Architectural Modifications to Enhance Steganalysis with Convolutional Neural Networks

Remigiusz Martyniak[1][0009-0005-9337-1082] and Bartosz Czaplewski[1][0000-0001-7904-5567]

[1] Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, 11/12 Gabriela Narutowicza Street, 80-233 Gdańsk, Poland
bartosz.czaplewski@pg.edu.pl

Abstract. This paper investigates the impact of various modifications introduced to current state-of-the-art Convolutional Neural Network (CNN) architectures specifically designed for the steganalysis of digital images. Usage of deep learning methods has consistently demonstrated improved results in this field over the past few years, primarily due to the development of newer architectures with higher classification accuracy compared to their predecessors. Despite the advances made, further improvements are desired to achieve even better performance in this field. The conducted experiments provide insights into how each modification affects the classification accuracy of the architectures, which is a measure of their ability to distinguish between stego and cover images. Based on the obtained results, potential enhancements are identified that future CNN designs could adopt to achieve higher accuracy while minimizing their complexity compared to current architectures. The impact of modifications on each model's performance has been found to vary depending on the tested architecture and the steganography embedding method used.

Keywords: Convolutional neural network, Deep learning, Steganalysis, Steganography.

1 Introduction

Every day, billions of images and graphics are transmitted over the Internet. However, this medium also provides the opportunity to conceal information within them in a way that is imperceptible to the human eye. This form of communication, where digital images act as carriers for hidden data, requiring specialized analysis for extraction, is considered a secure means of conveying information. In such scenarios, only the intended recipient, equipped with the necessary software to reveal the concealed information within the image, can decipher the true message being conveyed.

The field of techniques of concealing information is known as steganography. Steganography methods are used to conceal covert communication by embedding information within another medium known as the cover object, e.g. digital image.

Conversely, the development of techniques designed to detect hidden information falls under the field of steganalysis. It plays a crucial part in the information systems security field and is dedicated to detecting the presence of steganographic techniques.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

It is important to emphasize that steganalysis primarily focuses on confirming the presence of a hidden message within the cover object, rather than extracting it.

In research related to steganography and steganalysis, two embedding methods are often used: Wavelet Obtained Weights (WOW) [1] and Spatial Universal Wavelet Relative Distortion (S-UNIWARD) [2]. Both algorithms are content-adaptive, i.e. information is embedded primarily within the regions of an image where the complexity in terms of patterns and pixel value diversity is the highest. This approach makes the detection of hidden communication by steganalysis methods more difficult.

For many years, steganalysis has relied on algorithms grounded in advanced statistical models, referred to as SRM (Statistical Rich Models) [3]. Nevertheless, in recent years, the concept of using machine learning for this purpose has gained prominence. The primary goal of these algorithms is binary classification, determining whether a given digital image is containing a hidden information or not.

Currently, the most promising results in digital image steganalysis are achieved through the use of Deep Learning methods [4] employing Convolutional Neural Networks (CNNs). As noted in [5], steganalysis has traditionally followed a two-stage paradigm, involving manual feature extraction in the initial stage, followed by classification using Ensemble Classifiers or Support Vector Machines in the subsequent stage. However, the emergence of Deep Learning techniques has revolutionized steganalysis by unifying and automating these two distinct stages, leading to advancements that surpass the Rich Models with Ensemble Classifiers [5].

Therefore, the initial objective of this study was to identify the best-performing CNN architectures specifically designed for steganalysis. The second objective was to conduct experiments on these architectures by creating variants of them with structural modifications. The rationale behind conducting such experiments was to determine how specific modifications would impact the model's performance. Thus, structural changes that could increase the classification accuracy of state-of-the-art architectures would provide a contribution to the field of steganalysis.

The goal of this article is to present the research findings concerning the application of machine learning for image steganalysis. The research aims to investigate the impact of modifying CNNs on steganalysis performance. To the best of the authors' knowledge, similar results have not been previously documented in the literature.

The contributions of this paper are as follows. The paper explores the effects of various architectural modifications to the state-of-the-art CNNs on model performance. 17 variants for the Yedroudj-Net and 20 variants for the GBRAS-Net architectures were proposed. The study evaluates how individual modifications could introduce enhancements to existing networks that either improve classification efficiency or reduce the number of trainable parameters. The obtained results allowed to determine specific directions for further research.

The related work section provides an overview of prior research. The methodology section presents the research approach, the dataset, types of experiments, as well as the experimental setup details. The conclusions are presented in the last section.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

2 Related Work

Firstly, autoencoders were used for steganalysis [6]. The study demonstrated that initializing a CNN with filters from a pre-trained stack of convolutional autoencoders, combined with feature pooling layers, yielded promising results.

Subsequently, a CNN was introduced in [7], which was capable of autonomously learning feature representations via multiple convolutional layers. This model unified the feature extraction and classification steps within a single architecture.

Another application of CNNs in steganalysis was explored in [8]. In this scenario, the authors consistently used the same embedding key. Experimental results for a proposed CNN and a fully connected NN outperformed the RM+EC approach.

In [9] the authors employed CNNs as base learners for ensemble learning and tested various ensemble strategies. Their methodology included recovery of lost information caused by spatial subsampling in the pooling layers during feature vector formation.

JPEG steganalysis research was presented in [10]. The authors attempted to adapt CNNs to a comprehensive feature set through network pre-training. The primary challenge in surpassing rich-model-based frameworks were training convergence issues.

Another JPEG research was proposed in [11]. The authors incorporated JPEG awareness into CNNs. They introduced the catalyst kernel allowing the network to learn kernels more relevant for detecting stego signals introduced by JPEG steganography.

A method against J-UNIWARD method was presented in [12]. The author confirmed the role of both the pooling method and the depth of CNNs. It was demonstrated that a 20-layer CNN generally outperforms the most advanced feature model-based methods.

The Xu-Net against S-UNIWARD and HILL methods was presented in [13]. The results show the usefulness of using absolute values in early feature maps and the negative impact of larger filter sizes in convolutional layers on network efficiency.

Next, the SR-Net [14] utilized four original types of layers. In contrast to other nets, SR-Net doesn't initially employ a high-pass preprocessing filtering layer. However, the depth of this network is very high, leading to high model complexity.

Another significant architecture is Yedroudj-Net [15]. Yedroudj-Net employs the Trunc activation function [16], which limits data values to a chosen range. This eliminates the occurrence of large values that would be processed by subsequent layers.

The repeated use of the stego-key was explored in [17]. The study led to the CNN with the state-of-the-art efficiency while having 20 times fewer learnable parameters, which means easier convergence and reduced memory and computing power demands.

The GBRAS-Net [18] incorporated the depthwise layer and the separable layer. The goal is to achieve an effect similar to a regular convolutional layer while reducing the number of parameters. This architecture does not utilize fully connected layers.

The work presented in this paper is based on two different deep neural networks: Yedroudj-Net [15] and GBRAS-Net [18]. These models were selected due to their high performance in terms of classification accuracy. Both models employ preprocessing that includes 30 predefined high-pass filters. The weights of these filters are not learned during the training process. The output signal from the preprocessing convolutional layer is propagated through the deep neural network, where further processing occurs.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

3 Methodology

The steganalysis begins with cover and stego images serving as the input (Fig. 1). These images undergo processing by a modified CNN. The model's output is a binary result: '0' for a cover image prediction or '1' for a stego image prediction. Based on the obtained model's output and true labels assigned to the input images, the overall classification accuracy of the architecture variant can be calculated.

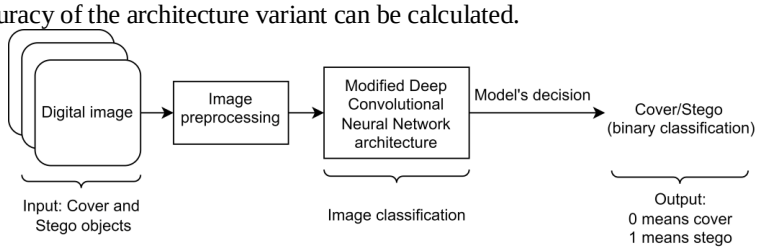

Fig. 1. Steganalysis workflow using a CNN.

3.1 Dataset

For this study, the BOSSBase 1.01 dataset [19] was used for training, validation, and testing. The rationale behind choosing BOSSBase stems from the fact that it is the most commonly used dataset in the steganalysis field. This facilitates the comparison of results with those described in other studies. The mentioned dataset consists of 10,000 grayscale images in PGM format. Following the approach presented in the aforementioned papers, each image was resized from 512×512 px to 256×256 px. It significantly reduces the computational resources required for image processing, thereby reducing the training time for each variant.

The dataset was divided into three subsets: 8,000 images for the training set, 1,000 images for the validation set, and the remaining 1,000 images for the testing set. This 8:1:1 split ratio ensures that the neural network receives ample training examples while maintaining a sufficient number of images for accurate testing. This particular split also proved to yield the best results in the test scenarios described in [20].

3.2 Steganographic Techniques

For the embedding algorithms, WOW and S-UNIWARD with embedding capacity set to 0.4 bpp were used. These algorithms are widely employed in steganalysis research work, allowing for meaningful comparison of obtained results with other studies.

All of the original 10,000 images (cover objects) from BOSSBase were embedded with a message using a different stego-key for each image. This process resulted in a dataset comprising a total of 20,000 images: 10,000 cover and 10,000 stego images. MATLAB implementations from [21] were used for the embedding process, with minor code modifications to load and embed the payload in large batches of images.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

3.3 Architectures

The selection of models for the experiments was based on three primary criteria. Firstly, their selection was based on their classification accuracy stated in the papers. Secondly, it was based on the number of trainable parameters (architecture's complexity). Lastly, the chosen models were required to exhibit distinct architectural approaches. For these reasons, Yedroudj-Net and GBRAS-Net were selected. They exhibited high classification accuracy, relatively low complexity, and notable structural differences. A detailed comparison of both CNNs, including their architecture and training hyperparameters, is provided in Table 1.

Table 1. Comparison of Yedroudj-Net and GBRAS-Net.

Information	Yedroudj-Net	GBRAS-Net
No. of trainable parameters	445 k	166 k
Preprocessing stage	Convolutional layer with 30 5x5 SRM filters	Convolutional layer with 30 5x5 SRM filters
No. of depthwise-separable layers	0	4
No. of convolutional layers	5	8
No. of parameters added by depthwise-sep. and conv. layers	298 k	166 k
Filter (kernel) sizes	5x5 and 3x3	3x3 and 1x1
No. of filters in conv. layers	30, 30, 32, 64, 128	30, 30, 60, 60, 60, 60, 30, 2
Activation after conv. layers	Trunc, ReLU	ELU
Pooling type after conv. layers	Average Pooling with 5x5 kernel and (2,2) stride	Average Pooling with 2x2 kernel and (2,2) stride
Weight initialization method	Glorot Uniform	Glorot Uniform
No. of dense layers	3	0
No. of neurons in subsequent dense layers (input-output)	128-256, 256-1024, 1024-2	-
No. of trainable parameters in dense layers	263 k	0
Activation after dense layers	ReLU	-
Optimizer	SGD	Adam
Learning rate	0.01	0.001
Weight decay	0.0001	0
Momentum	0.95	0.2
Pooling at the output	Global Average Pooling	Global Average Pooling
Activation function at the output	Softmax	Softmax

3.4 Types of Experiments

The changes introduced to the architectures can be categorized into three main types:

- Addition and removal of dense or convolutional layers. These alterations of this kind typically have the most significant impact on both the model's accuracy and complexity.
- Modifications in the structure of individual layers within the original architecture. This category primarily involves changes such as altering filter sizes in successive convolutional layers.
- Simple-to-implement changes that do not increase the model's complexity in terms of the number of trainable parameters but do affect the training process and may, for example, prolong it. Examples include using a different weight

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

optimization algorithm, weight initialization method, and activation functions after each convolutional layer.

A detailed description of the variants is provided in the Results Section.

3.5 Metrics

For each experiment, the training and validation accuracy values for every epoch were calculated. It allowed to closely monitor the model's learning process. Moreover, the final classification accuracy for the testing dataset was calculated, which is the main performance evaluation criterion in this paper. It provides valuable information about the overall model's performance and its ability to generalize to unseen data.

Classification accuracy alone does not provide a comprehensive assessment of the model's performance. This is a very general measure that does not provide a complete picture of the behavior of the analyzed model. Therefore, two additional metrics were utilized: Receiver Operating Characteristic (ROC) curve and the confusion matrix.

ROC curve provides a comprehensive visualization of a model's True Positive Rate (TPR) against its False Positive Rate (FPR) across all classification thresholds. In steganalysis, this curve offers insights into the trade-off between correctly identifying stego images and incorrectly classifying cover images, which helps to visually compare the performance of different models. This is the best method for comparing models. One can compare TPRs at a given FPR or compare Area Under Curve (AUC) to indicate a better detector.

Confusion matrix complements ROC curve by providing a detailed breakdown of a model's predictions, including true positives, true negatives, false positives, and false negatives. This breakdown helps to identify specific areas for improvement in a model's performance. Moreover, all other measures, such as precision, recall, F1 score, or specificity, can be derived from the confusion matrix.

The ROC curve, confusion matrix, and overall classification accuracy results are the most important evaluation metrics to determine the performance of a deep neural network in binary classification problems like steganalysis.

3.6 Experimental Setup

The experiments were performed locally using NVIDIA GeForce RTX 2080 Ti GPU with the support of CUDNN v11.6. To implement the variants of Yedroudj-Net and GBRAS-Net architectures, Python v3.9.16 programming language, PyTorch v1.13.1+cu116, and PyTorch Lightning v2.0.0 libraries were used. The results of the experiments were recorded using Neptune.ai API and stored on its cloud platform.

4 Results

4.1 Modifications for Yedroudj-Net

The Yedroudj-Net architecture is more complex than GBRAS-Net, i.e. 445 thousand trainable parameters, which is 279 thousand more. This difference primarily stems from

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

the inclusion of three fully connected layers in the classification stage. Therefore, the initial experiments focused on simplifying the original architecture.

Simplification Strategies. In the first two Yedroudj-Net variants, simplification strategies by removal of dense layers were explored:

- Variant 1: Removal of the last dense layer, resulted in a minor decrease in accuracy for WOW, but surprisingly led to an increase of 0.55 pp for S-UNIWARD. Notably, by removing a dense layer with 1024 neurons, a significant reduction of 264 thousand trainable parameters was achieved.
- Variant 2: Removal of the last two dense layers caused a decrease in accuracy by 1.45 pp for WOW and 1.6 pp for S-UNIWARD.

The obtained results raise a question regarding the necessity of utilizing as many as three dense layers from a complexity-to-performance trade-off point of view. A similar simplification approach was adopted for the convolutional layers:

- Variant 3: Removal of the last convolutional layer, which contains 64 filters at the input and 128 at the output, resulted in an increase in classification accuracy by 0.7 pp for WOW and a decrease of 0.15 pp for S-UNIWARD.
- Variant 4: Removal of the last two convolutional layers led to a loss of more than 4 pp in accuracy for both embedding algorithms, making this type of change not justified from a performance perspective.

Expanding the Architecture. In this set of experiments, the objective was to extend the original architecture by incorporating additional and dense layers. Note that after each added convolutional layer, batch normalization and ReLU function were applied.

Initially, one dense layer with 512 neurons was added (variant no. 5). This modification resulted in a minor decrease in accuracy while increasing the number of parameters. It was decided not to further experiment with the addition of more dense layers, as there was no indication that it would enhance the model's performance.

Similarly, one and two convolutional layers were integrated into the Yedroudj-Net architecture after the preprocessing layer:

- Variant 6: Added one convolutional layer with 30 filters, 1×1 kernel size, a stride of 1, and no padding, ensuring that the output feature map has the same dimensions as the input feature map. Classification accuracy increased for both embedding algorithms: by 1.75 pp for WOW and 1.35 pp for S-UNIWARD.
- Variant 7: Added two convolutional layers with 30 filters, but this time employing a 3×3 kernel size, with a stride and padding of 1. What is more, it was decided to maintain uniform filter sizes across all layers, thereby changing kernel sizes of the original first and second Yedrouj-Net layers from 5×5 to 3×3. This adjustment also helped to mitigate the increase in the number of trainable parameters caused by the addition of two convolutional layers. Accuracy increased by 4.5 pp for WOW, and 6.8 pp for S-UNIWARD. The training/validation accuracy plot for this variant is in Fig. 2, and the ROC curve is shown in Fig. 3. According to the results provided in Table 2, the model exhibited a high degree of accuracy in classifying both cover and stego images.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

Fig. 2. Training and validation accuracy for variant no. 7 of Yedroudj-Net.

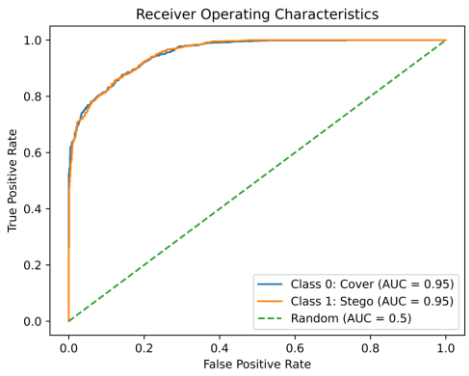

Fig. 3. ROC curve for variant no. 7 of Yedroudj-Net architecture.

Table 2. Normalized confusion matrix for variant no. 7 of Yedroudj-Net.

	Predicted Cover	Predicted Stego
Actual Cover	0.88 (True Negative)	0.12 (False Positive)
Actual Stego	0.15 (False Negative)	0.85 (True Positive)

Convolutional layer parameter tuning. The following set of experiments involved altering key parameters within the convolutional layers, such as kernel sizes and average pooling operation. The results of these experiments are as follows:

- Variant 8: Changing filter sizes from 3×3 to 5×5 in the last three convolutional layers resulted in a decrease in accuracy for both steganography algorithms. Furthermore, it significantly increased the model's complexity as the last three layers contain a higher number of filters compared to the initial layers.
- Variant 9: Altering filter sizes from 5×5 to 3×3 in the first two convolutional layers, resulting in a kernel size of 3×3 for each layer, leading to an increase in accuracy by 0.55 pp for WOW and a decrease by 0.55 pp for S-UNIWARD.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

- Variant 10: Changing filter sizes from 5×5 to 3×3 in the first three convolutional layers and from 3×3 to 5×5 in the last two convolutional layers, led to an increase in classification accuracy by 0.55 pp for WOW and by 0.85 pp for S-UNIWARD. However, this change also significantly increased the number trainable parameters, because the last two convolutional layers of Yedroudj-Net feature a greater number of filters (64 and 128 at the output), compared to these at the beginning of the architecture.
- Variant 11: Altering filter sizes from 5×5 to 1×1 in the first convolutional layer resulted in increased accuracy for both WOW and S-UNIWARD by 0.85 and 1.4 pp, respectively. It also reduced the number of parameters by 36 thousand.

Based on the results obtained from this series of experiments, the most significant conclusion is that employing a smaller kernel size in the initial convolutional layer or layers can lead to improved classification accuracy.

In this series of experiments related to convolutional layers, the impact of average pooling was investigated with the following variants:

- Variant 12: Removal of average pooling led to a 1.7 pp increase in classification accuracy for WOW, and a decrease by 1.85 for S-UNIWARD.
- Variant 13: Adjustment of the average pooling settings from (5,2) to (3,1), where the first number represents the kernel size, and the second number denotes the stride, resulted in a 1.8 pp increase in classification accuracy for WOW, and a 1.65 pp decrease for S-UNIWARD.

Table 3. Classification accuracy results for Yedroudj-Net architecture variants.

#	Description	Change in classification accuracy (in pp)		Change in the number of parameters (in thousands)
		WOW	S-UNIWARD	
1	Removed the last dense layer	-0.25	+0.55	-264
2	Removed the last two dense layers	-1.45	-1.6	-297
3	Removed the last two conv. layers	-4.2	-6.65	-117
4	Added one dense layer	-0.35	-0.25	+524
5	Removed the last conv. layer	+0.7	-0.15	-90
6	Added one conv. layer	+1.75	+1.35	+1
7	Added two conv. layers	+4.5	+6.8	+138
8	5x5 kernel size in the last 3 conv. layers	-0.5	-0.8	+180
9	3x3 kernel size in the first 2 conv. layers	+0.55	-0.55	-28
10	3x3 kernel size in the first 3 conv. 5x5 kernel size in the last 2 conv. layers	+0.55	+0.85	+136
11	1x1 kernel size in the first layer	+0.85	+1.4	-36
12	No average pooling	+1.7	-1.85	0
13	(3,1) average pooling instead of (5,2)	+1.8	-1.65	0
14	Adam optimizer instead of SGD	-5	-1.45	0
15	Kaiming Normal initialization instead of Glorot Uniform	-0.2	-0.3	0
16	ELU activation instead of ReLU	-3.8	+5	0
17	ReLU activation instead of Trunc	+1.3	-1.1	0

Optimization and activation strategies. The final four experiments focused on modifications to the optimization algorithm, weight initialization method, and activation functions. Changing the optimizer from SGD to Adam (variant no. 14), as well as the

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

weight initialization method from Glorot Uniform to Kaiming Normal (variant no. 15) resulted in decreased classification accuracy.

However, more intriguing results emerged when the ReLU activation functions were replaced with ELU activation (variant no. 16). The classification accuracy decreased by 3.8 pp for the WOW algorithm but increased by 5 pp for S-UNIWARD, highlighting the significant role of the activation function in the context of using CNNs for steganalysis, in which case the steganography embedding algorithm used for cover communication is most often unknown.

Contrarily, changing the Truncated Linear Unit (TLU) activation function, which is used twice in Yedroudj-Net, to ReLU (variant no. 17 in Table 3) resulted in an increase of 1.3 pp for WOW and a decrease of 1.1 pp for S-UNIWARD.

In Table 3, the results of the conducted experiments are shown, comprising seventeen different variants of the Yedroudj-Net architecture. Changes in the classification accuracy compared to the original architecture are expressed in percentage points.

4.2 Modifications for GBRAS-Net

Similar to the modifications for Yedroudj-Net, four types of experiments were conducted to assess their impact on the GBRAS-Net architecture performance.

Simplification Strategies. Convolutional and depthwise layers were removed, as the original GBRAS-Net architecture does not incorporate dense layers.

- Variant 1: Removal of five convolutional layers (3rd–7th) led to a decrease in classification accuracy by 1.95 pp for WOW and 1 pp for S-UNIWARD. This change involved the removal of a substantial portion of convolutional layers, resulting in a reduction of trainable parameters by 121.3 thousand.
- Variant 2: Removal of the five aforementioned convolutional layers, along with the 1st depthwise layer. The test of this variant was carried out to assess the significance of depthwise layers. This modification resulted in a major decrease in accuracy, exceeding 5 pp for both embedding algorithms. This underscores the critical role of depthwise layers within the GBRAS-Net.

Additionally, three more variants (no. 3, 4, 5) were explored, involving the removal of three, four and seven convolutional layers, all of which yielded similar results.

Expanding the Architecture. In the subsequent series of experiments, the impact of the following additions was investigated:

- Variant 6: Two convolutional layers at the beginning of the network. These layers consisted of 30 filters, kernel size of 3×3, stride and padding set to 1.
- Variant 7: Two convolutional layers following the 4th depthwise layer. Given that in the original architecture, the convolutional layer before the 3rd and 4th depthwise layer has 60 filters, the added convolutional layers also had 60 filters. They featured a 3×3 kernel size, stride, and padding of 1.

The addition of convolutional layers in both cases resulted in a decrease in accuracy for WOW and S-UNIWARD algorithms. Batch normalization and the ReLU activation function was applied after each additional convolutional layer.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

Furthermore, the impact of incorporating dense layers at the end of the network was examined. Initially, two layers were added (variant no. 8) with 512 and 2 neurons, respectively. In the subsequent variant (no. 9) three dense layers were employed with 512, 1024 and 2 neurons. In both variants, a minimal increase in accuracy for WOW was observed, but conversely, a decrease for S-UNIWARD can be noticed: by 3.05 pp for variant no. 17 and by 0.3 pp in the case of variant no. 18.

Convolutional Layer Parameter Tuning. The impact of kernel size was explored:

- Variant 10: Decrease in filter sizes across subsequent layers, from 5×5 to 3×3, was implemented. The kernel size of the first three convolutional layers was increased from 3×3 to 5×5, while the following convolutional layers maintained a 3×3 kernel size, including the very last layer, which originally features 30 input and 2 output neurons with 1×1 kernel size.
- Variant 11: Further reduction in filter size, from 7×7 to 5×5 and then to 3×3, was applied. The first three convolutional layers had 7×7 kernels, the next two: 5×5, and the subsequent two: 3×3.

Both variants 10 and 11 resulted in reduced performance for both algorithms.

The next three experiments involved increasing filter sizes of the last conv. layers:

- Variant 12: Increase from 1×1 to 3×3 in the last two convolutional layers increased accuracy by 0.15 pp for WOW and by 1.05 pp for S-UNIWARD.
- Variant 13: Changing the filter size of the 5th-8th convolutional layer from 3×3 to 5×5. This alteration led to an increase in accuracy by 0.25 pp for WOW and a decrease of 1.4 pp for S-UNIWARD.
- Variant 14: Changing the filter size of the 5th and 6th convolutional layers from 3×3 to 5×5 and of the 7th and 8th from 3×3 to 7×7. As a result, the accuracy increased by 0.7 pp for WOW and decreased by 0.5 pp for S-UNIWARD. The training/validation accuracy plot for this variant is in Fig. 4, and the ROC curve is shown in Fig. 5. Based on the results presented in Table 4, it can be stated that this variant demonstrates slightly better performance in the correct classification of images that are steganographic.

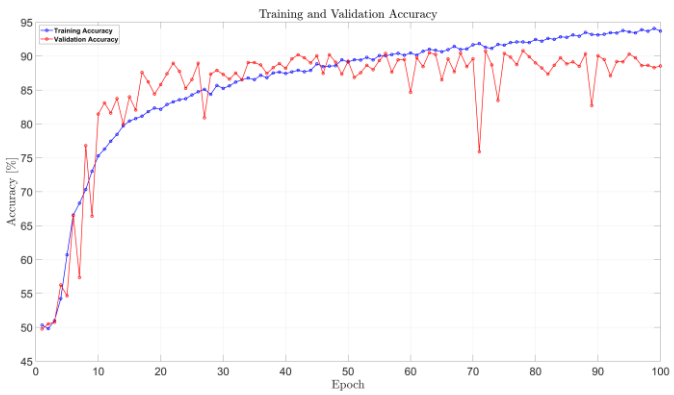

Fig. 4. Training and validation accuracy for variant no. 14 of GBRAS-Net architecture.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

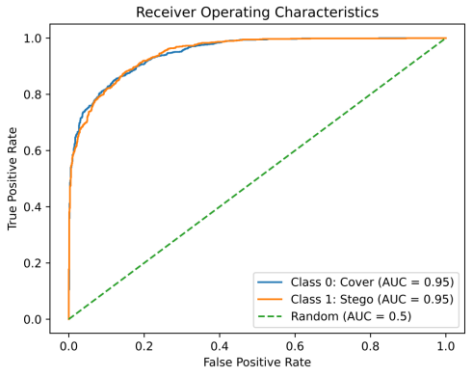

Fig. 5. ROC curve for variant no. 14 of GBRAS-Net architecture.

Table 4. Normalized confusion matrix for variant no. 14 of GBRAS-Net.

	Predicted Cover	Predicted Stego
Actual Cover	0.83 (True Negative)	0.17 (False Positive)
Actual Stego	0.11 (False Negative)	0.89 (True Positive)

The impact of altering the settings of the average pooling operation was explored:

- Variant 15: Removal of the average pooling operation, which was used three times in the architecture, resulted in a decrease by more than 2 pp for both WOW and S-UNIWARD. It is important to note that global average pooling, employed just before the softmax activation function, remained unchanged.
- Variant 16: Adjustment of the average pooling settings from (2,2) to (5,2) led to a decrease in accuracy by 1.6 pp for WOW and an increase by 0.1 pp for S-UNIWARD. Similar results were obtained in the subsequent variant (no. 17), where average pooling (3,1) was applied: a decrease of 1.55 for WOW and by 0.5 for S-UNIWARD.

Optimization and Activation Strategies. Switching from Adam to SGD optimizer (variant no. 18) resulted in a noticeable accuracy decrease for both algorithms.

Using the Kaiming Normal weight initialization method instead of Glorot Uniform (variant no. 19) resulted in an increase by 0.1 pp for WOW and decrease of 0.8 pp for S-UNIWARD.

Replacing ELU activation with ReLU (variant no. 20) resulted in improved accuracy by 0.5 pp for WOW, but also a 1.55 pp decrease for S-UNIWARD.

In Table 5, the results of the conducted twenty experiments for GBRAS-Net are presented. Changes in the classification accuracy compared to the original architecture are expressed in percentage points.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

Table 5. Classification accuracy results for GBRAS-Net architecture variants.

#	Description	Change in classification accuracy (in pp)		Change in the number of parameters (in thousands)
		WOW	S-UNIWARD	
1	Removed five (3^{rd}–7^{th}) conv. layers	-1.95	-1	-121.3
2	Removed five (3^{rd}–7^{th}) conv. and the 1^{st} depthwise layer	-5.35	-7.65	-121.1
3	Removed three (5^{th}–7^{th}) conv. layers	-0.7	-1.2	-97
4	Removed four (4^{th}–7^{th}) conv. layers	-0.65	+1.45	-113
5	Removed seven (2^{nd}–8^{th}) conv. layers	-2.6	-0.35	-129
6	Added two conv. layers	-0.7	-1.45	17
7	Added two conv. layers after the 4^{th} depthwise layer	-0.35	-0.85	66
8	Added two dense layers	+0.25	-3.05	18
9	Added three dense layers	+0.3	-0.3	566
10	Decreasing filter size in subsequent layers: 5x5 → 3x3	-2.8	-4.8	81
11	Decreasing filter size in subsequent layers: 7x7 → 5x5 → 3x3	-2.65	-1.15	287
12	3x3 kernel size in the last two layers	+0.15	+1.05	15
13	Increasing filter size in subsequent layers: 3x3 → 5x5	+0.25	-1.4	216
14	Increasing filter size in subsequent layers: 3x3 → 5x5 → 7x7	+0.7	-0.5	346
15	No average pooling	-2.2	-2.75	0
16	(5,2) average pooling instead of (2,2)	-1.6	0.1	0
17	(3,1) average pooling instead of (2,2)	-1.55	-0.5	0
18	SGD optimizer instead of Adam	-0.7	-2.25	0
19	Kaiming Normal initialization instead of Glorot Unifom	+0.1	-0.8	0
20	ReLU activation instead of ELU	+0.5	-1.55	0

5 Conclusions

Firstly, it is evident that different modifications introduced to CNN architectures yield varying classification accuracies depending on the steganographic algorithms employed to embed the payload within images. Notable examples are as follows.

Yedroudj-Net: The replacement of the ReLU activation with ELU led to a significant decrease in accuracy (by 3.8 pp) for WOW and a substantial increase in classification accuracy for S-UNIWARD (by 5 pp). Notably, this was the only case in which the model exhibited better performance for the S-UNIWARD compared to WOW.

GBRAS-Net: Increasing filter sizes in subsequent layers resulted in improved classification accuracy for WOW but decreased it for S-UNIWARD, like in the case of changing the activation function from ReLU to ELU.

The impact of various architectural changes on the effectiveness of CNNs depends significantly on their fundamental structure and the embedding algorithms used. Consequently, identifying a universal modification that can consistently lead to improved classification performance for any given architecture and embedding algorithms is challenging. However, the following set of modifications could be advantageous.

For Yedroudj-Net: The addition of one or two extra convolutional layers after the preprocessing layer significantly improved the classification accuracy for both algo-

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

rithms without a substantial increase in spatial complexity. This change can also be combined with adjustments to the filter sizes, employing smaller kernels in the initial convolutional layers and larger ones (5×5) in the final two layers. To reduce the spatial complexity of the model with these implemented changes, it is possible to remove the last dense layer to achieve a similar number of trainable parameters as in the original architecture with minimal impact on the neural network performance.

For GBRAS-Net: Increasing the kernel sizes in the last two convolutional layers from 1x1 to 3x3 resulted in an increase in classification accuracy for both algorithms, with only minimal additional spatial complexity. If there is a need to further reduce the complexity (to less than 166 thousand trainable parameters) with minimal impact on classification accuracy, up to five convolutional layers could be removed.

The change of activation function is a simple and cost-effective change that may significantly impact classification accuracy. It is worth noting that the popular ReLU activation function decreases the classification accuracy in the case of S-UNIWARD, but improves it in the case of WOW (variants 16 and 17 in Table 3, and variant 20 in Table 5), e.g. using ELU instead of ReLU decreases the accuracy by 3.8 pp for the WOW but increased by 5 pp for S-UNIWARD in Yedroudj-Net. In some cases, ReLU function is worse at dealing with the problems of vanishing gradient and dead neurons. Similarly, the addition of convolutional layers, especially at the beginning of the architecture with filters of small dimensions (e.g. 1×1 or 3×3), can have a minor impact on network complexity while substantially improving steganalysis results.

Further research could involve the creation of additional variants for the architectures and embedding algorithms studied in this work, as well as different steganographic techniques, such as J-UNIWARD. Future work could also involve experimentation with different variants of preprocessing methods, as the preprocessing stage is a crucial component of current state-of-the-art models. Image datasets other than BOSSBase are also worth investigating as a future work.

Acknowledgments. This study was funded by the Faculty of Electronics, Telecommunications, and Informatics of Gdańsk University of Technology, and by the research subsidy from the Polish Ministry of Science and Higher Education.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Holub, V., Fridrich, J.: Designing steganographic distortion using directional filters. In: 2012 IEEE International Workshop on Information Forensics and Security (WIFS), 234–239 (2012)
2. Holub, V., Fridrich, J., Denemark, T.: Universal distortion function for steganography in an arbitrary domain. EURASIP Journal on Information Security (2014)
3. Fridrich, J.: Steganography in Digital Media: Principles, Algorithms, and Applications. Cambridge University Press, United Kingdom (2009)
4. Czaplewski, B.: Current trends in the field of steganalysis and guidelines for constructions of new steganalysis schemes. Telecommun. Rev. + Telecommun. News 10(3), 1121–1125 (2017)

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4

5. Reinel, T.-S., Raúl, R.-P., Gustavo, I.: Deep Learning Applied to Steganalysis of Digital Images: A Systematic Review. IEEE Access 7, 68970-68990 (2019)
6. Tan, S., Li, B.: Stacked convolutional auto-encoders for steganalysis of digital images. In: Proc. Signal and Information Processing Association Annual Summit and Conference, pp. 1–4. APSIPA, Siem Reap, Cambodia (2014)
7. Qian, Y., Dong, J., Wang, W., Tan, T.: Deep learning for steganalysis via convolutional neural networks. In: Proc. Media Watermarking, Security, and Forensics 2015, MWSF'2015, Part of IS&T/SPIE Annual Symposium on Electronic Imaging, SPIE'2015, pp. 94090J–94090J–10, San Francisco, California, USA (2015)
8. Pibre, L., Pasquet, J., Ienco, D., Chaumont, M.: Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover source-mismatch. In: Proc. Media Watermarking, Security, and Forensics, MWSF, Part of I&ST International Symposium on Electronic Imaging, pp. 1–11. EI, San Francisco, California, USA (2016)
9. Xu, G., Wu, H.Z., Shi, Y.Q.: Ensemble of CNNs for steganalysis: an empirical study. In: Proc. 4th ACM Workshop on Information Hiding and Multimedia Security, pp. 103–107. IH&MMSec'16, Vigo, Galicia, Spain (2016)
10. Zeng, J., Tan, S., Li, B., Huang, J.: Pre-training via fitting deep neural network to rich-model features extraction procedure and its effect on deep learning for steganalysis. In: Proc. Media Watermarking, Security, and Forensics 2017, MWSF'2017, Part of IS&T Symposium on Electronic Imaging, p. 6. EI, Burlingame, California, USA (2017)
11. Chen, M., Sedighi, V., Boroumand, M., Fridrich, S.: JPEG-phase-aware convolutional neural network for steganalysis of JPEG images. In: Proc. 5th ACM Workshop on Information Hiding and Multimedia Security, pp. 75–84. IH&MMSec'17, Drexel University in Philadelphia, PA (2017)
12. Xu, G.: Deep convolutional neural network to detect JUNIWARD. In: Proc. 5th ACM Workshop on Information Hiding and Multimedia Security, pp. 67–73. IH&MMSec'17, Drexel University in Philadelphia, PA (2017)
13. Xu, G., Wu, H.Z., Shi, Y.Q.: Structural design of convolutional neural networks for steganalysis. IEEE Signal Processing Letters 23(5), 708-712 (2016)
14. Boroumand, M., Chen, M., Fridrich, J.: Deep Residual Network for Steganalysis of Digital Images. IEEE Transactions on Information Forensics and Security 14(5), 1181-1193 (2019)
15. Yedrouj, M., Comby, F., Chaumont, M.: Yedrouj-Net: An efficient CNN for spatial steganalysis. In: Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'2018, Calgary, Alberta, Canada (2018)
16. Ye, J., Ni, J., Yi, Y.: Deep Learning Hierarchical Representations for Image Steganalysis. IEEE Transactions on Information Forensics and Security 12(11), 2545-2557 (2017)
17. Czaplewski, B.: An Improved Convolutional Neural Network for Steganalysis in the Scenario of Reuse of the Stego-Key. In: 28th International Conference on Artificial Neural Networks and Machine Learning – ICANN 2019, Munich (2019)
18. Reinel, T.-S., et al.: GBRAS-Net: A Convolutional Neural Network Architecture for Spatial Image Steganalysis. IEEE Access 9, 14340-14350 (2021)
19. Binghamton's University DDE download section, https://dde.binghamton.edu/download, last accessed 2023/11/22
20. Tabares-Soto, R., et al.: Sensitivity of deep learning applied to spatial image steganalysis. PeerJ Comput. Sci. 7, e616 (2021)
21. Binghamton's University DDE Steganographic Algorithms section, https://dde.binghamton.edu/download/stego_algorithms, last accessed 2023/11/22

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63751-3_4