# Exploring Diffusion-Inspired Pixel Predictors for WS Steganalysis

Martin Beneš
martin.benes@uibk.ac.at
University of Innsbruck
Innsbruck, Austria

Rainer Böhme
rainer.boehme@uibk.ac.at
University of Innsbruck
Innsbruck, Austria

## ABSTRACT

Analytical estimators of the steganographic change rate in images, such as WS steganalysis, often operate on the noise residual. The residual can be obtained by estimating the cover content with pixel predictors and subtracting it from the image under analysis. In recent years, we have witnessed the success of new deep learning-based denoisers, such as U-Net, in various fields of image processing. In this study, we revisit WS steganalysis using a U-Net variant as a drop-in replacement for the linear filters originally proposed for cover prediction. A novel property of this U-Net variant is its hand-crafted loss function, which ensures that when predicting from stego images, the prediction errors are uncorrelated with the stego noise, an assumption required by WS steganalysis. Improving especially in the textured regions, the proposed predictor produces accurate and consistent change rate estimates. When used as a detector, our model significantly reduces false positives and thus potentially sets a new baseline for LSB replacement steganalysis.

## CCS CONCEPTS

• **Security and privacy**; • **Computing methodologies** → **Neural networks**; **Image processing**;

## KEYWORDS

WS steganalysis; LSB replacement; U-Net; pixel prediction

**ACM Reference Format:**
Martin Beneš and Rainer Böhme. 2024. Exploring Diffusion-Inspired Pixel Predictors for WS Steganalysis. In *Proceedings of the 2024 ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec '24), June 24–26, 2024, Baiona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3658664.3659645

## 1 INTRODUCTION

Steganography is the art and science of hiding secret messages inside innocently looking cover objects. A common approach to steganographic embedding is to modify an existing cover, e.g., pixels of a digital image, such that an adversary, the steganalyst, cannot distinguish between the cover and stego objects. Least significant bit replacement (LSBR), known since the early 1990s, is an embedding scheme replacing the least significant bit (LSB) of pixels with a message bit [20, 48]. Many steganographic tools available on the internet still use LSBR [8].

The steganalysis literature has introduced a plethora of detection methods targeted against LSBR, such as the $\chi^2$-test [48], Regular–Singular (RS) analysis [22], Sample Pair Analysis (SPA) [18], Weighted Stegoimage (WS) [21], or Triples analysis [28]. Subsequent advances in steganography [19, 39] have taken steganalysis from analytical detectors toward machine learning with handcrafted feature vectors [23, 26, 34, 44], and later to deep neural networks [2, 6, 10, 11, 51]. Analytical detectors and feature vectors often operate on a noise residual, and thus rely on accurate content removal, typically involving a pixel predictor, such as denoisers, filter banks, single-pixel inpainting models, etc. [21, 26, 44].

In the related field of image forensics, deep learning-based denoisers were a game changer, e.g., for camera identification [16, 33, 52]. In synthetic image generation, we see the success of denoisers at the core of diffusion models [25]. The architecture that particularly stands out is U-Net, used in the image generators DALL-E [3], or Stable Diffusion [40]. It is natural to ask whether these pixel predictors can improve steganalysis, similar to the fields above.

Our contributions in this paper are

(1) a systematic comparison of pixel predictors,
(2) the first use of U-Net in steganalysis, and
(3) improved WS steganalysis against LSBR by using a U-Net-based predictor.

This paper is structured as follows. Section 2 recalls the background on LSBR steganography, WS steganalysis, and the U-Net architecture. Section 3 describes the dataset and the models used for the experiments. Section 4 presents the results for the detector, and Sec. 5 provides a discussion. Section 6 summarizes the state of the art in WS steganalysis and learning-based pixel prediction, and Sec. 7 concludes.

## 2 BACKGROUND

### 2.1 Notation

We typeset scalars in a regular font, $x$, vectors in bold, $\boldsymbol{x}$, and vector elements with index in subscript, $x_i$. An image $\boldsymbol{x}$ has $N$ elements, called pixels in the spatial domain. Its $i^{\text{th}}$ pixel is denoted $x_i$, $i \in \{1, \dots, N\}$. From a cover image $\boldsymbol{x}^{(0)}$, we can create a stego image $\boldsymbol{x}^{(\alpha)}$ by embedding a payload of length $M$ and relative length $\alpha = \frac{M}{N}$, also called the embedding rate. The difference between the cover and the stego image, $\boldsymbol{\delta} = \boldsymbol{x}^{(\alpha)} - \boldsymbol{x}^{(0)}$, is called the change vector or stego noise. It has range $\delta_i \in \{0, \pm 1\}$ if we embed at most one bit per pixel. The embedding makes $\sum_{i=1}^{N} |\delta_i| = N\beta$ changes to the cover, where the change rate $\beta$ can be interpreted as the average



*IH&MMSec '24, June 24–26, 2024, Baiona, Spain*

ACM ISBN 979-8-4007-0637-0/24/06.
https://doi.org/10.1145/3658664.3659645

**Table 1: Comparison of embedding rates $\alpha$ of LSB replacement, with and without perfect syndrome coding. Perfect coding allows much longer messages to be embedded with the same number of changes.**

| Embedding rate | Efficiency of perfect syndrome coding | Efficient $\alpha$ assuming perfect syndrome coding |
|---|---|---|
| $\alpha$ | $\frac{\alpha}{H^{-1}(\alpha)}$ | $\frac{\alpha}{2} \cdot \frac{\alpha}{H^{-1}(\alpha)}$ |
| 0.200 | 6.423 | 0.642 |
| 0.100 | 7.690 | 0.384 |
| 0.050 | 8.894 | 0.222 |
| 0.010 | 11.744 | 0.059 |

probability of changing a pixel. The embedding efficiency, defined as $e = \frac{\alpha}{\beta}$, relates the embedding rate to the change rate.

For content-adaptive steganography, the probabilities of changing different pixels can vary depending on the cover content. Thus, the previously scalar change rate becomes a vector $\boldsymbol{\beta}$, with $\beta_i$ being the probability of changing the $i^{\text{th}}$ pixel. The vector $\boldsymbol{\beta}$ connects to the scalar $\beta$, s.t., $\beta = \frac{1}{N}\sum_{i=1}^{N}\beta_i$.

The loss functions used for training are based on the mean absolute error $\text{MAE}(x;y) = |y - x|$ for scalars and $\text{MAE}(\boldsymbol{x};\boldsymbol{y}) = \frac{1}{N}\sum_{i=1}^{N}|y_i - x_i|$ for vectors. The semicolons in the loss notation separate data available to the steganalyst from ground truth.

## 2.2 LSB replacement steganography

LSB replacement is an embedding operation defined as,

$$x_i^{(\alpha)} = x_i^{(0)} + |\delta_i| \cdot (-1)^{x_i^{(0)}}. \tag{1}$$

The embedding positions, where the absolute value of the change vector are one, follow a $|\delta_i| \overset{\text{i.i.d.}}{\sim} \text{Bernoulli}(\beta_i)$ distribution. The change direction depends on the parity of the cover pixel: LSBR increments even values and decrements odd values. The property that the cover pixel determines a single possible change direction allows for powerful analytical attacks, such as WS [21].

Many older papers do not consider coding and report the results for the embedding efficiency $e = 2$, i.e., $\beta = \frac{\alpha}{2}$, because there is a 50% chance that the cover LSB is set to the value needed. More recent steganography research uses syndrome coding [19], which assumes that $e$ is at its upper bound, $e \leq \frac{\alpha}{H^{-1}(\alpha)}$, where $H^{-1}(\cdot)$ is the inverse of the binary entropy, $H(p) = -p\log(p) - (1-p)\log(1-p)$ [47]. Thus, steganography with perfect coding makes fewer changes than standard LSBR for the same reported embedding rate. This difference can be especially relevant for quantitative steganalysis, which tries to estimate the embedding rate instead of just distinguishing between cover and stego [20]. Table 1 allows the reader to translate our embedding rates $\alpha$ to the embedding rates in the literature that assumes perfect syndrome coding. For instance, LSBR at $\alpha = 0.2$ makes the same number of changes as embedding with perfect coding at $\alpha = 0.642$.

## 2.3 Adaptive LSBR

The choice of $\boldsymbol{\beta}$ controls the position of changes. Standard LSBR uses permutative straddling, i.e., constant $\beta_i = \frac{\alpha}{2}, \forall i$. Adaptivity can be introduced by conditioning $\beta_i$ on the image content through an adaptivity criterion (or cost function) $\zeta\left(\boldsymbol{x}^{(0)}\right)$, s.t., $\zeta\left(\boldsymbol{x}^{(0)}\right) \overset{\alpha}{\mapsto} \boldsymbol{\beta}$, where $\overset{\alpha}{\mapsto}$ is a mapping parameterized by the embedding rate $\alpha$.

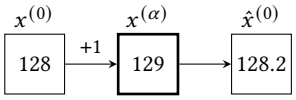



**Figure 1: Example with building blocks of WS steganalysis. The input is the stego image (thick border). The cover is unknown and the cover prediction is derived from the stego image.**

*HILL.* HIgh-pass Low-pass Low-pass (HILL) [36] is a state-of-the-art steganographic cost function defined as,

$$\zeta_{\text{HILL}}(\boldsymbol{x}^{(\alpha)}) = \frac{1}{\left|\boldsymbol{x}^{(\alpha)} * \mathcal{F}_{\text{KB}}\right| * \mathcal{F}_{\text{AVG}}^{(3\times3)}} * \mathcal{F}_{\text{AVG}}^{(15\times15)}. \tag{2}$$

As the name suggests, the input is convolved by three filters: a high-pass Ker–Böhme (KB) filter $\mathcal{F}_{\text{KB}}^{(3\times3)}$ [31], and two low-pass averaging filters. The absolute value $|\cdot|$ and the reciprocal $\frac{1}{\boldsymbol{x}}$ are computed element wise. Fig. 2 shows the filters in Eqs. (4) and (5).

## 2.4 WS steganalysis

Weighted stegoimage (WS) is a steganalysis method to estimate the change rate $\beta$ from the stego image with LSBR. The estimated change rate $\hat{\beta}$ is given by the covariance between the change direction of LSBR and the prediction direction,

$$\hat{\beta} = \max\Big(0, \frac{1}{N}\underbrace{(\bar{\boldsymbol{x}}^{(\alpha)} - \boldsymbol{x}^{(\alpha)})^T}_{\text{change direction}}\ \underbrace{(\hat{\boldsymbol{x}}^{(0)} - \boldsymbol{x}^{(\alpha)})}_{\text{prediction direction}}\Big), \tag{3}$$

where $\bar{\boldsymbol{x}}^{(\alpha)}$ is the stego with its LSBs flipped and $\hat{\boldsymbol{x}}^{(0)}$ is the predicted cover derived from the stego $\boldsymbol{x}^{(\alpha)}$ using a pixel predictor. In the presence of the predicted cover $\hat{\boldsymbol{x}}^{(0)}$, we call $\boldsymbol{x}^{(0)}$ the true cover. In contrast to [21, 31], Eq. (3) estimates the change rate $\hat{\beta}$ instead of the embedding rate $\hat{\alpha}$. Moreover, the order of the subtractions is changed for easier interpretability while maintaining the mathematical equivalence.

Figure 1 shows an example to build an intuition for WS. The cover element be $x^{(0)} = 128$, changed by the LSBR embedding to $x^{(\alpha)} = 129$. The flipped stego value is $\bar{x}^{(\alpha)} = 128$, and the pixel predictor yields $\hat{x}^{(0)} = 128.2$. The change direction is $-1$, the prediction direction is $-0.8$ and the prediction error is 0.2. In an LSBR stego image, the signs of the change direction and the prediction direction match more often than in a cover image.

Usual pixel predictors are linear filters with the central weight 0: $3 \times 3$ averaging filter AVG' [21] from Eq. (6), KB filter from Eq. (4), or an adaptive linear filter [31].

We compare the pixel predictors in how close the predicted covers are to the true covers, using the prediction error, $\hat{\boldsymbol{x}}^{(0)} - \boldsymbol{x}^{(0)}$. Linear filters are good pixel predictors, but their performance worsens at edges and in the textured areas [21].

$$\mathcal{F}_{\text{KB}} = \frac{1}{4}\begin{bmatrix} -1 & +2 & -1 \\ +2 & 0 & +2 \\ -1 & +2 & -1 \end{bmatrix} \quad (4) \qquad \mathcal{F}^{k\times k}_{\text{AVG}} = \frac{1}{k^2}\begin{bmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \\ 1 & & 1 \end{bmatrix} \quad (5) \qquad \mathcal{F}^{3\times 3}_{\text{AVG}'} = \frac{1}{8}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (6)$$

**Figure 2: Linear filters used in this paper: KB filter (Eq. (4)), averaging filter AVG (Eq. (5)), and averaging filter AVG' with zero center weight (Eq. (6)).**

*2.4.1 Weighted WS.* The weighted variant of WS accounts for the variability in predictability by introducing weights $\boldsymbol{w}$,

$$\hat{\beta}_{\text{w}} = \max\Big(0, \sum_{i=1}^{N} w_i \left(\bar{x}_i^{(\alpha)} - x_i^{(\alpha)}\right)\left(\hat{x}_i^{(0)} - x_i^{(\alpha)}\right)\Big). \quad (7)$$

The weighted WS amplifies the evidence from flat areas and attenuates the evidence from textured areas. The weights $\boldsymbol{w}$ can be chosen freely, e.g., inversely proportional to local variance $\boldsymbol{\sigma}^2$ or as an inverse of a steganographic adaptivity criterion $\zeta(\cdot)$, and are subject to $\sum_{i=1}^{N} w_i = 1$. A common choice is $w_i \propto \frac{1}{5+\sigma_i^2}$ [21, 31].

*2.4.2 Requirements.* For WS to be a consistent estimator of the change rate, several assumptions must hold.

*LSBR steganography.* LSBR can change each cover element only in one possible change direction, as seen from Eq. (1). It is this property that WS exploits. Embedding operations that draw a random change direction for each element cannot be detected with WS.

*Prediction error uncorrelated with the stego noise.* The cover prediction error and the stego noise $\boldsymbol{\delta}$ must be uncorrelated, as shown in Eq. (8),

$$\underbrace{\text{corr}(\boldsymbol{\delta}, \hat{\boldsymbol{x}}^{(0)} - \boldsymbol{x}^{(0)})}_{\triangleq \rho} = 0. \quad (8)$$

By contrast, subtracting the prediction from the stego image $x^{(\alpha)}$ would unduly amplify or attenuate the stego noise, which adds a bias to $\hat{\beta}$, weakening its consistency [5, p. 75]. In practice, this requirement can be violated by accidentally including the center element of the linear filter into the prediction.

*Uniform payload distribution.* Weighted WS assumes a uniform distribution of embedding changes, a property of steganography with permutative straddling. Adaptive methods concentrate the embedding changes in the textured areas, which, due to the weighting, are largely ignored by the steganalyst. If the steganalyst suspects adaptive LSBR, weighted WS should be avoided [].

## 2.5 U-Net

U-Net [41] is a fully convolutional network originally proposed for medical image segmentation. The main idea is to subsample the image multiple times, and process different resolutions separately. Then the image is upsampled back step-by-step, combining the local information from the current resolution with the global information from the upsampled lower resolution. Therefore it is a natural choice for the pixel prediction task in WS steganalysis. The local information can be directly derived from the neighbors and the global information allows the network to adjust the prediction to the wider context.

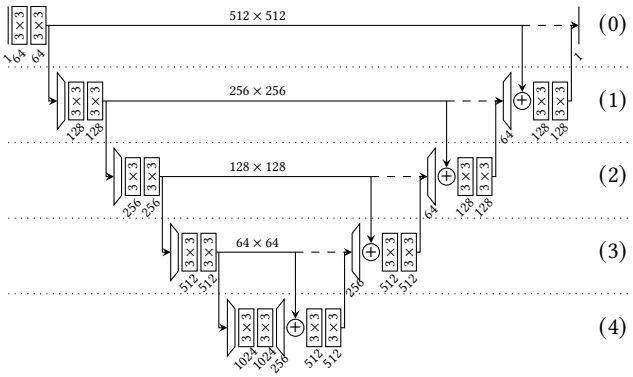



**Figure 3: U-Net architecture from [41], consisting of convolutional layers, concatenation, max-pooling for downsampling, and transposed convolution for upsampling.**

Figure 3 visualizes the U-Net architecture. The network consists of the encoding and decoding parts, with 5 resolution levels, denoted (0)-(4), each containing a skip connection. Each level of the encoding part contains two $3 \times 3$ 2D convolutional layers with symmetric padding and a ReLU activation function. The resolution is reduced by a $2\times2$ max-pooling layer, at the entry to the next level. The decoding part consists of $2 \times 2$ transposed convolutional layers, which perform the upsampling, with zero-padding and without an activation function. The skip connections are realized by taking the activations before the max-pooling and concatenating them along the channel axis with the upsampled activations. The concatenated activations are then processed by $3\times3$ 2D convolutional layers with ReLU activation functions. The final layer is a $1 \times 1$ convolution with the desired number of output channels at the resolution of the input. A sigmoid activation converts the values to the range $[0, 1]$ [41].

The U-Net design has often been modified for specific tasks [43]. In this paper, we use a subset of U-Net, U-Net$^{(k)}$, $k \in \{0, \dots, 4\}$, with only $k$ subsampling and upsampling levels. In Fig. 3, this is denoted by the dashed connections, e.g., for U-Net$^{(2)}$, the dashed connection at level (2) is active and the solid connections with level (3) are detached. This modification shrinks the network substantially, reducing the number of weights and its receptive field. This is acceptable for steganalysis because pixel prediction is a local neighborhood task.

# 3 EXPERIMENTS

The objective of this paper is to explore how a U-Net based pixel predictor can improve WS steganalysis. We proceed experimentally

and describe our setup in this section: dataset (Sec. 3.1), steganography (Sec. 3.2), models (Sec. 3.3) and how we train them (Sec. 3.4), as well as performance metrics used for the evaluation (Sec. 3.5).

## 3.1 Data

The experiments are carried out on the BOSSBase dataset [1], consisting of 10 000 grayscale images of size $512^2$. We split the dataset and use 6 000 images for training, 2 000 images for validation and hyperparameter search, and the remaining 2 000 for evaluation.

To evaluate the models' generalization capabilities, we draw 2 000 raw images from the ALASKA2 dataset [11], process them with a demosaicking method randomly chosen from DCRAW's options {ahd, `linear`, ppg, vng} [9], crop them to $512^2$ using smart crop [10], and finally convert them to grayscale.

## 3.2 Steganography

We simulate steganographic embedding with LSBR and HILLR at relative payloads $\alpha \in \{0.2, 0.1, 0.05, 0.01\}$, which follows the setup from [42]. HILLR is an adaptive LSBR variant with the HILL cost from Eq. (2) as the adaptivity criterion. Similarly to NUGO [42], LSBR is applied to elements with HILL costs lower than the $\alpha$-quantile of the cost map, with $\alpha$ being the relative payload.

## 3.3 Models

*3.3.1 Linear filters.* As the experimental baseline, we use two linear filters, the $3 \times 3$ filter $\mathcal{F}^{(3\times3)}_{\text{AVG}'}$ from Eq. (6) (denoted AVG'), used in [21], and the $3 \times 3$ filter $\mathcal{F}_{\text{KB}}$ from Eq. (4) (denoted KB), used in [31]. Pixel prediction $\mathcal{F}$ is done by convolution,

$$\hat{\boldsymbol{x}} = \boldsymbol{x} * \mathcal{F}. \tag{9}$$

*3.3.2 U-Net pixel predictor.* We propose a pixel predictor U-Net$^{(k)}$ based on the U-Net architecture, described in Sec. 2.5, and modified by taking only $k$ subsampling steps. Table 6 shows the number of parameters, receptive field, FLOP count, training time, and training batch sizes for different choices of $k$. We carry out a comparison of the networks in Sec. 4.4, including a grid search over $k$, for which the best choice turns out to be $k = 2$.

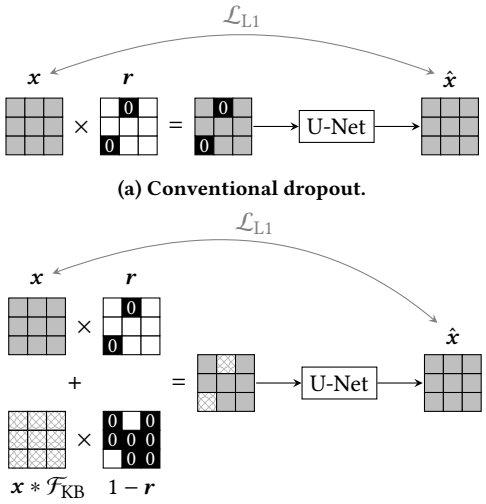
*KB dropout.* U-Net is trained using a dropout. The objective is to fill the "holes" in the input image with values close to the original value. We introduce a KB dropout, shown in Fig. 4, as a modification of the dropout in [45]. KB dropout is formally defined as

$$\boldsymbol{x}' = \boldsymbol{x}^T \boldsymbol{r} + (\boldsymbol{x} * \mathcal{F}^{(3\times3)}_{\text{KB}})^T (1 - \boldsymbol{r}), \tag{10}$$

where $\boldsymbol{r} \overset{\text{i.i.d.}}{\sim} \text{Bernoulli}(p)$ is the dropout mask and $\mathcal{F}^{(3\times3)}_{\text{KB}}$ is the KB filter from Eq. (4). Parameter $p$ controls the density of the dropout.

Unlike the conventional dropout [45], KB dropout does not zero the activations but replaces them with their KB prediction. KB dropout does not degrade the prediction of the neighboring pixels as severely as the conventional dropout. Replacing the pixel with its KB prediction does not change the signal amplitude, so the remaining activations do not have to be scaled up.

*3.3.3 B0 benchmark.* Our deep-learning steganalysis benchmark is EfficientNet B0 [46] in three variants: vanilla [46], with removed stride in the stem layer (ns-B0) [51], and with removed stride in the stem layer and the LSBR reference channel, i.e., a second channel with the input image whose LSBs are set to 0, (ns/r-B0) [8]. Note that this benchmark is a detector, not a quantitative change rate estimator.




**(a) Conventional dropout.**

**(b) Proposed KB dropout.**

**Figure 4: Schema of training a predictor using a dropout on the input layer. U-Net predicts the entire frame but does not know which pixels are dropped out.**

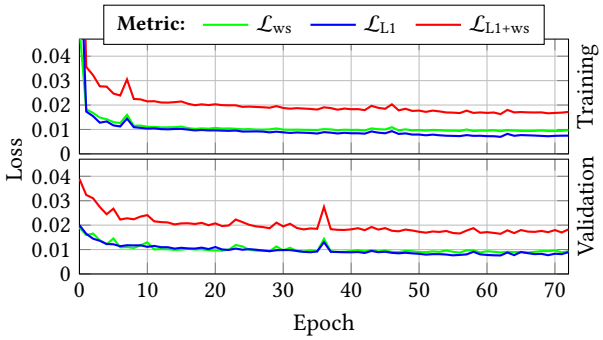



**Figure 5: The progress of training and validation loss $\mathcal{L}_{\text{L1+ws}} = \mathcal{L}_{\text{L1}} + \mathcal{L}_{\text{ws}}$. Both loss components decrease simultaneously. No overfitting is observed.**

## 3.4 Training

Both U-Net and B0 were initialized from scratch, with uniform initialization. Training was done using the optimizer AdamW, and learning rate $10^{-4}$, without augmentations. The specifications for training of each model are given below.

*3.4.1 U-Net.* U-Net$^{(2)}$ is trained in two modes: with covers only and on pairs of cover–stego. The modes use different loss functions described below.

*Cover-only.* The cover-only training is done on image samples without steganography. To prevent the network from copying the

input directly to the output, the input KB dropout, described in Sec. 3.3, is applied before the first layer at $p = 0.1$.

The training loss for the cover-only training, $\mathcal{L}_{\text{L1}}$, is the MAE between the predicted cover $\hat{\boldsymbol{x}}^{(0)}$ and the true cover $\boldsymbol{x}^{(0)}$,

$$\mathcal{L}_{\text{L1}}(\hat{\boldsymbol{x}}^{(0)}; \boldsymbol{x}^{(0)}) = \text{MAE}(\hat{\boldsymbol{x}}^{(0)}, \boldsymbol{x}^{(0)}). \tag{11}$$

*Cover–stego.* The second training mode builds on the idea to directly minimize the MAE between the estimated change rate $\hat{\beta}$ using WS (Eq. (3)) and the true change rate $\beta$,

$$\mathcal{L}_{\text{ws}}(\boldsymbol{x}^{(\alpha)}, \hat{\boldsymbol{x}}^{(0)}; \beta) = \text{MAE}(\hat{\beta}, \beta). \tag{12}$$

For this to work, both covers and stego images are used in training. However, training with the loss $\mathcal{L}_{\text{ws}}$ from Eq. 12 does not converge. Thus, we combine $\mathcal{L}_{\text{ws}}$ with $\mathcal{L}_{\text{L1}}$ from Eq. 11 into a final compound loss,

$$\mathcal{L}_{\text{L1+ws}}(\boldsymbol{x}^{(\alpha)}, \hat{\boldsymbol{x}}^{(0)}; \boldsymbol{x}^{(0)}, \beta) = \mathcal{L}_{\text{L1}}(\cdot) + \mathcal{L}_{\text{ws}}(\cdot). \tag{13}$$

Note that $\mathcal{L}_{\text{ws}}$ is computed from the input image, the output image, and the true embedding rate as a label. Usual losses, including $\mathcal{L}_{\text{L1}}$, are computed from the output and the label only. Both $\mathcal{L}_{\text{ws}}$ and $\mathcal{L}_{\text{L1+ws}}$ require cover and stego as inputs, thus they can only be used in cover–stego training mode.

We train two models, for LSBR and HILLR steganography, at a constant embedding rate $\alpha = 0.4$. We ensure a balanced representation of covers and stegos in the batch but do not constrain the corresponding cover–stego pair to occur together. Unlike in the cover-only mode, no input dropout is applied. Figure 5 shows the progress of training and validation loss for these functions.

*3.4.2 B0.* All three B0 variants are binary classifiers of cover and stego (LSBR). The training was done using the cross-entropy loss, with curriculum learning on the relative payload, $\alpha = 0.4 \to 0.2 \to 0.1 \to 0.05 \to 0.01$.

## 3.5 Performance metrics

*3.5.1 Prediction.* We report the predictability of cover pixels either as the absolute error, $|\hat{\boldsymbol{x}}^{(0)} - \boldsymbol{x}^{(\alpha)}|$, or the mean absolute error (MAE) from Eq. (11). In addition, we calculate the MAE on the subset of hard-to-model pixels, which we identify by taking the $\lambda$-quantile of the HILL cost function,

$$\text{MAE}^{(\lambda)}_{\text{sub}}(\hat{\boldsymbol{x}}^{(0)}, \boldsymbol{x}^{(\alpha)}) = \frac{1}{\lambda N} \sum_{i=1}^{N} \mathbb{1}_{\rho_i < q_\lambda(\boldsymbol{\rho})} |x_i^{(\alpha)} - \hat{x}_i^{(0)}|. \tag{14}$$

Here, $\mathbb{1}_{\text{condition}}$ is the indicator function and $q_\lambda(\boldsymbol{\rho})$ is the $\lambda$-quantile of the cost vector $\boldsymbol{\rho}$. In the experiments, we set $\lambda = 0.1$, i.e., taking 10% of the pixels.

*3.5.2 Estimation.* As the performance measure of the change rate estimation, we use the MAE from Eq. (12). To show the violation of the WS assumption, we measure the correlation $\rho$ from Eq. (8).

*3.5.3 Detection.* The natural way to turn an estimator into a detector is by thresholding the change rate estimate. We report the detection performance via Receiver Operating Characteristic (ROC) curves, which compare the false positive rate, $\text{FPR} = \frac{\text{FP}}{\text{N}}$, to the true positive rate, $\text{TPR} = \frac{\text{TP}}{\text{P}}$, at varying thresholds $\tau$. We focus on low FP rates, as required in practice. For comparison we also project the ROC curve to the scalar metric $\text{P}_\text{E} = \arg\min_\tau \frac{\text{FPR}(\tau)+1-\text{TPR}(\tau)}{2}$, which relates to the point on the ROC curve where the sum of both error rates is minimal [6, 20].

**Table 2: Qualitative summary of results comparing the proposed method against state of the art.**

| Test case | Sec. | U-Net$^{(2)}$ | |
|---|---|---|---|
| | | $\mathcal{L}_{\text{L1}}$ | $\mathcal{L}_{\text{L1+ws}}$ |
| Pixel prediction | 4.1 | improves a lot | improves |
| WS estimation | 4.2 | fails | improves |
| Binary detection | 4.3 | — | improves |
| Choice of $k$ | 4.4 | $k = 2$ | |
| Correlated prediction | 4.5 | yes | no |
| Uses center pixel | 4.6 | yes | adaptively |
| Robust to cover source | 4.7 | — | yes |
| Robust to embedding fct. | | — | no |


**Figure 6: Spatial distribution of absolute prediction errors $|\boldsymbol{x}^{(\alpha)} - \hat{\boldsymbol{x}}^{(0)}|$ on a sample image `6.pgm` from BOSSBase. Deep learning-based prediction excels at edges.**

**Table 3: Errors of the pixel predictors on BOSSBase over the entire image (MAE) and over textured areas only ($\text{MAE}^{(0.1)}_{\text{sub}}$). Lower values are better.**

| | AVG' | KB | U-Net$^{(2)}$ | |
|---|---|---|---|---|
| | | | $\mathcal{L}_{\text{L1+ws}}$ | $\mathcal{L}_{\text{L1}}$ |
| MAE | 4.018 | 2.681 | 2.147 | 0.652 |
| $\text{MAE}^{(0.1)}_{\text{sub}}$ | 9.593 | 7.083 | 4.815 | 1.168 |

# 4 RESULTS

Table 2 shows the outline of this section. A final Sec. 4.8 compares the proposed loss functions.

## 4.1 Pixel prediction

In Fig. 6, we use a sample image `6.pgm` from BOSSBase to illustrate the errors of the pixel predictors, KB and U-Net$^{(2)}$/$\mathcal{L}_{\text{L1+ws}}$. The errors of both predictors are mainly located in the complex regions of the image. Observe that the U-Net predictions improve over the KB filter primarily at the edges.

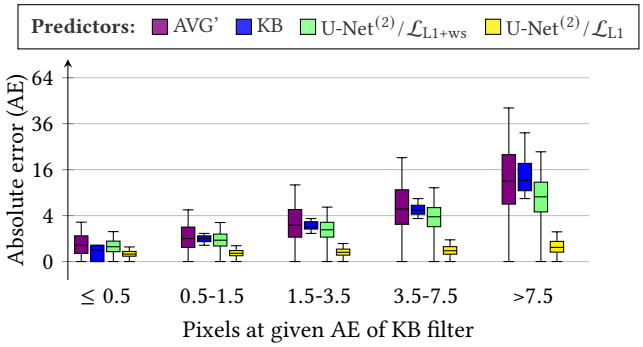



**Figure 7: Error distribution of pixel predictors, split into bins of KB filter errors. Lower values are better. U-Net improves in the areas where the linear filters have the largest error. Note the non-linear $y$ axis.**

**Table 4: MAEs of WS estimates for different pixel predictors. Lower values are better. U-Net$^{(2)}$/$\mathcal{L}_{\text{L1+ws}}$ improves over the linear filters. U-Net$^{(2)}$/$\mathcal{L}_{\text{L1}}$ fails.**

| $\alpha$ | AVG' | KB | U-Net$^{(2)}$ | |
|---|---|---|---|---|
| | | | $\mathcal{L}_{\text{L1+ws}}$ | $\mathcal{L}_{\text{L1}}$ |
| Cover | 0.026 | 0.007 | 0.004 | 0.009 |
| 0.01 | 0.027 | 0.009 | 0.006 | 0.009 |
| 0.05 | 0.029 | 0.011 | 0.008 | 0.021 |
| 0.1 | 0.028 | 0.012 | 0.008 | 0.042 |
| 0.2 | 0.028 | 0.009 | 0.008 | 0.088 |
| 0.4 | 0.025 | 0.010 | 0.009 | 0.183 |

Next, we aggregate the prediction error over 2 000 images. According to Tab. 3, the best predictor is the U-Net with a loss $\mathcal{L}_{\text{L1}}$. Both U-Net-based predictors outperform the linear filters, especially in the $\text{MAE}^{(0.1)}_{\text{sub}}$ metric. This means that the U-Net predictors improve mainly in low-cost, i.e., hard-to-predict areas.

To confirm where the improvement comes from, we split the pixels into bins based on the prediction error of the KB filter. This allows us to make a more conclusive comparison of a selected pixel predictor with the KB filter. From Fig. 7 we see that U-Net$^{(2)}$/$\mathcal{L}_{\text{L1}}$ improves in all the bins. U-Net$^{(2)}$/$\mathcal{L}_{\text{L1+ws}}$ outperforms the linear filters in complex areas but gets slightly worse in smooth areas.

## 4.2 Change rate estimation

We use the models as pixel predictors for WS and compute the change rate estimates $\hat{\beta}$. Fig. 8 shows their distribution for two embedding functions and different embedding rates $\alpha$. U-Net$^{(2)}$/$\mathcal{L}_{\text{L1+ws}}$ yields accurate estimates with relatively narrow error bars, confidently outperforming the linear filter predictors. U-Net$^{(2)}$/$\mathcal{L}_{\text{L1}}$, the best pixel predictor from Sec. 4.1, fails as a WS pixel predictor. All its change rate estimates $\hat{\beta}$ are close to 0. The reason behind the failure of this model is explained in Sec. 4.5. We exclude this model from further experiments.

**Table 5: Detection performance of WS with different predictors and variants EfficientNet-B0 on LSBR, reported as $P_E$. Lower values are better. U-Net$^{(2)}$/$\mathcal{L}_{\text{L1+ws}}$ improves over the linear filters, outperforms B0s for $\alpha = 0.05$ and $\alpha = 0.1$.**

| $\alpha$ | WS | | | B0 | | |
|---|---|---|---|---|---|---|
| | AVG' | KB | U-Net$^{(2)}$ | B0 | ns-B0 | ns/r-B0 |
| 0.01 | 0.429 | 0.381 | 0.358 | 0.371 | 0.220 | 0.266 |
| 0.05 | 0.236 | 0.130 | 0.094 | 0.215 | 0.134 | 0.132 |
| 0.1 | 0.184 | 0.047 | 0.023 | 0.184 | 0.114 | 0.102 |
| 0.2 | 0.065 | 0.011 | 0.003 | 0.169 | 0.114 | 0.082 |

**Table 6: Number of parameters, receptive field, training batch size, floating point operations (FLOP) during forward pass, and training time for different choices of $k$.**

| | # param. [$10^3$] | Rec. field | Batch size | FLOPs [$10^9$] | Training time |
|---|---|---|---|---|---|
| U-Net$^{(0)}$ | 37 | 5 | 32 | 19.66 | 2h33 |
| U-Net$^{(1)}$ | 403 | 12 | 32 | 123.82 | 4h23 |
| U-Net$^{(2)}$ | 1862 | 26 | 16 | 227.98 | 4h30 |
| U-Net$^{(3)}$ | 7696 | 54 | 16 | 332.13 | 7h11 |
| U-Net$^{(4)}$ | 31030 | 110 | 8 | 436.28 | 7h17 |
| B0 | 3967 | 512 | 32 | 3.9 | 0h34 |
| ns-B0 | 3967 | 512 | 16 | 15.46 | 1h11 |
| ns/r-B0 | 3967 | 512 | 16 | 15.76 | 2h16 |

## 4.3 Steganography detection

We convert the WS estimator into a detector by setting a threshold $\tau$ for the change rate estimate $\hat{\beta}$. The benchmark is EfficientNet B0 in three variants, described in Sec. 3.3. The B0-based detectors are trained for a single operational point, $\tau \approx 0.5$. We derive the quasi-ROC curves from their output softmax scores.

Figure 9 compares the detection performance via ROC curves for the embedding rates 0.01, 0.05, and 0.1. Detection of higher embedding rates is near-perfect. WS + U-Net$^{(2)}$ confidently outperforms the other WS-based detectors: at $\alpha = 0.05$ and TPR 0.8, WS+U-Net$^{(2)}$ achieves half the false positives compared to WS+KB. WS + U-Net$^{(2)}$ is also better than the B0 variants for $\alpha \in \{0.05, 0.1\}$, yet B0 performs better for a low embedding rate $\alpha = 0.01$.

Table 5 presents these results using the $P_E$ metric. As before, U-Net$^{(2)}$ is the best pixel predictor for WS. For $\alpha = 0.01$, specialized B0s outperform the WS detectors. Using the reference channel in B0 does not pay off for short payloads, and the improvements are smaller than the ones reported for the SRNet architecture [8].

## 4.4 Grid search for $k$

Up until now, the comparison was done with the predictor U-Net$^{(2)}$, i.e., for $k = 2$. This section provides the evidence for this choice. We repeat the change rate estimation (Sec. 4.2) of all $k \in \{0, 1, \ldots, 4\}$ and keep track of predictor performance as well as training and prediction effort.

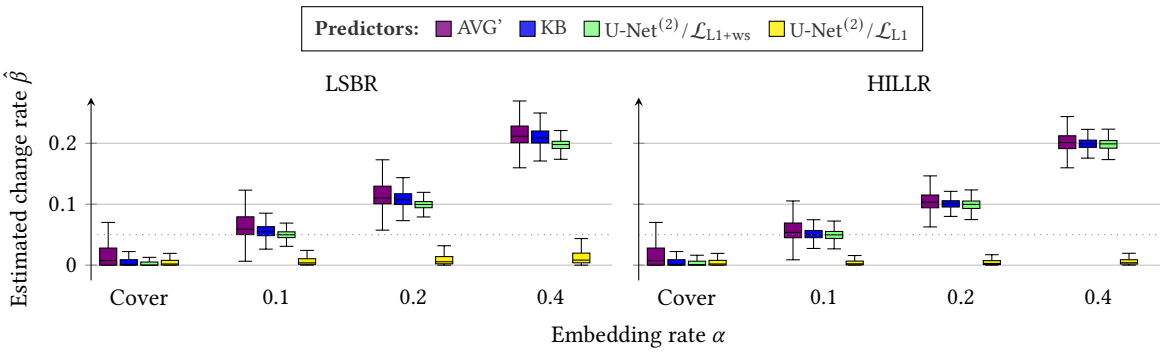



**Figure 8: Performance of different pixel predictors in WS. U-Net$^{(2)}$/$\mathcal{L}_{L1+ws}$ improves over the linear filters. U-Net$^{(2)}$/$\mathcal{L}_{L1}$ fails.**

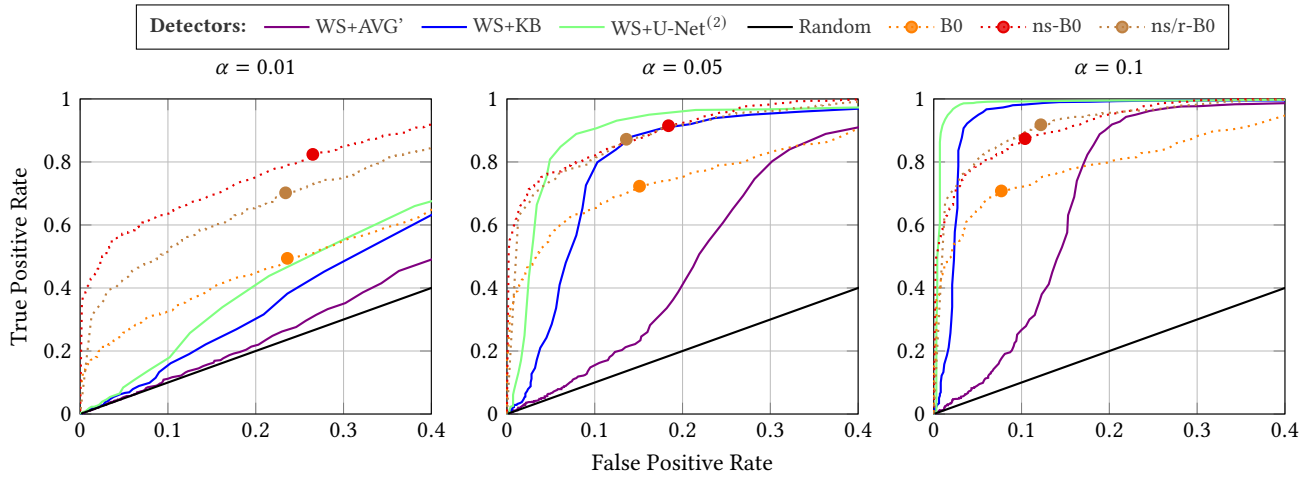



**Figure 9: Performance of LSBR detectors for different relative payloads $\alpha$ as ROC curves. U-Net$^{(2)}$, trained with the loss $\mathcal{L}_{L1+ws}$, improves over the linear filters and outperforms the B0 benchmarks for $\alpha = 0.05$ and $\alpha = 0.1$.**

Figure 10 shows that all models are comparable on both LSBR and HILLR. The models for $k \in \{2, 3, 4\}$ perform nearly identically. Table 6 shows the properties of the models. The network size grows approximately exponentially in $k$, whereas the receptive field grows geometrically. For models $k \in \{2, 3, 4\}$, we had to reduce the training batch sizes to 16 and 8 due to memory limitations. The time is specified for training U-Net$^{(k)}$/$\mathcal{L}_{L1+ws}$ on LSBR at $\alpha = 0.4$ on an NVIDIA A40 GPU with 8 CPU cores, with the patience set to 5 epochs and the batch size unified to 8. For reference, we also report the training time and parameters of the B0 variants. Observe that the training time for the proposed U-Net predictor is non-linear in $k$ and exhibits a step jump from $k = 2$ to 3.

Taking into consideration the performance and the model parameters, we choose U-Net$^{(2)}$.

## 4.5 Why does the best predictor fail in WS?

U-Net$^{(2)}$ with loss $\mathcal{L}_{L1}$ was shown to be the best predictor, yet it completely failed in the change rate estimation in Sec. 4.2. Recall from Sec. 2.4, that the cover prediction error $\hat{\boldsymbol{x}}^{(0)} - \boldsymbol{x}^{(0)}$ must be uncorrelated with the stego noise $\boldsymbol{\delta}$. We test which models break

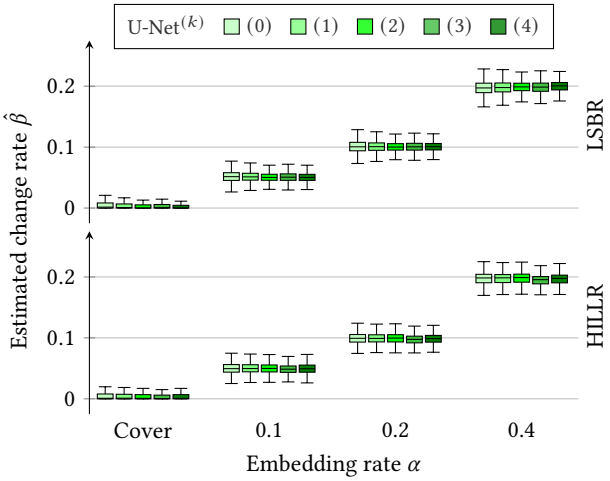



**Figure 10: Grid search for depth parameter $k$ of our model U-Net$^{(k)}$/$\mathcal{L}_{L1+ws}$. The performance is similar for different choices of $k$.**

**Table 7: Median estimate of correlation from Eq. (8) and median p-value for different pixel predictors on LSBR at $\alpha = 1$. WS assumes the prediction error to be uncorrelated with the stego noise. Predictors 1, AVG, and U-Net$^{(2)}/\mathcal{L}_{\mathrm{L1}}$ produce correlated predictions.**

| | AVG' | KB | U-Net$^{(2)}$ | | 1 | AVG |
|---|---|---|---|---|---|---|
| | | | $\mathcal{L}_{\mathrm{L1+ws}}$ | $\mathcal{L}_{\mathrm{L1}}$ | | |
| $\hat{\rho}$ | −0.001 | −0.001 | −0.000 | 0.015 | 0.015 | 0.001 |
| p-value | 0.062 | 0.058 | 0.052 | <u>0.000</u> | <u>0.000</u> | <u>0.030</u> |

this assumption, i.e., $|\rho| = |\mathrm{corr}(\boldsymbol{\delta}, \hat{\boldsymbol{x}}^{(0)} - \boldsymbol{x}^{(0)})| > 0$, using Pearson's correlation test,

$$\hat{\rho}\sqrt{\frac{N-2}{1-\hat{\rho}^2}} \sim t(N-2), \tag{15}$$

where $\rho$ and $\hat{\rho}$ are the true and estimated correlation, and $t(N-2)$ is Student's $t$-distribution with $N-2$ degrees of freedom. We add two filters with non-zero central pixel weight, which should produce correlated prediction, and are thus inappropriate for WS: $1 \times 1$ unit filter, denoted 1, and $3 \times 3$ AVG from (5).

The results are shown in Tab. 7. On the significance level 0.05, AVG', KB, and U-Net$^{(2)}/\mathcal{L}_{\mathrm{L1+ws}}$ have non-significant correlation. Both filters with non-zero central weights and U-Net$^{(2)}/\mathcal{L}_{\mathrm{L1}}$ have a significant correlation, and thus produce predictions correlated with the stego noise. The model U-Net$^{(2)}/\mathcal{L}_{\mathrm{L1}}$ fails as a WS predictor presumably because it breaks one of its assumptions.

## 4.6 Receptivity of the predictor

We illustrate the U-Net receptivity to the pixel neighborhood via signed saliency maps, i.e., the gradients of the input pixels w.r.t. a single output pixel $\hat{x}_i^{(0)}$,

$$g_i(d) = \frac{\partial \hat{x}_i^{(0)}}{\partial x_{i+d}^{(\alpha)}}, \tag{16}$$

where $x_{i+d}^{(\alpha)}$ is the input pixel with offset $d$ to the output pixel, i.e., $d = 0$ for the center pixel. For Fig. 11, we expand the one-dimensional notation to two dimensions as $d \to (d_x, d_y)$. Recall that saliency maps may resemble the linear filters, however, their interpretation is different. They visualize the direction and magnitude of change to the output pixel caused by changing each input pixel by one unit.

We choose as center pixels four selected positions in the sample image `6.png` from BOSSBase – the pixels with the largest and the smallest gradient, and the strongest horizontal as well as vertical edge, as shown in Fig. 11. Figures 11b and 11c show the saliency maps for each of the four center pixels, aligned according to Fig. 11a.

Interpreting Fig. 11b, U-Net$^{(2)}/\mathcal{L}_{\mathrm{L1}}$ predicts the output mainly from the center pixel regardless of the content. The KB dropout does not prevent copying of the center pixel through. By contrast, U-Net$^{(2)}/\mathcal{L}_{\mathrm{L1+ws}}$ combines the neighborhood surrounding the center pixel, as shown in Fig. 11c. The neighborhood for the prediction is wider than $3\times3$ used by the linear filters, but the weights fade out with distance. Note that the pixel prediction is content-adaptive. In the smooth areas, the center pixel is omitted and the neighborhood is weighted in a checkerboard pattern, similar to the KB filter. In the textured areas, the center pixel contributes to the prediction.

To ensure that these saliency maps are representative for cover as well as stego images, and rule out that the U-Net has learned to detect steganography, we repeat the same analysis on stego images with LSBR at different embedding rates. We could not observe any noteworthy differences.

**Table 8: Effect of dataset mismatch. MAE of predictors trained on BOSSBase, evaluated on the ALASKA2 dataset. U-Net$^{(2)}/\mathcal{L}_{\mathrm{L1+ws}}$ still outperforms the linear filters but with a smaller margin.**

| Predictors | AVG' | KB | U-Net$^{(2)}$ |
|---|---|---|---|
| | | | $\mathcal{L}_{\mathrm{L1+ws}}$ |
| MAE | 3.976 | 2.776 | <u>2.696</u> |
| $\mathrm{MAE}_{\mathrm{sub}}^{(0.1)}$ | 5.623 | 3.992 | <u>3.641</u> |

## 4.7 Generalization of the predictor

In this section, we evaluate the capability of the predictor to generalize to two mismatched conditions: mismatching dataset, mismatching embedding strategy, and their combination.

*4.7.1 Mismatched dataset.* We choose ALASKA2 in order to evaluate the predictor's performance on a mismatching dataset. We run these three steps: prediction, change rate estimation, and detection (but keep training on BOSSBase).

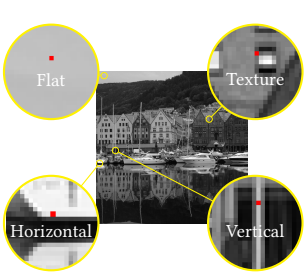
The predictability results, shown in Tab. 8, are to be directly compared to the matched case in Tab. 3. U-Net with $\mathcal{L}_{\mathrm{L1+ws}}$ still performs better than the linear filters, although the improvement margin is smaller. Moreover, the improvement margin between AVG' and KB filters also gets smaller. This is perhaps due to the harsh downsampling of BOSSBase, which makes this dataset intrinsically more difficult to predict than ALASKA2.

The robustness to the dataset is also explored on change rate estimation. The results on ALASKA2 in Fig. 12 are to be compared to the matched case in Fig. 8. The WS estimates with the U-Net$^{(2)}$ predictor get slightly worse, unlike WS with the KB filter.

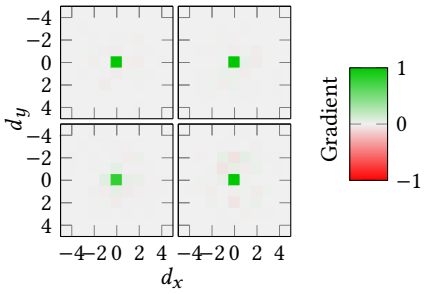
We convert the change rate estimator to a detector and show ROC curves in Fig. 13, which is comparable to the matched case in Fig. 9. Observe that the margin between U-Net$^{(2)}$ and KB becomes very thin. Comparing Fig. 13 to Fig. 9 reveals that analytical detectors, including our proposed one with a learning-based predictor, react in very different ways to dataset mismatch than a purely learned detector. While the performance of all three WS variants is better than on BOSSBase, EfficientNet B0 gets significantly worse due to the dataset mismatch.

*4.7.2 Mismatched embedding function.* We evaluate the effect of mismatch in the embedding function using the predictors trained on LSBR to estimate the change rate of HILLR steganography on BOSSBase.

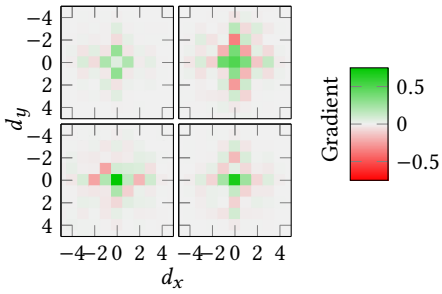
Figure 14 shows the results and should be compared to the matched case in Fig. 8. Observe that the WS estimates of U-Net$^{(2)}$




(a) Locations in a sample image `6.png` from BOSSBase.




(b) U-Net$^{(2)}$/$\mathcal{L}_{\text{L1}}$ uses mostly the center pixel.




(c) U-Net$^{(2)}$/$\mathcal{L}_{\text{L1+ws}}$ uses a larger neighborhood, but also the center pixel in three out of four locations.

**Figure 11: Signed saliency maps with gradients $g_i(d)$ of selected pixels to illustrate the receptivity of the predictors in different local neighborhoods. The grids are aligned according to Fig. 11a. Predictors are adaptive w.r.t. the complexity and directionality of the texture.**

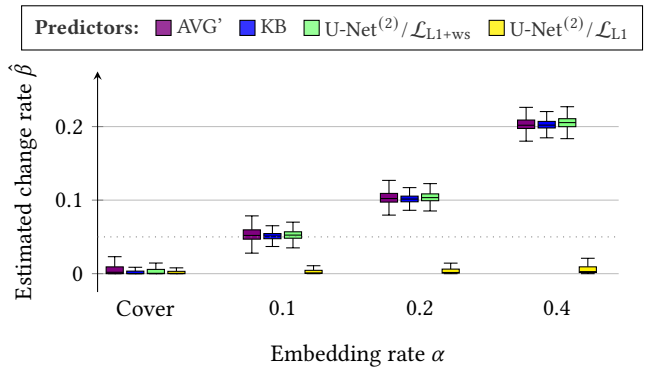



**Figure 12: Effect of dataset mismatch. Performance of different WS predictors, trained on BOSSBase, evaluated on ALASKA2. The performance of U-Net$^{(2)}$/$\mathcal{L}_{\text{L1+ws}}$ and linear filters is comparable.**

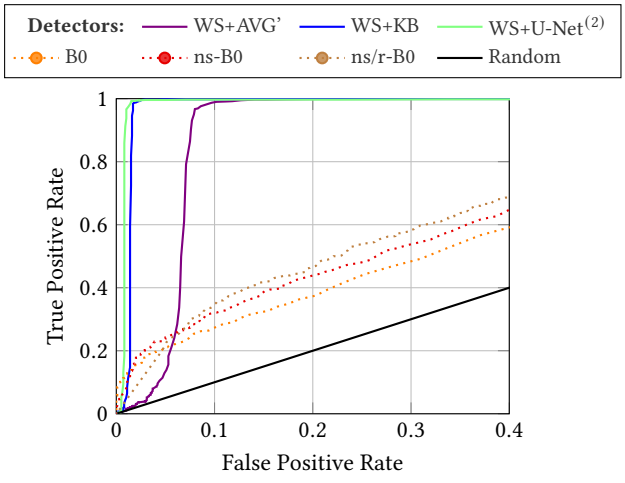



**Figure 13: Effect of dataset mismatch. ROC curves of LSBR detectors trained on BOSSBase and evaluated on ALASKA2 at embedding rate $\alpha = 0.1$.**

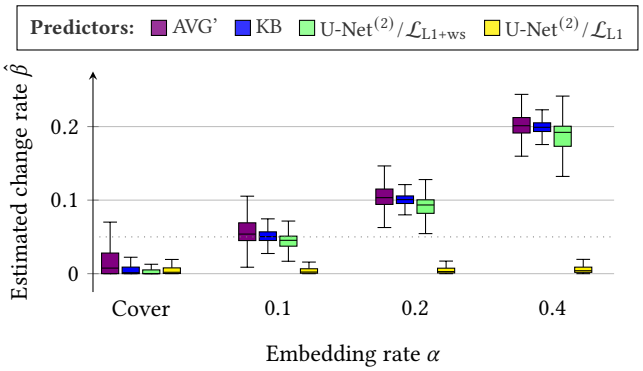



**Figure 14: Effect of mismatching embedding functions. Performance of different predictors of WS, trained on LSBR, evaluated on HILLR. The predicted change rates of U-Net$^{(2)}$/$\mathcal{L}_{\text{L1+ws}}$ are off.**

with $\mathcal{L}_{\text{L1+ws}}$ are inconsistent and perform inferior to the linear predictors. However, they may still serve as a usable detector (unlike U-Net$^{(2)}$/$\mathcal{L}_{\text{L1}}$), because they separate stego images and covers.

*4.7.3 Mismatched dataset and embedding function.* As a final experiment, we combine both mismatch conditions and evaluate the predictors on the ALASKA2 dataset with HILLR steganography. Again, we measure the change rate estimation only. Figure 15 shows that the effect is very similar to the case of mismatching embedding function only.

## 4.8 Comparison of the loss functions

We close the results section by recalling why the choice of the loss function is critical. The rightmost columns of Tab. 2 (above) summarize the presented results in qualitative statements for each loss function.

Although $\mathcal{L}_{\text{L1}}$ leads to the best model in the pixel prediction, it fails in the WS estimation. As shown in Sec. 4.5, predictors trained with this loss function violate the WS assumption in Eq. 8 and their predictions include the center pixel. By contrast, our proposed

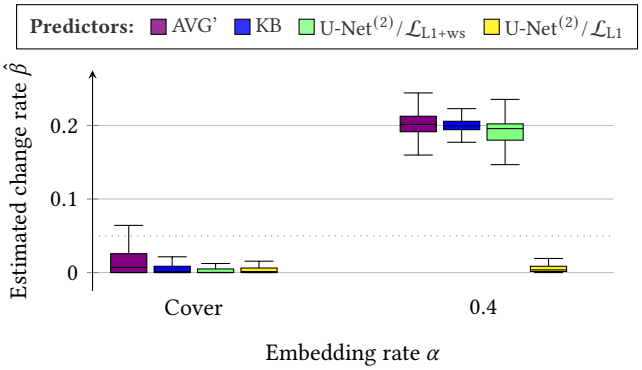



**Figure 15: Effect of combined mismatch in dataset and embedding function. Training on BOSSBase with LSBR, evaluation on ALASKA2 with HILLR.**

U-Net$^{(2)}$ using $\mathcal{L}_{\text{L1+ws}}$ improves in all three test cases compared to the linear filters. The predictors trained with this loss function seem to satisfy the assumption although they make use of the center pixel in certain conditions (see Sec. 4.6).

## 5 DISCUSSION

Our proposed U-Net-based pixel predictor improves over conventional linear predictors in WS steganalysis. By thresholding the WS estimate of the change rate, we construct a steganalysis detector, which outperforms the state-of-the-art detectors for LSBR in certain cases. Our construction as a change rate estimator lets the method generalize to unseen payload lengths quite naturally.

As diffusion models are primarily used to denoise an image, using them for pixel prediction in steganalysis is not straightforward. The most critical aspect when adapting U-Net to the steganalysis task is the choice of the loss function, which must reflect the requirements of the downstream estimator. WS assumes that the prediction error does not correlate with the stego noise. The loss function we propose includes a term that penalized undue correlation. As a result, the receptivity of the predictor spans a broader neighborhood and tends to deemphasize the center pixel.

While we cannot claim generality, our results also provide first indications that incorporating deep learning as a component into analytical steganalysis methods may generalize better to unseen conditions than detectors based solely on deep learning. This may open an unexplored avenue toward mitigation of cover-source mismatch [38].

*Limitations.* Replacing linear filters with a U-Net predictor introduces additional overhead. The number of parameters increases from none to millions. The processing time per image on our test CPU increases from 6 to 660 milliseconds. A previously absent learning phase, connected with computational resources and the collection of training samples, is required to estimate U-Net weights. A common limitation of all deep learning-based steganalysis is higher uncertainty regarding generalization to unseen data and embedding methods. While this effect was not drastic in the scenarios studied here, we cannot rule out that very different cover sources or adaptivity criteria exist that lead to failure. A general limitation of WS is that it is tailored to detect LSBR steganography, which has structural weaknesses that help the steganalyst compared to state-of-the-art spatial-domain steganography.

*Future work.* Our modified loss function is critical for the successful use of U-Net as a predictor in WS steganalysis. In its current form, undue correlation is penalized indirectly via the error of the WS estimate, which requires training in cover–stego mode. A next step would be to avoid this complication and enable cover-only training, for example by adding a penalty term that is directly tied to the amount of undesired correlation. Another possible way to adapt U-Net without touching the loss function would be to prune the connection structure of the network such that the center pixel cannot be used for the prediction.

In this work, we have experimented with *un*weighted WS because adaptive steganography was part of the experiments. Weights in WS remain an open question. This is also related to the known sensitivity of WS to previous JPEG compression [4]. Further experiments could be done regarding the generalization capabilities, for example to quantify the effect of the input size, as recently studied by [7] for CNN detectors.

The U-Net architecture was chosen for its popularity and modularity. Future work should also consider other possible network architectures for pixel prediction, such as DnCNN [53]. More generally, deep-learning based pixel predictors have not fully been appreciated in information hiding. Their prediction error could be used to construct new distortion metrics and detectors.

## 6 RELATED WORK

*WS steganalysis.* WS steganalysis was introduced in 2004 [21] and revisited in 2008 with an improved pixel predictor, modified weighting, and introducing bias correction [5, 31]. WS was also extended for LSBR in the JPEG domain [4], for adaptive LSBR [42], and payload localization in batch steganalysis [29]. [32] showed that WS can be a near-perfect detector when linear demosaicking is used. [13, 14, 54] explore the relationship between WS and likelihood-ratio test. Later works on LSB matching detection [12, 15] use WS filtering, estimating the pixel prediction and local variance in the same way as WS.

*U-Net-based pixel prediction.* Pixel predictor is an umbrella term for models estimating pixel values, with the pixels known either approximately, or not at all [31, 53]. This encompasses denoising and inpainting, very different tasks, which share similarities in the single-pixel scenario. U-Net, a fully convolutional neural network architecture originally proposed for biomedical image segmentation [41], was successfully applied to many different image processing tasks, including denoising [24, 35], image inpainting [27, 37, 49], or synthetic image generation [3, 25, 40]. In steganography, U-Net has been previously used for probability map generation [50] and in a complete scheme involving U-Net-based embedding and extraction functions [17]. This paper brings a new deep-learning architecture to steganalysis, similar to [6, 51]. Unlike them, it does not propose a steganalysis detector, but a pixel prediction component to be used in an existing framework. We choose U-Net for its universality and popularity and keep other similar networks, such as DnCNN [53], for future work.

## 7 CONCLUSION

This work explores how to combine the best of both worlds: an analytical, well-understood quantitative steganalysis method, improved by replacing one of its components with deep learning. Our results give some hope to the research of analytical methods. Clean ideas can outperform the blind application of deep learning. We should use neural networks in a smart way, combined with prior domain knowledge, if we want to overcome the challenges, such as the cover-source mismatch [38], or moving from the laboratory to the real world [30].

The source code, trained models, and selected data underlying the figures and tables are available at https://github.com/uibk-uncover/ws-unet.

## ACKNOWLEDGMENTS

We thank Benedikt Lorch, Nora Hofer, and Verena Lachner for their valuable support and feedback, and other lab members for their comments. This work is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 101021687 (UNCOVER). The computational results have been calculated on the Vienna Scientific Cluster (VSC).

## REFERENCES

[1] Patrick Bas, Tomáš Filler, and Tomáš Pevný. 2011. "Break Our Steganographic System": The Ins and Outs of Organizing BOSS. In *Information Hiding (IH)*. Springer, 59–70.
[2] Martin Beneš, Benedikt Lorch, and Rainer Böhme. 2023. JPEG Steganalysis Using Leaked Cover Thumbnails. In *Workshop on Information Forensics and Security (WIFS)*. IEEE, 1–6.
[3] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, Wesam Manassra, Prafulla Dhariwal, Casey Chu, Yunxin Jiao, and Aditya Ramesh. 2023. Improving Image Generation with Better Captions. *Computer Science* 2, 3 (2023), 8.
[4] Rainer Böhme. 2008. Weighted Stego-image Steganalysis for JPEG Covers. In *Information Hiding (IH)*. Springer, 178–194.
[5] Rainer Böhme. 2010. *Advanced Statistical Steganalysis*. Vol. 284. Springer Berlin.
[6] Mehdi Boroumand, Mo Chen, and Jessica Fridrich. 2018. Deep Residual Network for Steganalysis of Digital Images. *IEEE Transactions on Information Forensics and Security (TIFS)* 14, 5 (2018), 1181–1193.
[7] Jan Butora and Patrick Bas. 2024. Size-Independent Reliable CNN for RJCA Steganalysis. *IEEE Transactions on Information Forensics and Security (TIFS)* (2024).
[8] Mo Chen, Mehdi Boroumand, and Jessica Fridrich. 2019. Reference Channels for Steganalysis of Images with Convolutional Neural Networks. In *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. ACM, 188–197.
[9] Dave Coffin. 2008. DCRAW: Decoding Raw Digital Photos in Linux. https://www.dechifro.org/dcraw/ Accessed: Mar 21, 2024.
[10] Rémi Cogranne, Quentin Giboulot, and Patrick Bas. 2019. The ALASKA Steganalysis Challenge: A First Step Towards Steganalysis. In *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. ACM, 125–137.
[11] Rémi Cogranne, Quentin Giboulot, and Patrick Bas. 2020. ALASKA#2: Challenging Academic Research on Steganalysis with Realistic Images. In *Workshop on Information Forensics and Security (WIFS)*. IEEE, 1–5.
[12] Remi Cogranne and Florent Retraint. 2013. An Asymptotically Uniformly Most Powerful Test for LSB Matching Detection. *IEEE Transactions on Information Forensics and Security (TIFS)* 8, 3 (2013), 464–476.
[13] Rémi Cogranne, Cathel Zitzmann, Lionel Fillatre, Igor Nikiforov, Florent Retraint, and Philippe Cornu. 2011. Reliable Detection of Hidden Information based on a Non-Linear Local Model. In *Statistical Signal Processing Workshop (SSP)*. IEEE, 493–496.
[14] Rémi Cogranne, Cathel Zitzmann, Lionel Fillatre, Florent Retraint, Igor Nikiforov, and Philippe Cornu. 2011. A Cover Image Model for Reliable Steganalysis. In *Information Hiding (IH)*. Springer, 178–192.
[15] Rémi Cogranne, Cathel Zitzmann, Florent Retraint, Igor Nikiforov, Lionel Fillatre, and Philippe Cornu. 2012. Statistical Detection of LSB Matching using Hypothesis Testing Theory. In *Information Hiding (IH)*. Springer, 46–62.
[16] Davide Cozzolino and Luisa Verdoliva. 2019. Noiseprint: A CNN-based Camera Model Fingerprint. *IEEE Transactions on Information Forensics and Security (TIFS)* 15 (2019), 144–159.
[17] 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.
[18] Sorina Dumitrescu, Xiaolin Wu, and Zhe Wang. 2003. Detection of LSB Steganography via Sample Pair Analysis. In *Information Hiding (IH)*. Springer, 355–372.
[19] Tomáš Filler, Jan Judas, and Jessica Fridrich. 2011. Minimizing Additive Distortion in Steganography Using Syndrome-Trellis Codes. *IEEE Transactions on Information Forensics and Security (TIFS)* 6, 3 (2011), 920–935.
[20] Jessica Fridrich. 2009. *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press.
[21] Jessica Fridrich and Miroslav Goljan. 2004. On Estimation of Secret Message Length in LSB Steganography in Spatial Domain. In *Security, Steganography, and Watermarking of Multimedia Contents VI (SSWMC)*, Vol. 5306. SPIE, 23–34.
[22] Jessica Fridrich, Miroslav Goljan, and Rui Du. 2001. Reliable Detection of LSB Steganography in Color and Grayscale Images. In *Workshop on Multimedia and Security (MMSec)*. ACM, 27–30.
[23] Jessica Fridrich and Jan Kodovský. 2012. Rich Models for Steganalysis of Digital Images. *IEEE Transactions on Information Forensics and Security (TIFS)* 7, 3 (2012), 868–882.
[24] Javier Gurrola-Ramos, Oscar Dalmau, and Teresa Alarcón. 2021. A Residual Dense U-Net Neural Network for Image Denoising. *IEEE Access* 9 (2021), 31742–31754.
[25] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. *Conference on Neural Information Processing Systems (NeurIPS)* 33 (2020), 6840–6851.
[26] Vojtěch Holub and Jessica Fridrich. 2014. Low-complexity Features for JPEG Steganalysis Using Undecimated DCT. *IEEE Transactions on Information Forensics and Security (TIFS)* 10, 2 (2014), 219–228.
[27] Amreen Kaur, Ankit Raj, N Jayanthi, and S Indu. 2020. Inpainting of Irregular Holes in a Manuscript Using U-Net and Partial Convolution. In *International Conference on Inventive Research in Computing Applications (ICIRCA)*. IEEE, 778–784.
[28] Andrew Ker. 2005. A General Framework for Structural Steganalysis of LSB Replacement. In *Information Hiding (IH)*. Springer, 296–311.
[29] Andrew Ker. 2008. Locating Steganographic Payload via WS Residuals. In *Workshop on Multimedia and Security (MMSec)*. ACM, 27–32.
[30] Andrew Ker, Patrick Bas, Rainer Böhme, Rémi Cogranne, Scott Craver, Tomáš Filler, Jessica Fridrich, and Tomáš Pevný. 2013. Moving Steganography and Steganalysis from the Laboratory into the Real World. In *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. ACM, 45–58.
[31] Andrew Ker and Rainer Böhme. 2008. Revisiting Weighted Stego-Image Steganalysis. In *Security, Forensics, Steganography, and Watermarking of Multimedia Contents X (SFSWMC)*, Vol. 6819. SPIE, 56–72.
[32] Matthias Kirchner and Rainer Böhme. 2014. Steganalysis in Technicolor: Boosting WS Detection of Stego Images from CFA-Interpolated Covers. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 3982–3986.
[33] Matthias Kirchner and Cameron Johnson. 2019. SPN-CNN: Boosting Sensor-based Source Camera Attribution with Deep Learning. In *Workshop on Information Forensics and Security (WIFS)*. IEEE, 1–6.
[34] Jan Kodovský and Jessica Fridrich. 2013. Quantitative Steganalysis using Rich Models. In *Media Watermarking, Security, and Forensics (MWSF)*, Vol. 8665. SPIE, 228–238.
[35] Rina Komatsu and Tad Gonsalves. 2020. Comparing U-Net-Based Models for Denoising Color Images. *AI* 1, 4 (2020), 465–486.
[36] Bin Li, Ming Wang, Jiwu Huang, and Xiaolong Li. 2014. A New Cost Function for Spatial Image Steganography. In *International Conference on Image Processing (ICIP)*. IEEE, 4206–4210.
[37] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. 2018. Image Inpainting for Irregular Holes using Partial Convolutions. In *European Conference on Computer Vision (ECCV)*. Springer, 85–100.
[38] Antoine Mallet, Martin Beneš, and Rémi Cogranne. 2024. Cover-Source Mismatch in Steganalysis: Systematic Review. (2024). https://www.researchsquare.com/article/rs-3812991 Accessed: Mar 21, 2024.
[39] Tomáš Pevný, Tomáš Filler, and Patrick Bas. 2010. Using High-dimensional Image Models to Perform Highly Undetectable Steganography. In *Information Hiding (IH)*. Springer, 161–177.
[40] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 10684–10695.
[41] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Springer, 234–241.
[42] Pascal Schöttle, Stefan Korff, and Rainer Böhme. 2012. Weighted Stego-Image Steganalysis for Naive Content-Adaptive Embedding. In *Workshop on Information Forensics and Security (WIFS)*. IEEE, 193–198.

[43] Nahian Siddique, Sidike Paheding, Colin P Elkin, and Vijay Devabhaktuni. 2021. U-Net and its Variants for Medical Image Segmentation: A Review of Theory and Applications. *IEEE Access* 9 (2021), 82031–82057.

[44] Xiaofeng Song, Fenlin Liu, Chunfang Yang, Xiangyang Luo, and Yi Zhang. 2015. Steganalysis of Adaptive JPEG Steganography Using 2D Gabor Filters. In *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. ACM, 15–23.

[45] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research (JMLR)* 15, 1 (2014), 1929–1958.

[46] Mingxing Tan and Quoc Le. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *International Conference on Machine Learning (ICML)*. PMLR, 6105–6114.

[47] Andreas Westfeld. 2016. Multimedia Security. In *Information Hiding*, Stefan Katzenbeisser and Fabien Petitcolas (Eds.). Artech House, Chapter 2, 21–43.

[48] Andreas Westfeld and Andreas Pfitzmann. 1999. Attacks on Steganographic Systems: Breaking the Steganographic Utilities EzStego, Jsteg, Steganos, and S-Tools and some Lessons Learned. In *Information Hiding (IH)*. Springer, 61–76.

[49] Chaohao Xie, Shaohui Liu, Chao Li, Ming-Ming Cheng, Wangmeng Zuo, Xiao Liu, Shilei Wen, and Errui Ding. 2019. Image Inpainting with Learnable Bidirectional Attention Maps. In *International Conference on Computer Vision (ICCV)*. IEEE, 8858–8867.

[50] Jianhua Yang, Danyang Ruan, Xiangui Kang, and Yun-Qing Shi. 2019. Towards Automatic Embedding Cost Learning for JPEG Steganography. In *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. ACM, 37–46.

[51] Yassine Yousfi, Jan Butora, Jessica Fridrich, and Clément Fuji Tsang. 2021. Improving Efficient-Net for JPEG Steganalysis. In *Workshop on Information Hiding and Multimedia Security (IH&MMSec)*. ACM, 149–157.

[52] Hui Zeng, Morteza Darvish Morshedi Hosseini, Kang Deng, Anjie Peng, and Miroslav Goljan. 2021. A Comparison Study of CNN Denoisers on PRNU Extraction. *arXiv preprint arXiv:2112.02858* (2021).

[53] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. 2017. Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising. *IEEE Transactions on Image Processing (TIP)* 26, 7 (2017), 3142–3155.

[54] Cathel Zitzmann, Rémi Cogranne, Florent Retraint, Igor Nikiforov, Lionel Fillatre, and Philippe Cornu. 2011. Statistical Decision Methods in Hidden Information Detection. In *Information Hiding (IH)*. Springer, 163–177.