A study on the invariance in security whatever the dimension of images for the steganalysis by deep-learning
Kévin Planolles, Marc Chaumont, Frédéric Comby

To cite this version:
Kévin Planolles, Marc Chaumont, Frédéric Comby. A study on the invariance in security whatever the dimension of images for the steganalysis by deep-learning. 2023. lirmm-04001355

HAL Id: lirmm-04001355
https://hal-lirmm.ccsd.cnrs.fr/lirmm-04001355
Preprint submitted on 23 Feb 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
A STUDY ON THE INVARIANCE IN SECURITY WHATEVER THE DIMENSION OF IMAGES FOR THE STEGANALYSIS BY DEEP-LEARNING

Kévin PLANOLLES*, Marc CHAUMONT†, Senior Member, IEEE Frédéric COMBY*

* LIRMM, Univ. Montpellier, CNRS 161 rue Ada, 34392 Montpellier Cedex 05, France
† Univ. Nîmes Place Gabriel Péri, 30000 Nîmes Cedex 01, France

ABSTRACT

In this paper, we study the performance invariance of convolutional neural networks when confronted with variable image sizes in the context of a more “wild steganalysis”. First, we propose two algorithms and definitions for a fine experimental protocol with datasets owning “similar difficulty” and “similar security”. The “smart crop 2” algorithm allows the introduction of the Nearly Nested Image Datasets (NNID) that ensure “a similar difficulty” between various datasets, and a dichotomous research algorithm allows a “similar security”. Second, we show that invariance does not exist in state-of-the-art architectures. We also exhibit a difference in behavior depending on whether we test on images larger or smaller than the training images. Finally, based on the experiments, we propose to use the dilated convolution which leads to an improvement of a state-of-the-art architecture.

Index Terms— Steganalysis, Images of arbitrary size, Invariance in security, NNID dataset, Deep Learning.

1. INTRODUCTION

Steganalysis is the subject of many academic publications but its use in real-world conditions (“in the wild”) is often far from the laboratory protocols [1] [2]. In laboratory conditions, the worst-case attack is used, meaning that: payload size, image development (demosaicing, gamma correction, blur, color balance, compression parameters, use of color, etc.), are known by the steganalyst. This paper, therefore, fits in the spirit of works on a more “wild/realistic” steganalysis. We are specifically focusing on the case where the dimension of the images is not known by the steganalyst. Said differently, we would like to keep detection performances constant whatever the dimension of the considered image.

Since the advent of deep-learning in steganalysis in 2015 [3], a few architectures have the intrinsic capability to accept input images of variable size [4], [5], [6], [7], [8], [9], [10], [11], etc. Nevertheless, it is reasonable to ask ourselves if those architectures are efficient in detection whatever the dimension. More precisely we would like to know if the used architectures and/or the tricks ensure an invariance in security. This notion of invariance in security will be formally defined in the section 2.

The contributions of that paper are: 1) a well thought experimental setting (dataset building, payload size tuning, and learning protocol) leading to the definition of difficulty and to the Nearly-Nested Image Datasets (NNID), 2) the definition of the invariance in security, 3) the reported experimental observations and finally, 4) the proposition of an update of the architectures in order to be closer to invariance.

In Section 2 we briefly discuss the payload size with a recall of the Square Root Law [12], and we present the state-of-the-art architectures able to process variable dimension images. In Section 3 we describe the way the NNID are built; this comes with the definitions of “same” difficulty of the datasets, and “same” security due to the embedding in each of those datasets. We also propose an upgrade to the deep-learning architectures. Then, in Section 4 we give the experimental protocol, and the experimental results, and discussed the invariance notion.

2. TOOLS USED FOR OUR STUDY

2.1. Few words on the square root law

It is well known that the size of a cover object is a major factor in its capacity to hide information, but the relationship between secure steganographic capacity and cover size is still a discussed subject. In Ker et al.’s paper [12], it is shown that payload size should be proportional to the square root of the cover size. From the Square Root Law we can derive, given a positive constant $k \in \mathbb{R}^+$, that the relative payload size, $\alpha$, for an adaptive embedding, ensuring the same security whatever the dimension $w \times h$ of an image, should be (in bit per pixel):

$$\alpha = \frac{k}{wh} \times \sqrt{wh} \times \log(wh). \quad (1)$$
But in practice, the statistical properties of the cover has
to be taken into account, and it appears that the square root
law is not usable as it is. For example, a homogeneous cover
will be much less secure than a cover with a lot of textures.

In our study, we propose another solution to choose the
relative payload ensuring a similar security considering the
dimension of images. It is based on careful datasets building
and the use of a detector i.e. a deep-learning network (see
Section 3).

2.2. State-of-the-art deep learning architectures

There are roughly two families of deep learning architecture
that have the property to accept images of various dimensions,
those only using the average, and those using more than only
one statistical moment.

For the family based on the average, we can mention
the following networks. The Yedroudj-Net [4] architecture
is based on the Global Average Pooling (GAP) technique to
aggregate the values of the feature maps just before feeding
the classification block. The Zhu-Net [5] architecture is in-
spired by Yedroudj-Net. Its main characteristic is the use of
a Pyramidal Global Average pooling to retrieve features at
multiple scales. A few other ”small” architectures such as
GBRAS-Net [6], CC-Net [7], etc, dealing with spatial image
steganalysis have then improved the Yedroudj-Net and the
Zhu-Net, and are also integrating a GAP. We should also
mention a very recent proposition integrating a transformer
[9] with a GAP. Finally, EWNet [8] is an original approach
that uses a coder-decoder, and proposes to compute the aver-
age of the image of scores for the cover (resp. the stego).

For the family based on more than only one statistical
moment, we can mention the following networks. The SID [10]
architecture is based on a modified version of the Ye-Net
network [13]. The main idea of the SID architecture is to extract
four statistical moments (minimum, maximum, average, and
variance) from the last feature maps and use those moments
for classification. The SiaSteg [11] architecture is also based
on the extraction of statistical moments (the same as SID),
and uses a siamese network in order to add a contrastive loss
in conjunction with the classification loss.

In some of those papers, there are experiments on the “ro-
bustness” to the variation in dimension, and for a subset of
those papers, the notion of secure steganographic capacity
is mentioned, but in none of them the results agreed with this
secure steganographic capacity notion. Additionally, none
of them has set a common difficulty (see definition in section 3.1)
between the dimensions, which ensures that the variation be-
tween datasets is only due to the relative payload size. In this
paper, our experimental protocol ensures the same empirical
security (thanks to adapted payloads) between the dimensions
and works with datasets of various dimensions and similar
difficulty.

3. PROPOSITION

3.1. Nearly-Nested Image Datasets (NNID) ensuring the
same difficulty

For mastered experiments, we need to build what we name
Nearly-Nested Image Datasets (NNID) such that each dataset
owns images of the same dimension, and each dataset is
issued from a cropped version of the images belonging to
the dataset with the biggest dimensions. This last dataset
is named mother dataset and the images are named mother
images.

By using NNID we ensure that the development is the
same in all of the datasets. We also impose, as an additional
constraint, that the difficulty of each dataset is the same. By
same difficulty we mean that the distribution of costs is the
same whatever the dataset. This additional constraint implies
a specific way to crop the images, and most importantly, en-
sure that the experimental results obtained between the vari-
ous dimension will be comparable since the source cost dis-
tribution of each dataset is the same. With the NNID we are
able to avoid any impact of the development or the difficulty,
on the experimental results; All the datasets are very similar
except for the dimension.

We use, as a mother dataset, the LSSD dataset [14]. LSSD
is a mix of RAW images from ALASKA#2, BOSS, StegoApp
DB, Wesaturalte, RAISE, and Dresden datasets and uses a
modified development script issued from the Alaska competi-
tion.

For a given image from the mother dataset, we define the
smart crop 2 as the crop (i.e the area of the mother image)
that keeps the same distribution of costs between the mother
image and the cropped one. To compare the distribution, we
used symmetrized Kullback-Leibler distance:

$$D_{KL}(P, Q) := \sum_i P(i) \log \frac{P(i)}{Q(i)} + \sum_i Q(i) \log \frac{Q(i)}{P(i)}$$  (2)

for given discrete probability distributions P and Q. Note
that in this paper, we consider the cost obtained with the S-
UNIWARD algorithm [15], which is one of the most efficient
and generic embedding algorithm [16].

Note that a brute force approach has to be done to test all
the areas and find the one with the minimal Kullback-Leibler
symmetrized distance. To avoid redundant computation and
reduce the computational cost, we use the histogram integral
approach [17] which is an extension of the well-known integral
image approach, defined in the paper of Viola and Jones
[18]. Given a mother image of size n × m, and a crop area of
size w × w, the complexity for computing all the histograms
for all the positions in the mother image is O(n × m × w^2)

\[ \text{See } \text{https://www.lirmm.fr/~chaumont/LSSD.html} \]

\[ \text{For the NNID, the search space is square areas of dimensions 256x256, 512x512, 1024x1024, and 2048x2048.} \]
whereas it is a much smaller complexity of \(O(n \times m)\) with the histogram integral approach.

Figure 1 shows a mother image of size \(2594 \times 3898\) with its costs histogram, and a near-nested image of size \(256 \times 256\), issued from the smart crop 2 of the mother image, with its costs histogram.

![Fig. 1. A mother image of size 2594 × 3898 with its costs histogram, and a near-nested image of size 256 × 256, issued from the smart crop 2 of the mother image, with its costs histogram.](image)

### 3.2. Relative payload in each dataset ensuring the same security

Given the NNID and an embedding algorithm, we need to find the number of bits to embed in each dataset. In this paper, we only need to focus on the average accuracy obtained by a classifier for each dataset. Using the square root law in order to find the correct payload size to embed in each dataset (such that the security is the same for all) does not ensure, in practice, the same accuracy for all the datasets.

In order to obtain the relative payload size to embed for each dimension, we thus go by a dichotomous method, by running, for each dimension, multiple detections until finding the desired accuracy. Note that we use the Square Root Law as an initialization of this empirical research, for finding the relative payload size for each dimension (see Equation 1). This protocol is the one that was used to set the payload sizes in the experiments of this article.

### 3.3. Invariance in security

We give here the definition of the *invariance in security*. Let’s suppose an NNID that ensures similar pixel distributions, similar contents, the same development, and the *same difficulty* for all the datasets, plus a payload size for the embedding (described in Section 3.2) that ensures the *same empirical security*. We define a deep learning network *invariant in security with respect to the dimension* when its obtained average accuracy is the same whatever the dimensions. This definition, even if very simple, allows a much finer analysis of the invariance property than done previously.

### 3.4. Use of dilated convolution

Some authors have proposed to introduce invariance to the dimension by performing multiple convolutions in parallel on resized versions of the input image [19] [20]. Nevertheless, we cannot perform resampling without losing information on the steganographic noise. We, therefore, propose to use the principle of dilated convolution [21], which consists in spacing the elements of the convolution kernel, as an approximation of the resizing. There is then no more subsampling of the input image.

The dilated convolution operator is defined, for an image, \(z : \mathbb{Z}^2 \rightarrow \mathbb{R}\), a kernel, \(k : \mathbb{Z}^2 \rightarrow \mathbb{R}\), and a scalar, \(d \in \mathbb{N}\), standing for the dilation factor, by [21]:

\[
(z \ast k)(x, y) = \sum_i \sum_j z(x - d \cdot i, y - d \cdot j)k(i, j).
\]

Lots of the networks designed for steganalysis have their first convolution block that serves to remove the influence of the image content [3]. Additionally, most of them have their second convolution block that outputs feature maps whose height and width are equal to these of the image fed to the network.

One thus can modify most of the networks by substituting the second convolution block with an inception block [22] made of dilated convolutions of various dilation factors. This can be easily done without changing neither the number of parameters of the network, either the depth, width, or height of the feature maps.

Applied to the Yedroudj-Net architecture (see Figure 2), the second convolution block still inputs and outputs 30 features maps of size \(256 \times 256\), but the 30 convolution filters of size \(5 \times 5\) are replaced by 10 standard convolutions (without dilatation), 10 convolutions with dilatation of 2 (i.e. the kernel elements are spaced by one pixel), and 10 convolutions with dilatation 4 (i.e. the elements of the kernel are spaced by three pixels), all of them of size \(5 \times 5\).

![Fig. 2. The Yedroudj-Net architecture with the block where dilated convolutions are applied is highlighted.](image)

### 4. EXPERIMENTS

**4.1. Experimental protocol**

In the NNID, each dataset contains images whose dimensions are all equal. Each of them is named \(UNI_w\), with \(w \in \{256, 512, 1024, \text{or} 2048\}\), the size \(w \times w\) of the images of the dataset. \(UNI\) stands for unidimensional.

For each \(UNI\) dataset, the train part is made of 12000 cover/stego pairs (24000 images in total) with 19200 images...
dedicated to the train, and 4800 images to the validation. For each UNI dataset, the test dataset is made of 3000 cover/stego pairs (6000 images in total).

An additional dataset, named MULTI, is derived from the datasets UNI_w, with \(w \in \{256, 512, 1024\}\) such that 4000 cover/stego pairs of each UNI dataset are randomly selected, in order to obtain 12000 cover/stego pairs, and thus a total of 24000 images, with 19200 images dedicated to the train, and 4800 images to the validation.

We used S-UNIWARD\(^5\) for the embedding in the spatial domain with its MATLAB implementation\(^6\).

We trained the SID network\(^7\) and the Yedroudj-Net network\(^8\) using the hyper-parameters from the GitHubs, with a batch of 32. We trained the "Dilated" Yedroudj-Net\(^9\) with the same parameters as Yedroudj-Net, except that the batch size is set to 16.

For all networks, the learning rate is divided by two if the validation’s accuracy does not improve for ten consecutive epochs. We set the maximum number of epochs to 400 for all networks, but if the validation’s accuracy does not improve for fifty consecutive epochs, the learning is stopped. After the training, we saved the weights which gave the best validation accuracy and use them for the test on the test dataset (The standard deviation of networks is extremely low: As an example, for a SID trained on \(256 \times 256\) images it is below 0.02% when testing the top-5 trained networks).

Probably due to unadapted hyper-parameters, SiaSteg\(^{10}\) and Zhu-Net\(^{11}\) networks did not converge. We thus restricted our study to SID and Yedroudj-Net which are two representatives of each family of approaches.

### 4.2. Relative payload for the UNI datasets

As a preliminary experiment, we embedded at a relative payload size of 0.4 bpp (bits per pixels) in the \(256 \times 256\) images, and deduced with the Square Root Law mentioned previously, a relative payload of 0.225 (resp. 0.125, and 0.06875) for \(512 \times 512\) (resp. \(1024 \times 1024\), and \(2048 \times 2048\) images). We then trained and tested a SID on \(256 \times 256\) images, and another one on \(512 \times 512\) images. Accuracies were not at all equal with respectively 69% and 62%, indicating that some hypotheses (asymptotic hypothesis + independencies assumption) of the Square Root Law were not met and that this law could not be used given our experimental conditions.

We thus updated the relative payload size used for each dimension as explained in section 3.2 such that after a manual dichotomous approach the accuracy for each UNI dataset for the Yedroudj-Net model is 76%. Note that this accuracy is

\(^{5}\)See [http://dde.binghamton.edu/download/stego_algorithms/](http://dde.binghamton.edu/download/stego_algorithms/)

\(^{6}\)SID is from the GitHub of SiaSteg: [https://github.com/SiaStg/SiaStegNet](https://github.com/SiaStg/SiaStegNet)

\(^{7}\)Yedroudj-Net: [https://github.com/yedmed/steganalysis](https://github.com/yedmed/steganalysis)

\(^{8}\)Dilated Yedroudj-Net: [https://github.com/Kevin-Planolles/steg...](https://github.com/Kevin-Planolles/steg...)

\(^{9}\)SiaSteg: [https://github.com/SiaStg/SiaStegNet](https://github.com/SiaStg/SiaStegNet)

\(^{10}\)Zhu-Net: [https://github.com/1204BUPT/Zhu-Net-image-steganalysis](https://github.com/1204BUPT/Zhu-Net-image-steganalysis)

<table>
<thead>
<tr>
<th>Table 1. Relative payload for each dimension.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimension</td>
</tr>
<tr>
<td>256</td>
</tr>
<tr>
<td>512</td>
</tr>
<tr>
<td>1024</td>
</tr>
</tbody>
</table>

A meaningful accuracy that lets a sufficient margin for future experiments.

The table\(^{12}\) gives the adjusted payload for each dimension and the corresponding accuracy.

### 4.3. Learn on a UNI dataset

Our first case study consists to learn a network on a UNI dataset (i.e. images have the same dimensions) and evaluating its invariance in security on the other UNI dataset. We note as ARCHI-DIM, where ARCHI denotes the name of the architecture, DIM denotes the dimension on which the network was trained.

The results for SID, Yedroudj-Net, noted \(Y\), and Dilated-Yedroudj-Net, noted \(DY\), are reported in Table\(^2\). The diagonals correspond to the clairvoyant scenario without any mismatch in dimension. The accuracy for the diagonal values for SID (resp. \(DY\)) are close. It confirms that the security by accuracy defined in Section 3.2 is a fine grain definition of the security since given a detector, for each UNI dataset it gave close accuracy (close to 69.9% for SID and close to 77.5% for \(DY\)).

<table>
<thead>
<tr>
<th>Table 2. Accuracies for the SID, the Yedroudj-Net (noted (Y)), and the Dilated-Yedroudj-Net (noted (DY)), models.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dim</td>
</tr>
<tr>
<td>256 \times 256</td>
</tr>
<tr>
<td>512 \times 512</td>
</tr>
<tr>
<td>1024 \times 1024</td>
</tr>
</tbody>
</table>

| Dim | Y-256 | Y-512 | Y-1024 |
|---|
| 256 \times 256 | 76.97% | 73.48% | 71.76% |
| 512 \times 512 | 74.55% | 76.38% | 74.97% |
| 1024 \times 1024 | 72.83% | 73.57% | 76.78% |

| Dim | DY-256 | DY-512 | DY-1024 |
|---|
| 256 \times 256 | 77.7% | 76.25% | 71.92% |
| 512 \times 512 | 75.21% | 77.3% | 76.2% |
| 1024 \times 1024 | 72.03% | 76.88% | 77.53% |

Looking at the non-diagonal results it appears that the performance systematically decrease compared to the diagonal. The biggest performance loss is for the learning at 1024 when evaluated on smaller dimensions, with a loss from 8% (SID) to 5% (Yedroudj-Net and Dilated-Yedroudj-Net).

SID and Yedroudj-Net are two representatives of the two families of networks accepting images of various dimensions.
and are also well-established networks. The above results let us guess that current networks from the literature are not intrinsically invariant to the change in dimension since their accuracies, when confronted with dimension never seen during the learning, are not constant.

Going deeper into the observation of the variance in security phenomenon, considering for example the DY-512, (see Table 2), it appears that the accuracies are very similar (around 76%) for a test on the lower dimension $256 \times 256$ or on the higher dimension $1024 \times 1024$, but the behavior varies greatly with image size. Indeed, as observed in Table 3 when tested on a lower dimension, the model tends to over-classify in stego, while when tested on a higher dimension, it over-classifies images as covers.

### Table 3. Confusion matrices of DY-512.

<table>
<thead>
<tr>
<th>Truth</th>
<th>Pred.</th>
<th>Cover</th>
<th>Stego</th>
<th>Truth</th>
<th>Pred.</th>
<th>Cover</th>
<th>Stego</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cover</td>
<td>2140</td>
<td>860</td>
<td></td>
<td>Cover</td>
<td>2530</td>
<td>470</td>
<td></td>
</tr>
<tr>
<td>Stego</td>
<td>565</td>
<td>2435</td>
<td>Stego</td>
<td>917</td>
<td>2083</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Percentage</td>
<td>45%</td>
<td>55%</td>
<td>Percentage</td>
<td>56%</td>
<td>43%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We checked if this variance in security was due to a wrong decision threshold that should be updated when facing a dimension not seen during the learning. It appears in experiments with Y-512 and SID-512 that updating the threshold does not change the accuracy for the unseen dimensions.

We also checked if there was a difference in the noise pattern depending on the dimension which could explain the variance in performance. Using the activation map (using integrated gradient [23]) for a few stego images of size $512 \times 512$, and the associated nearly nested images $256 \times 256$, such that the first group is well classified and the second group is wrongly classified, it appears that the activation maps are very close. Thus the latent space may be very sensitive to small differences indicating that the feature maps are not sufficiently robust (i.e varies) to the change in dimension.

Finally, we should mention that the introduction of the dilated convolution to the Yedroudj-Net (see Table 2) increases the accuracy by 1% (see diagonal results) and slightly improves the results on unseen dimensions (non-diagonal results), so probably encouraging the invariance to the dimension.

### 4.4. Learn on the MULTI dataset

Our second case study consists of learning a network on the MULTI dataset (i.e. images have various dimensions) and evaluating its invariance in performance on the UNI datasets. We refer as ARCHI-MULTI, the networks trained on multiple dimensions, where ARCHI denotes the name of the architecture.

The results for SID, Yedroudj-Net, and Dilated-Yedroudj-Net are reported in Table 4. While those models are not invariant with respect to dimension, the variations in accuracies are less important compared to learning on a UNI dataset, particularly for $512 \times 512$ and $1024 \times 1024$ images. This confirms that training on multiple dimensions improves the invariance, which was a trick already used in past publications, as a measure for improving the robustness to various dimensions.

Nevertheless, as we can see in Table 4 the invariance is not reached since the accuracy for the dimension $256 \times 256$ is lower by 1.6% to 3.7% than those obtained at dimension $512 \times 512$ or $1024 \times 1024$. This phenomenon is also observed with the Dilated-Yedroudj-Net-MULTI which highlights the fact that future work has to be done to ensure invariance. We should also note that the Dilated-Yedroudj-Net-MULTI allows obtaining the best results which is a piece of interesting practical information for the topic of invariance in dimension.

### 5. CONCLUSIONS

In this paper, we explain how to build the Near Nested Image Datasets (NNID) thanks to a smart crop 2 applied to an initial dataset. Those NNI Datasets, all each containing images of the same dimension, have similar pixel distribution, similar semantic content, same development, and the same distribution of costs such that we talk about same difficulty for all of the datasets. We also propose to dichotomously find a relative payload size for each dataset such that a pre-chosen neural network exhibit the same accuracies, leading to the notion of same security for all the cover/stego datasets. During the building of the NNI Datasets, we thus exhibit that the theoretical results of the square root law might not apply in a real-world context.

Then, using the resulting cover/stego datasets allows us to observe that even in the case of learning with images of various dimensions, there is not any property of invariance in security for the current state-of-the approaches. We also remarked a difference in behavior depending on whether the network was tested on a larger or smaller dimension than the one on which it was trained. In the case of a lower dimension, there is an overestimation of the number of stegos while in the case of a higher dimension, there is an overestimation of the number of covers. Finally, we proposed a new architecture, Dilated-Yedroudj-Net, which gave better results than the other networks.\(^{11}\)

\(^{11}\)FAQ can be found there [https://www.lirmm.fr/~chaumont/publications/FAQ-ICASSP2023.pdf](https://www.lirmm.fr/~chaumont/publications/FAQ-ICASSP2023.pdf)
6. REFERENCES


