization image, and the developer can decode the source code and use it in a visualization application.

5.3 Visualization Retargeting

In addition to source code embedding, another application of VisCode is to retarget a visualization. With the development of big data and deep learning technologies, visualization retargeting based on pattern recognition has gained the attention of researchers in recent years, as in the work of Poco et al. [40]. Visualization retargeting is very useful for the creation of visualizations. On the one hand, it reduces the designer's workload. On the other

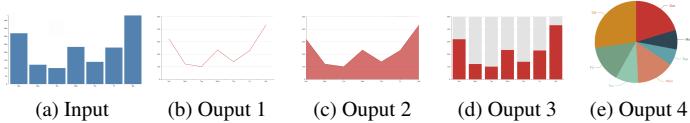


Fig. 12. Retargeting of the form of representation of information. The input visualization can be converted to other visualization forms.

hand, it broadens the artist’s creative space. For some common chart types, existing research has yielded methods of extracting data from a visualization image and retargeting the visualization. For example, Poco et al. [40] presented an approach for identifying values, legends, and coordinate axes from a bar chart. Similar work has also been applied for information extraction and retargeting based on color density maps. However, for many types of visualizations, such as network diagrams, such recognition is still difficult. Visualization retargeting via pattern recognition is not feasible for some visualizations if the visualization process is not irreversible. For example, even if the regions of a density map can be accurately identified, the original scattered data constituting the density map cannot be recovered.

However, if VisCode is used to encode the original data into a static image, then for a visualization that is saved in a certain format, such as JSON, high-quality visualization retargeting can be achieved without pattern recognition. Our implementation includes representation retargeting and theme retargeting. First, we encode the original data of the visualization and the visualization type into a static visualization image in JSON format. After decoding, the data are displayed in a visualization of the specified type. For users, the input is an encoded static image, and the output is a visual effect of different styles, as shown in Fig. 12. We define this form of retargeting as representation retargeting. Second, we can implement different color themes for users to choose based on an existing representation. This capability of color theme switching is called theme retargeting. This application of VisCode can provide rich visual styles and color themes. It can also be used to enable deformations for the visualization of complex network diagrams from which information is difficult to extract, as shown in Fig. 1(a). From an encoded color density map, the original scattered data can be extracted through the decoding function of VisCode. Based on these scattered data and the kernel density algorithm [46], a new density map with a personalized bandwidth can be generated. Fig. 13 presents an example of the retargeting of a visualization of a color density map. Fig. 13(a) shows the input static image. Fig. 13(b) shows the extracted scattered data, and Fig. 13(c-d) show two personalized color density maps.

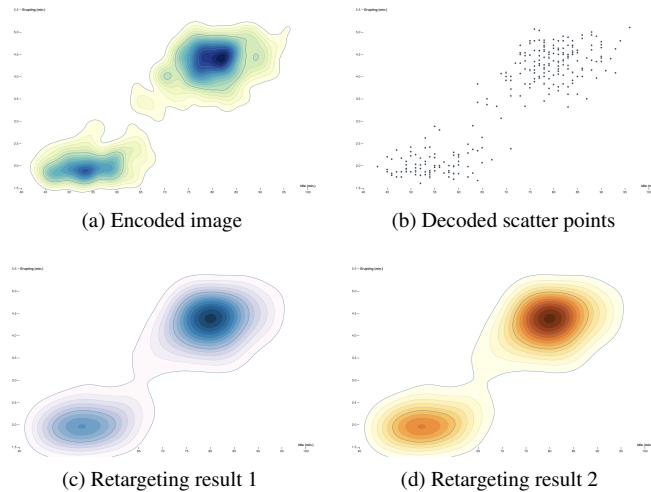


Fig. 13. Retargeting the visualization of a color density map. Scatter points are extracted from the input image and new colorful density maps are generated through the kernel density estimation.

6 EVALUATION

We evaluate our VisCode model from three aspects: *steganography indices*, the evaluation of encoded image quality with the different number of message bits; *steganography defense*, the evaluation of decoding accuracy in various corruption scenarios through digital transmission; and *time performance*, the evaluation of embedding information and recovery time of different settings.

6.1 Steganography Indices

We used three metrics to evaluate the performance of our VisCode model: the peak signal-to-noise ratio (PSNR) [2], the SSIM [50], and the learned perceptual image patch similarity (LPIPS) [59]. The PSNR [2] is a widely used metric for evaluating image distortion. We computed the PSNR by comparing the coded image I_c' and the original chart image I_c , each consisting of N pixels:

$$PSNR = 20 \cdot (\log_{10} P_{\max} - \log_{10} \frac{1}{N} \sum_{p=1}^N (I_{c_p} - I_{c_p}')^2) \quad (9)$$

where P_{\max} is the maximum possible difference between two pixel values, which we set to 1.0 in our experiment since the RGB values of our images are ranged from 0 to 1.

Because the PSNR measures the distance between each pair of pixels independently, we also considered the SSIM to measure the structural similarity between two images. The SSIM is defined as shown in Equation 4. It is a patch-level perceptual metric. However, whereas both the PSNR and SSIM are low-level metrics, Zhang et al. [59] have demonstrated that perceptual similarity is influenced by visual representations instead of being a special function. Accordingly, the LPIPS, which relies on deep features, can better reflect human judgments.

We construct a test dataset of 500 chart images with various resolutions and types, as mentioned in Sect. 4.1. We compare our method with StegaStamp [47] and SteganoGAN [58]. However, there are some limitations to these two methods. StegaStamp [47] can embed only 100 bits data. SteganoGAN [58] can embed small-scale data, but it will cost huge time when the number of bits increases to a threshold. Therefore, the two methods StegaStamp and SteganoGAN are not suitable for embedding large-scale information in data visualizations. For a fair comparison, we resize the chart images to the same resolution, and embed the same numbers of data bits into the test dataset. Higher PSNR, higher SSIM, and lower LPIPS are better. As shown in Table 3, the comparison result demonstrates that VisCode can embed information in various types of chart images with better visual quality. Besides, we design a user study to evaluate the visual effects of the results by human judgments. We select 15 images from the test dataset as the input to generate steganographic results by 3 methods and recruit 30 users to choose the best one. VisCode achieves a mean proportion of being selected of 83.6%, while SteganoGAN achieves 15.1% and StegaStamp achieves 1.3%. Our method is rated higher than others.

Our VisCode system is designed to store an extensive amount of information. Embedding large data is challenging due to the tradeoff between the visual quality of encoded images and the amount of data. Therefore, we evaluate the quality of encoded images with different input text lengths. The resolutions of the test chart images are set to 1600×1600 . Fig. 14 reports the results of our steganography index evaluation, where X-axis represents the number of bits of input text, and Y-axis shows the average values of associated image quality metrics. Our method achieves 29.18 PSNR, 0.9873 SSIM, and 0.0042 LPIPS on average for embedding 204,800 bits of information. It demonstrates that VisCode can encode large-scale data while preserving the perceptual quality of the visualizations.

Table 3. Evaluation results of steganography indices

| Method | Size | 100 bits | | | 3200 bits | | |
|------------|------------------|--------------|---------------|----------------|--------------|---------------|----------------|
| | | $P \uparrow$ | $S \uparrow$ | $L \downarrow$ | $P \uparrow$ | $S \uparrow$ | $L \downarrow$ |
| StegaStamp | 400×400 | 34.09 | 0.9312 | 0.0413 | - | - | - |
| SteganoGAN | | 35.26 | 0.9386 | 0.0247 | 35.19 | 0.9380 | 0.0249 |
| VisCode | | 40.63 | 0.9959 | 0.0007 | 39.19 | 0.9930 | 0.0008 |
| | 800×800 | 42.08 | 0.9990 | 0.0003 | 41.70 | 0.9987 | 0.0004 |

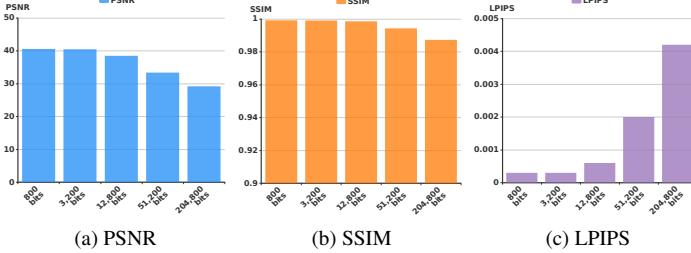


Fig. 14. Steganography index evaluation demonstrates that VisCode can encode large-scale data while preserving the perceptual quality of the visualizations.

Table 4. Evaluation results of steganography defense.

| Method | Watermark | Brightness | Rotation | JPEG |
|----------------|---------------|---------------|-------------|--------------|
| LSB [33] | 0.02% | 0% | 0% | 0% |
| DeepSteg [5] | 3.45% | 32.75% | 100% | 9.8% |
| VisCode (Ours) | 55.29% | 32.79% | 100% | 10.2% |

6.2 Steganography Defense

If we use normal digital transmission without large deformations of the coded images, the decoding text recovery accuracy achieves over 90%. When users share encoded chart images through digital transmission, corruption, or attack may occur. Therefore, we analyze four typical scenarios: adding watermark, adjusting brightness, applying rotation and applying JPEG compression. For each image in the test dataset, we first generate the encoded image with text of 12,000 bits and then apply the following operations: (1) Adding a visible watermark in the same position of the encoded image. (2) Adjusting $\pm 10\%$ brightness of the image. (3) Rotating the image 180° clockwise. (4) Applying JPEG compression to the image with quality factor $Q = 90$.

We use Text Recovery Accuracy (TRA) to assess the performance of our VisCode model in the decoding stage. TRA is defined as the proportion of recovered characters in all input characters. Since it is difficult to embed large-scale data in chart images using StegaStamp [47] and SteganoGAN [58], we compare VisCode with other two methods LSB [33] and DeepSteg [5]. In this part, we maintain the diverse resolutions of the testing dataset from 300×300 to 3000×3000 , as mentioned in Sect. 4.1.

Table 4 shows the comparison result in steganography defense evaluation. LSB is a traditional steganography method using fixed principles, unable to defend any perturbations. We find that the performance of VisCode in adjusting brightness, applying rotation and JPEG compression is close to DeepSteg [5], since the three manipulations apply changes on all pixels. Watermarks apply local changes to the image, which may destroy the important features of the image. In this case, VisCode outperforms DeepSteg [5] by embedding information in insensitive regions.

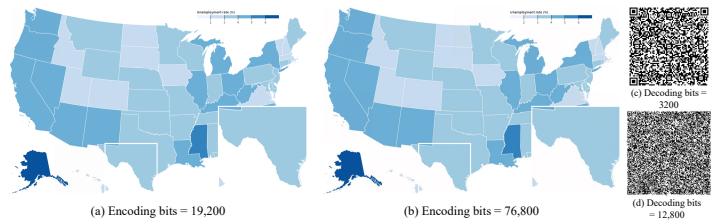


Fig. 16. The sample steganographic images generated by VisCode and decoded results with various bits of data. The input chart image is with a resolution of 420×670 . There are fewer visible artifacts in Fig. 16a than that in Fig. 16b. Moreover, Fig. 16c is decoded correctly, while embedding too much information may cause a high decoding error rate, as shown in Fig. 16d.

6.3 Time Performance

The encoder-decoder network is integrated into VisCode with several other processes for interactive applications. We evaluate time performance for VisCode system of different scales of input chart images and text in Fig. 15. X-axis represents the number of bits of input text, and Y-axis shows the average time (in seconds) taken by the complete encoding stage (Fig. 15a) and decoding stage (Fig. 15b). Lines of different colors represent different resolutions of the input chart images, ranging from 200×200 to 1600×1600 . The environment configurations are introduced in Sect. 5.

In the encoding stage, we observe that the time performance is relatively stable for encoding different lengths of information to images with the same resolution. The complete encoding stage includes predicting the visual importance map of the input chart image, localizing the optimal regions, generating QR codes, sending corresponding image pairs to the encoder network, and obtaining the encoded chart image. When the resolution of images is constant, the increased time is mainly spent on generating QR codes, while the runtime of the encoder network is very fast. On the other hand, the leading cause of different time performance between images with diverse resolutions is the calculation of the visual importance map and the embedding region proposals. In the decoding stage, the time cost of the different resolution of images is similar owing to the fast decoder network, while larger message needs more time to convert QR codes to text with the error-correction scheme.

7 CONCLUSION

We have proposed an approach for encoding information into a static visualization image. Our applications implemented with VisCode are designed to support metadata embedding, source code embedding, and visualization retargeting. We have designed an encoder-decoder network to effectively encode large amounts of information into visualization images such that the original data can be decoded with high accuracy. The presented applications and evaluations demonstrate that VisCode can offer intuitive and effective information embedding for various types of visualizations.

The current version of VisCode has certain limitations in addressing large deformations of the coded images. In addition, for a fixed visualization image size, there is an upper limit on the amount of information that can be embedded, which reduces the effectiveness of VisCode. If we embed too much information in a chart image, which corresponds to more text in each QR code and lower error correction capability, it may cause a high decoding error rate. With the same image size and encoded data bits, images with more complex texture and rich color content are easier to decode successfully.

In the future, we plan to propose a more flexible framework to ensure a high decoding success rate while adapting to visualization image deformation. Camera capturing and light-aware steganography, as shown in the work of Wengrowski and Dana [53], is a potential research direction that can extend the application domain of VisCode. We also plan to collaborate with designers and developers to obtain feedback in the actual design and manufacturing applications [60] to improve the user experience of VisCode.

ACKNOWLEDGMENTS

The authors wish to acknowledge the support from NSFC under Grants (No. 61802128, 61672237, 61532002).

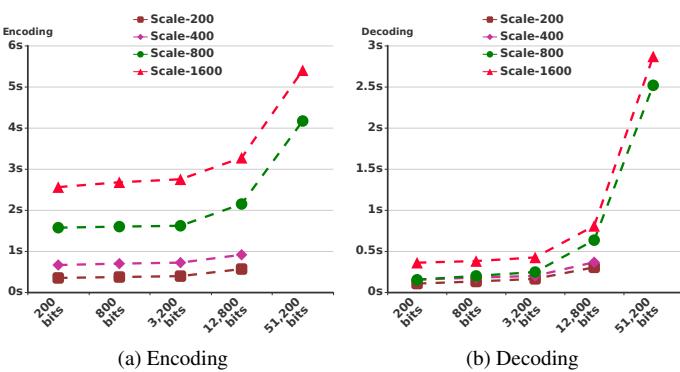


Fig. 15. Time performance comparison of encoding and decoding.

REFERENCES

- [1] Information technology automatic identification and data capture techniques qr code bar code symbology specification. *ISO/IEC 18004:2015*, 2015.
- [2] A. Almohammad and G. Ghinea. Stego image quality and the reliability of psnr. In *2010 2nd International Conference on Image Processing Theory, Tools and Applications*, pp. 215–220. IEEE, 2010.
- [3] A. Almohammad, R. M. Hierons, and G. Ghinea. High capacity steganographic method based upon jpeg. In *2008 Third International Conference on Availability, Reliability and Security*, pp. 544–549. IEEE, 2008.
- [4] H. Alok and H. Jian. Embedding meta information into visualizations. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–15, 2019.
- [5] S. Baluja. Hiding images in plain sight: Deep steganography. In *Advances in Neural Information Processing Systems*, pp. 2069–2079, 2017.
- [6] D. Boneh. Finding smooth integers in short intervals using crt decoding. *Journal of Computer and System Sciences*, 64(4):768–784, 2002.
- [7] M. A. Borkin, Z. Bylinskii, N. W. Kim, C. M. Bainbridge, C. S. Yeh, D. Borkin, H. Pfister, and A. Oliva. Beyond memorability: Visualization recognition and recall. *IEEE transactions on visualization and computer graphics*, 22(1):519–528, 2015.
- [8] M. A. Borkin, A. A. Vo, Z. Bylinskii, P. Isola, S. Sunkavalli, A. Oliva, and H. Pfister. What makes a visualization memorable? *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2306–2315, 2013.
- [9] R. C. Bose and D. K. Ray-Chaudhuri. On a class of error correcting binary group codes. *Information and control*, 3(1):68–79, 1960.
- [10] M. Bostock, V. Ogievetsky, and J. Heer. D³ data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309, 2011.
- [11] Z. Bylinskii, N. W. Kim, P. O’Donovan, S. Alsheikh, S. Madan, H. Pfister, F. Durand, B. Russell, and A. Hertzmann. Learning visual importance for graphic designs and data visualizations. In *Proceedings of the 30th Annual ACM symposium on user interface software and technology*, pp. 57–69, 2017.
- [12] J. Choi, S. Jung, D. G. Park, J. Choo, and N. Elmquist. Visualizing for the non-visual: Enabling the visually impaired to use visualization. In *Computer Graphics Forum*, vol. 38, pp. 249–260. Wiley Online Library, 2019.
- [13] I. Cox, M. Miller, J. Bloom, J. Fridrich, and T. Kalker. *Digital watermarking and steganography*. Morgan kaufmann, 2007.
- [14] W. Dai, M. Wang, Z. Niu, and J. Zhang. Chart decoder: Generating textual and numeric information from chart images automatically. *Journal of Visual Languages & Computing*, 48:101–109, 2018.
- [15] P.-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.
- [16] H. Enamul and M. Agrawala. Searching the visual style and structure of d3 visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):1236–1245, 2020.
- [17] H. M. Ghadirli, A. Nodehi, and R. Enayatifar. An overview of encryption algorithms in color images. *Signal Processing*, 2019.
- [18] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.
- [19] J. Greenberg. Understanding metadata and metadata schemes. *Cataloging classification quarterly*, 40(3-4):17–36, 2005.
- [20] D. Haehn, J. Tompkin, and H. Pfister. Evaluating graphical perception with cnns. *IEEE transactions on visualization and computer graphics*, 25(1):641–650, 2018.
- [21] T. Hao, R. Zhou, and G. Xing. Cobra: color barcode streaming for smartphone systems. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*, pp. 85–98, 2012.
- [22] J. Hayes and G. Danezis. Generating steganographic images via adversarial training. In *Advances in Neural Information Processing Systems*, pp. 1954–1963, 2017.
- [23] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- [24] V. Holub and J. Fridrich. Designing steganographic distortion using directional filters. In *2012 IEEE International workshop on information forensics and security (WIFS)*, pp. 234–239. IEEE, 2012.
- [25] S. Husien and H. Badi. Artificial neural network for steganography. *Neural Computing and Applications*, 26(1):111–116, 2015.
- [26] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.
- [27] K. Jo, M. Gupta, and S. K. Nayar. Disco: Display-camera communication using rolling shutter sensors. *ACM Transactions on Graphics (TOG)*, 35(5):1–13, 2016.
- [28] E. Kawaguchi and R. O. Eason. Principles and applications of bpcs steganography. In *Multimedia Systems and Applications*, vol. 3528, pp. 464–473. International Society for Optics and Photonics, 1999.
- [29] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [30] D. Li, H. Mei, Y. Shen, S. Su, W. Zhang, J. Wang, M. Zu, and W. Chen. Echarts: A declarative framework for rapid construction of web-based visualization. *Visual Informatics*, 2(2):136–146, 2018.
- [31] A. Maxmen. Three minutes with hans rosling will change your mind about the world. *Nature News*, 540(7633):330, 2016.
- [32] G. G. Méndez, M. A. Nacenta, and S. Vandenhende. ivolver: Interactive visual language for visualization extraction and reconstruction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 4073–4085, 2016.
- [33] J. Mielikainen. Lsb matching revisited. *IEEE signal processing letters*, 13(5):285–287, 2006.
- [34] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.
- [35] A. Neubeck and L. Van Gool. Efficient non-maximum suppression. In *18th International Conference on Pattern Recognition (ICPR’06)*, vol. 3, pp. 850–855. IEEE, 2006.
- [36] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.
- [37] T. Pevny, T. Filler, and P. Bas. Using high-dimensional image models to perform highly undetectable steganography. In *International Workshop on Information Hiding*, pp. 161–177. Springer, 2010.
- [38] L. Pibre, J. Pasquet, D. Ienco, and M. Chaumont. Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover sourcemismatch. *Electronic Imaging*, 2016(8):1–11, 2016.
- [39] J. Poco and J. Heer. Reverse-engineering visualizations: Recovering visual encodings from chart images. In *Computer Graphics Forum*, vol. 36, pp. 353–363. Wiley Online Library, 2017.
- [40] J. Poco, A. Mayhua, and J. Heer. Extracting and retargeting color mappings from bitmap images of visualizations. *IEEE transactions on visualization and computer graphics*, 24(1):637–646, 2017.
- [41] X. Qin, Z. Zhang, C. Huang, C. Gao, M. Dehghan, and M. Jagersand. Basnet: Boundary-aware salient object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7479–7489, 2019.
- [42] I. S. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the society for industrial and applied mathematics*, 8(2):300–304, 1960.
- [43] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pp. 91–99, 2015.
- [44] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.
- [45] C. E. Shannon. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423, 1948.
- [46] B. W. Silverman. *Density estimation for statistics and data analysis*, vol. 26. CRC press, 1986.
- [47] M. Tancik, B. Mildenhall, and R. Ng. Stegastamp: Invisible hyperlinks in physical photographs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2117–2126, 2020.
- [48] W. Tang, S. Tan, B. Li, and J. Huang. Automatic steganographic distortion learning using a generative adversarial network. *IEEE Signal Processing Letters*, 24(10):1547–1551, 2017.
- [49] E. R. Tufte. *The visual display of quantitative information*, vol. 2. 2001.
- [50] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.
- [51] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale structural similarity for image quality assessment. In *The Thirtieth Asilomar Conference on Signals, Systems & Computers, 2003*, vol. 2, pp. 1398–1402. Ieee, 2003.
- [52] D. WAVE. Qr code. <https://www.qrcode.com>.
- [53] E. Wengrowski and K. Dana. Light field messaging with deep photographic steganography. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1515–1524, 2019.
- [54] L. Wilkinson. The grammar of graphics. In *Handbook of Computational Statistics*, pp. 375–414. Springer, 2012.

- [55] P. Wu, Y. Yang, and X. Li. Stegnet: Mega image steganography capacity with deep convolutional network. *Future Internet*, 10(6):54, 2018.
- [56] C. Xiao, C. Zhang, and C. Zheng. Fontcode: Embedding information in text documents using glyph perturbation. *ACM Transactions on Graphics (TOG)*, 37(2):1–16, 2018.
- [57] Y. Yang, R. Pintus, H. Rushmeier, and I. Ivrissimtzis. A 3d steganalytic algorithm and steganalysis-resistant watermarking. *IEEE transactions on visualization and computer graphics*, 23(2):1002–1013, 2016.
- [58] K. A. Zhang, A. Cuesta-Infante, L. Xu, and K. Veeramachaneni. Steganogan: high capacity image steganography with gans. *arXiv preprint arXiv:1901.03892*, 2019.
- [59] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 586–595, 2018.
- [60] F. Zhou, X. Lin, C. Liu, Y. Zhao, P. Xu, L. Ren, T. Xue, and L. Ren. A survey of visualization for smart manufacturing. *Journal of Visualization*, 22(2):419–435, 2019.
- [61] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei. Hidden: Hiding data with deep networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 657–672, 2018.