JPEG Steganalysis Using Leaked Cover Thumbnails

Martin Beneš
Universität Innsbruck
martin.benes@uibk.ac.at

Benedikt Lorch
Universität Innsbruck
benedikt.lorch@uibk.ac.at

Rainer Böhme
Universität Innsbruck
rainer.boehme@uibk.ac.at

***Abstract*—JPEG steganalysis seeks to detect the presence of hidden messages in JPEG images. In some cases, additional information is available that may aid JPEG steganalysis. One instance is a thumbnail of the cover image, which may be acquired from a seized device or from the image file because the steganography software did not update the metadata. This paper studies how a leaked cover thumbnail can aid JPEG steganalysis. We distinguish two types of thumbnails: pre-compression thumbnails produced from uncompressed pre-covers, and post-compression thumbnails produced from JPEG-compressed covers. The analysis proceeds in three stages. First, we quantify the amount of information left in the cover thumbnail. Second, we demonstrate that post-compression cover thumbnails allow for near-perfect steganalysis by re-creating the thumbnail. Third, we study how an EfficientNet detector can benefit from the additional cover information in pre-compression thumbnails.**

***Index Terms*—steganalysis, thumbnail, side information**

I. Introduction

The goal of a steganographer is to secretly send a message. To do so, the message is embedded inside an innocuously looking cover object, typically by modifying an existing cover. One common type of covers are digital images. When using a lossless format, the message can be embedded into the image pixels. In lossy formats, such as JPEG, the message can be embedded into the quantized discrete cosine transform (DCT) coefficients. Adaptive embedding methods, e.g., UERD [1] or J-UNIWARD [2], place the changes in the cover based on its content. By contrast, non-adaptive methods, such as nsF5 [3], spread the changes uniformly over all embeddable positions.

The goal of a steganalyst, the adversary of the steganographer, is to distinguish covers from stego objects that contain a secret message. Steganalysis can target a steganographic method or be *blind* – applicable to any method. The standard approach to blind steganalysis involves a detector preceded by feature extraction [4]. In recent years, neural networks surpassed the feature-based approach in terms of performance, at the cost of limited explainability. For example, the contributions to the ALASKA2 competition [5] were dominated by deep learning, in particular the EfficientNet architecture.

Specific circumstances may facilitate steganalysis in practice [6]–[10]. Imagine the steganalyst obtains a thumbnail of the cover. Despite being subsampled and re-compressed, the cover thumbnail reveals partial information about the steganographic changes. Exploiting such information could improve detection performance.

The cover thumbnail could be acquired from a seized device as an outdated record in the operating system's (OS) thumbnail cache, which are often updated with some delay, based on user interactions [11]. The cover thumbnail could also appear in the stego object, e.g., by careless copying of the cover JPEG metadata to the stego object [12]. The recent image cropping bug *aCropalypse* [13] in popular software exemplifies how parts of the file can persist after editing.

To our best knowledge, there has not been any attempt of using thumbnails to support steganalysis. This paper contributes

1) a typology of thumbnails specific to JPEG steganalysis;
2) a quantification of the advantage the steganalyst gains by capturing the leaked cover thumbnail; and
3) empirical measurements of the improvement in detector performance using two types of thumbnails.

It is structured as follows: Section II presents related work in steganalysis and image forensics. Section III recalls how image thumbnails are generated. Section IV quantifies the amount of information in leaked cover thumbnails. Section V describes our experiments on using the thumbnail in a detector, and Sec. VI presents the results. Section VII discusses and Sec. VIII concludes the paper.

II. Related Work

Previous cases of side information in steganalysis are side-channel awareness (SCA) [8], [14]–[16] and phase awareness (PHA) [17]–[19]. SCA uses the distortion as an attention map, multiplied or concatenated along the channel axis with the main features. PHA exploits pixel dependencies in the decompressed domain caused by the JPEG block structure.

Another line of work uses physical side information, e.g., the color filter array (CFA) [6], [7] or the message length [8]–[10]. In both cases the side information is one-dimensional: the message length is scalar, and the CFA is a category due to the limited number of CFA configurations in practice. Compared to these cases, a thumbnail image conveys a large amount of information, positionally related to the main image.

In the related field of image forensics, thumbnails have been used to detect image manipulations [20]–[22]. In digital forensics, thumbnails were also used to efficiently scan large data partitions [11]. To our best knowledge, no research has studied thumbnails as side information for steganalysis so far.

III. Thumbnail Generation

A thumbnail image is a miniaturized, compressed version of a main image, used for preview without the need to decompress the main image. Thumbnails appear (1) in the

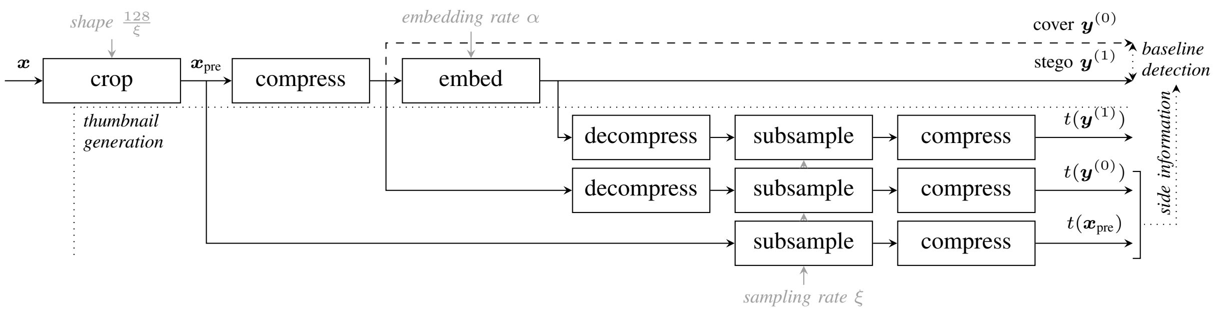


Fig. 1: System model of thumbnail generation for different thumbnail types: pre-cover thumbnail $t(\boldsymbol{x}_{\text{pre}})$, and post-compression cover and stego thumbnails $t(\boldsymbol{y}^{(0)})$ and $t(\boldsymbol{y}^{(1)})$, respectively.

image metadata, e.g., generated by the capturing device,[1] and (2) in OS thumbnail caches [23], where they are often updated only lazily.

Figure 1 shows the thumbnail generation pipeline. The thumbnail creation comprises at least image subsampling and JPEG compression. We denote an image in the spatial domain as $\boldsymbol{x}$ and in the DCT domain as $\boldsymbol{y}$. Thumbnails are denoted $t(\boldsymbol{x})$ and $t(\boldsymbol{y})$. Note that generating $t(\boldsymbol{y})$ involves an additional decompression step. An image $\boldsymbol{x}$ has N elements indexed as $x[1]$, $x[2]$, etc., a thumbnail $t(\boldsymbol{x})$ has M elements indexed $t(x)[1]$, $t(x)[2]$, etc. The elements are called pixels in the spatial domain, and coefficients in the DCT domain. Following the conventions in [24], a stego object has a superscript $\boldsymbol{y}^{(1)}$, while the corresponding cover object is $\boldsymbol{y}^{(0)}$.

The pre-cover $\boldsymbol{x}_{\text{pre}}$ is an image in the spatial domain, available to the steganographer, but not to the steganalyst. The cover $\boldsymbol{y}^{(0)}$ is a JPEG-compressed version of $\boldsymbol{x}_{\text{pre}}$. The stego $\boldsymbol{y}^{(1)}$ is constructed from $\boldsymbol{y}^{(0)}$ by embedding at rate α.

We distinguish three types of thumbnails: stego thumbnails $t(\boldsymbol{y}^{(1)})$, post-compression cover thumbnails $t(\boldsymbol{y}^{(0)})$, and pre-compression cover (or pre-cover) thumbnails $t(\boldsymbol{x}_{\text{pre}})$. The term cover thumbnail subsumes both post-compression and pre-compression cover thumbnails.

a) Image subsampling: Image subsampling is defined in Eq. 1 using 1-D notation for simplicity. The image $\boldsymbol{x}$ is first reconstructed in the continuous domain by convolving it with an interpolation kernel $\varphi(d)$ [25], and then resampled at sampling rate $\xi = \frac{M}{N}$, with grid shift $\frac{1}{2}$, to obtain the subsampled image $t(\boldsymbol{x})$:

$$t(x)[j] \propto \sum_{i=1}^{N} \varphi\left(j\xi^{-1} - i - \frac{1}{2}\right) x[i] \ . \tag{1}$$

The kernel $\varphi(d)$ is a continuous function, centered around 0, used to interpolate values between the pixels. Practical kernels have bounded support $d \in [-h, h]$. The common choices defined in Eq. 2 are the nearest, linear, or cubic kernels,

$$\varphi(d) = \begin{cases} \mathbb{1}_{|d| \le \frac{1}{2}} & \text{nearest} \\ (1-|d|)\mathbb{1}_{|d|<1} & \text{linear} \\ \left.\begin{array}{ll} \frac{3}{2}|d|^3 - \frac{5}{2}|d|^2 + 1 & |d| \in [0,1) \\ -\frac{1}{2}|d|^3 + \frac{5}{2}|d|^2 - 4|d| + 2 & |d| \in [1,2) \\ 0 & |d| \ge 2. \end{array}\right\} & \text{cubic} \end{cases} \tag{2}$$

Narrow kernel support can violate the sampling theorem and cause aliasing. Aliasing can be mitigated with anti-aliasing (AA) – widening the kernel support, $\varphi_{AA}(d) = \varphi(d/h)/h$. The nearest kernel with anti-aliasing is also known as the box kernel, $\varphi_{AA}(d) = (\lfloor d+h \rfloor - \lceil d-h \rceil)^{-1}\mathbb{1}_{|d|\le h}$. The parameter h controls the kernel width. In the box kernel, h relates to the number of elements to average over [25], [26].

b) JPEG compression: JPEG compression of grayscale image involves: DCT transform per 8×8 pixel block, (lossy) quantization, and encoding. A quality factor (QF) controls the rate–distortion tradeoff. JPEG decompression performs the same steps in reverse order [25].

In Fig. 1, the baseline detection task for the steganalyst is to distinguish between $\boldsymbol{y}^{(0)}$ and $\boldsymbol{y}^{(1)}$. In this paper, we explore to what extent capturing a post-compression cover thumbnail $t(\boldsymbol{y}^{(0)})$ or a pre-compression cover thumbnail $t(\boldsymbol{x}_{\text{pre}})$ can aid steganalysis.

IV. Feasibility Study

This section quantifies the number of changes between a post-compression cover thumbnail $t(\boldsymbol{y}^{(0)})$ and a post-compression stego thumbnail $t(\boldsymbol{y}^{(1)})$, as denoted in Fig. 1. We assume that a higher number of differences between cover and stego thumbnails increases the advantage gained by the steganalyst. This section omits pre-compression thumbnails, which are compressed once, while stego thumbnails are compressed twice. Hence, an evaluation of their differences would be distorted by the second JPEG compression.

A. Measuring the information left in the thumbnail

The mismatch between the thumbnails is measured using the change rate β [27, Ch. 4], the number of changes in

[1]The limits imposed by JPEG metadata and the DCF standard for thumbnails are compared in Appendix A.

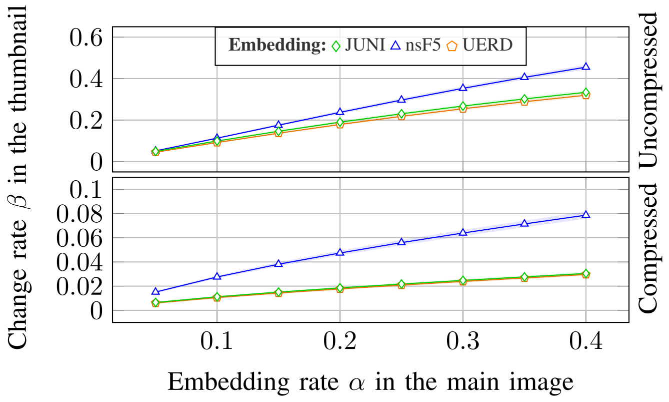


Fig. 2: Impact of the embedding rate α on the change rate β between cover and stego thumbnail.

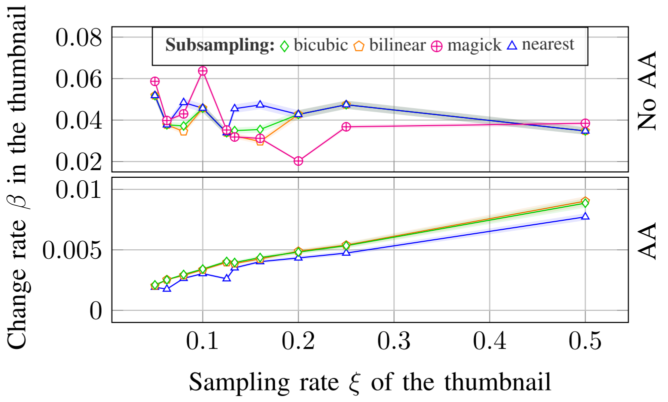


Fig. 3: Impact of the sampling rate ξ and anti-aliasing (AA) on the change rate β between compressed cover and stego thumbnails.

the stego thumbnail per cover element,

$$\beta = \frac{\text{\# of mismatching elements}}{\text{\# of cover elements}} . \quad (3)$$

The change rate of Eq. 3 is applicable to pixels in the spatial domain and to coefficients in the DCT domain.

1) Dataset: We randomly choose 1 000 raw images from the ALASKA2 dataset [5], convert them to grayscale with the `dcraw` tool, and crop them to 2560×2560. 967 images large enough to generate thumbnails using a realistic[2] sampling rate ξ are JPEG-compressed at QF75; 33 images with height or width smaller than 2560 are discarded.

Thumbnails of size 128×128 are generated from the decompressed covers using textbook implementations of nearest, bilinear, and bicubic interpolation, with and without AA, and with subsequent JPEG compression at QF75. The sampling rates are chosen s. t. $\xi \in [0.05, 0.5]$, with emphasis on the lower half. We add a real-world generator, ImageMagick. We deviate from the DCF standard and use square-shaped thumbnails, for comparability with related work in steganalysis.

The steganography is simulated at embedding rates $\alpha \in \{0.05, 0.10, \ldots, 0.40\}$, using J-UNIWARD, UERD, and nsF5, covering both adaptive and non-adaptive methods.

2) Experimental setup: The experiment has two parts, designed to provide insight into the impact of α and ξ on the thumbnail differences. The first part fixes the thumbnail configuration to the nearest kernel, $\xi = 0.25$, no AA, and evaluates β over different embedding methods and rates α, with and without the final JPEG compression. This thumbnail configuration was chosen as the simplest option to generate thumbnails. The second part focuses on nsF5, $\alpha = 0.2$, and evaluates β over different thumbnail configurations that vary the sampling rate ξ and the interpolation kernel, with and without AA. The embedding method was chosen as the most promising given the results of the first experiment.

[2]We assume an image resolution of 3–12 Mpx, which is 1600–4000 px along the longer dimension, and the DCF-compliant thumbnail size 160 × 120 px [28]. The sampling rate ranges between 0.04 and 0.10.

For a fair comparison, all thumbnails should have the same size regardless of the sampling rate. To maintain a constant thumbnail size of 128 × 128 while varying the sampling rate, we crop the pre-cover as shown in Fig. 1. The lowest sampling rate of $\xi = 0.05$ uses the full 2560 × 2560 pre-covers, while higher sampling rates require pre-cover cropping.

B. Results

Figure 2 shows the result of the first experiment. The top facet compares the thumbnails in spatial domain before the final thumbnail compression, while the bottom facet shows the change rate in the DCT domain. The change rate β between the thumbnails grows linearly with increasing α. The nsF5 method leaves more information in the thumbnail than the adaptive methods and the margin is even bigger after compression.

Note that β after compression is lower because the compression may suppress a portion of the changes, but also because embedding a single DCT coefficient may affect 8 × 8 spatial pixels. As a result, change rates calculated in spatial domain are on a higher scale.

The result of the second experiment is shown in Fig. 3. The two facets show the change rate in the DCT domain without (top) and with anti-aliasing (AA, bottom). Without AA, the change rate β stays approximately constant even for small thumbnails. All interpolation methods preserve a similar amount of changes. When AA is used, β decreases for lower thumbnail sizes. We conclude that AA removes information from the thumbnail. Real-world thumbnail generators, such as ImageMagick, produce similar change rates as textbook implementations without AA.

C. *Is it feasible to improve steganalysis with thumbnails ?*

Yes, in principle: even small thumbnails preserve a fair amount of information about the stego changes, at least without AA. A comparable amount of changes is present in the real-world generators, e.g., ImageMagick. The information in the thumbnail grows linearly with the message size.

V. Experimental Setup

We conduct two experiments to investigate how a cover thumbnail could aid current steganalysis methods.

The first experiment explores steganalysis with post-compression cover thumbnails. This experiment is motivated by the scenario where a cover thumbnail is found in the OS thumbnail cache. In this case, the straightforward approach is to re-compute the thumbnail from the main image and compare it to the given thumbnail. This experiment is denoted as *R*, as it assumes that the thumbnail generation is *reproducible*.

For steganalysis with pre-compression thumbnails, re-computing the thumbnail is not feasible because it would require access to the pre-cover. Therefore, the second experiment explores to what extent a learning-based detector can make use of the cover thumbnail as additional side information. We denote this experiment *I* (for *irreproducible*). The motivating example for this scenario is when the steganography software carelessly copies the cover metadata including a pre-compression thumbnail to the stego. A similar approach could be applicable for post-compression cover thumbnails with unknown or irreproducible pipeline too; but here we focus on pre-compression thumbnails.

A. Dataset

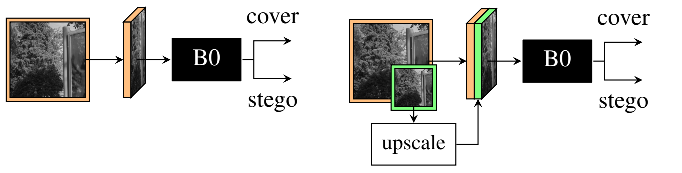
The experiments use a dataset consisting of 80 005 grayscale pre-covers of size 512×512 from the ALASKA2 dataset. Ten images with no content were dropped. To augment the dataset, the pre-covers are rotated by multiples of $90°$ prior to the compression at QF75. Stego objects are created from the covers by simulating J-UNIWARD (JUNI), UERD, and nsF5, at rates $\alpha \in \{0.1, 0.2, 0.4\}$. The thumbnails are generated from covers for the reproducible thumbnail scenario and from pre-covers for the irreproducible thumbnail scenario. The pre-cover thumbnails are irreproducible because the steganalyst does not have access to the pre-covers. We choose the configuration that resulted in the highest change rate in Sec. IV: nearest kernel, no AA, $\xi = 0.25$.

B. Detectors

The experiment *R* uses an analytical approach, i.e., without learning. The captured thumbnail is compared to a thumbnail reproduced from the main image. We follow the convention of a binary hypothesis test [27, Ch. 10], implying that any mismatch between the thumbnails is evidence for steganography.

The experiment *I* uses a convolutional neural network (CNN) based on the architecture EfficientNet B0 (abbr. B0), as shown in Fig. 4. All CNNs are initialized by pre-training on ImageNet. We use the original B0 architecture without any modifications from [29] because we favored a larger batch size over the increased memory requirement of the modified architecture.

The vanilla B0 shown in Fig. 4a serves as our baseline. The thumbnail should be injected into the network in a way to allow for positional comparison between the cover and the thumbnail. Our detector shown in Fig. 4b upsamples the thumbnail to the size of the image using the nearest kernel and



(a) *Baseline* detector, used as a reference.

(b) Side-informed detector; The upscaled pre-cover thumbnail (green) is appended along the channel axis with the image (orange).

Fig. 4: Detectors based on EfficientNet-B0 used in this study.

concatenates both signals together along the channel axis. The first (stem) layer of the side-informed detector has twice the number of input filters. The rest of the network is unmodified.

C. Training

The CNN is trained on 60% of the dataset using the random post-embedding horizontal and vertical flipping, and pre-embedding rotation mentioned in Sec. V-A to reduce overfitting. We deliberately avoided rotation as post-embedding augmentation to allow the CNN to capture directional traces. Training is performed at a constant learning rate of 10^{-4} with batch size 64 and dropout rate 25%. We use the cross-entropy loss and the AdamW optimizer. To achieve convergence for lower embedding rates, the CNN is trained using curriculum learning, by initializing the weights from the next higher embedding rate in the sequence $0.4 \to 0.2 \to 0.1$.

Another 20% of the dataset is used as a validation set. Training terminates when the validation loss has not improved for 8 consecutive epochs. The last 20% is used as test set to evaluate the performance without any test-time augmentation. We ensured that the corresponding cover and stego images are always in the same split.

D. Evaluation

Performance is reported using two metrics derived from an estimate of the receiver operating characteristic (ROC) curve. The ROC curve captures the dependency of type I and II errors, via the probability of false positive $\mathrm{P_{FP}}(\tau)$ and the probability of missed detection $\mathrm{P_{MD}}(\tau)$, on the decision threshold τ. The metrics are the probability of error, $\mathrm{P_E} = \min_\tau[\mathrm{P_{FP}}(\tau) + \mathrm{P_{MD}}(\tau)]/2$, and the missed detection rate at the false positive rate 5%, $\mathrm{P_{MD}^{5\%FP}} = \mathrm{P_{MD}}(\tau)|_{\tau \text{ s.t. } \mathrm{P_{FP}}(\tau)=5\%}$. While the $\mathrm{P_E}$ measures the performance at the best empirical τ, $\mathrm{P_{MD}^{5\%FP}}$ corresponds to the performance at a practical FP rate [30].

VI. Results

The results for the experiments *R* and *I* are presented below. Experiment *I* is followed by a post-hoc study, which evaluates the CNN's sensitivity to various thumbnail configurations.

A. Experiment with reproducible thumbnail

Table I shows the results of experiment *R*. The metric $P_{MD}^{5\%FP}$ is replaced with the absolute number of missed detections because there are no false positives. In this case, a missed detection means that the stego thumbnail is identical to the cover thumbnail. This approach achieves nearly perfect detection even for low embedding rates. The detector is slightly better for nsF5, compared to UERD and J-UNIWARD, because nsF5 leaves more traces in the thumbnail, as demonstrated by the feasibility study in Sec. IV.

TABLE I: Performance of the detector using leaked post-compression cover thumbnails (nearest, $\xi = 0.25$) and assuming a reproducible thumbnail generation pipeline.

Method	α	Missed	P_E
JUNI	0.4	31 / 79995	0.0002
JUNI	0.2	41 / 79995	0.0003
JUNI	0.1	60 / 79995	0.0004
nsF5	0.4	25 / 79995	0.0002
nsF5	0.2	34 / 79995	0.0002
nsF5	0.1	53 / 79995	0.0003
UERD	0.4	28 / 79995	0.0002
UERD	0.2	43 / 79995	0.0003
UERD	0.1	62 / 79995	0.0004

B. Experiment with irreproducible thumbnail

The results of experiment *I* are reported in Table II. The side-informed detector consistently outperforms the baseline across different embedding methods and rates, however at a rather small margin. For J-UNIWARD the decrease in error probability ΔP_E is in the range of 1.3–2.1 %-pts. For UERD it grows from about 0.5 %-pts for $\alpha = 0.4$ to almost 3 %-pts for $\alpha = 0.1$. For nsF5 it fluctuates for different rates between 0.25 and 1.5 %-pts.

Sensitivity to other thumbnail configurations: The CNN trained with one thumbnail generation pipeline is now examined for generalization to other pipelines. The performance decrease is compared to the change rate β from Sec. IV.

Anti-aliasing, the strongest factor for β, causes only a minor decrease within 1%-pt. The interpolation kernel, with a minor

TABLE II: Performance increase of B0 detector side-informed with pre-cover thumbnails (nearest, $\xi = 0.25$).

		Baseline		Thumbnail		
Method	α	P_E	$P_{MD}^{5\%FP}$	P_E	$P_{MD}^{5\%FP}$	ΔP_E
JUNI	0.4	0.0989	0.1662	0.0813	0.1262	−0.0176
JUNI	0.2	0.2232	0.5013	0.2098	0.4616	−0.0133
JUNI	0.1	0.3511	0.7600	0.3301	0.7295	−0.0210
nsF5	0.4	0.0177	0.0051	0.0138	0.0031	−0.0039
nsF5	0.2	0.1170	0.2316	0.1031	0.1898	−0.0139
nsF5	0.1	0.2735	0.6413	0.2706	0.6465	−0.0029
UERD	0.4	0.0739	0.1012	0.0691	0.0899	−0.0048
UERD	0.2	0.1622	0.3090	0.1515	0.2907	−0.0107
UERD	0.1	0.2811	0.5746	0.2518	0.5336	−0.0293

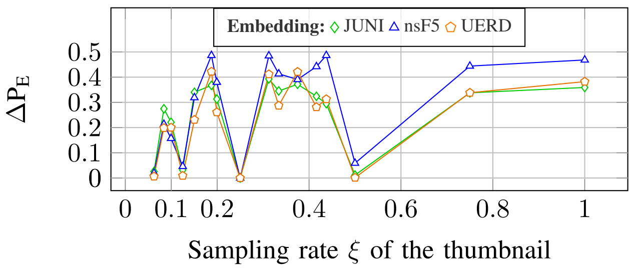


Fig. 5: Effect of changed sampling rate ξ of the thumbnail on the detection error P_E. The CNN was trained with $\xi = 0.25$.

difference in β, has almost no effect on the performance. After training on pre-compressed cover thumbnails, the CNN shows a minor performance drop when tested with post-compressed cover thumbnails. The sampling rate ξ, despite having little impact on β, has a profound impact on the detection performance, as shown by Figure 5. Unless ξ is a power of $\frac{1}{2}$, the accuracy degrades below the baseline. We hypothesize that this degradation is because upscaling the thumbnail with an arbitrary ξ deviates from the JPEG grid seen during training.

Alas, the strong impact of ξ means that steganalysts in practice may need to retrain for a specific cover and thumbnail size of the files under investigation.

VII. Discussion

Thumbnail reproduction is a powerful attack useful in practice. However, it can only be applied to post-compression cover thumbnails with a known and reproducible thumbnail pipeline. Even then there are pitfalls, e.g., thumbnail pipelines of thumbnail caches differ between Windows versions [23]. Future work could investigate whether a leaked cover thumbnail allows estimating the embedding rate in the main image.

The improvement in experiment *I* over the baseline is small but consistent. The CNN uses the information from the thumbnail channel, but does not seem to perform a local comparison with the main image. A limitation of our study is that the detector is only trained on one specific thumbnail generation pipeline. However, the thumbnail generation can vary between camera models [22]. We leave a further research towards generalization to future work.

An alternative to our CNN could be training a siamese network [31] on image-thumbnail pairs. Our initial experiments with siamese networks did not lead to a confident improvement. Another idea was to replace the upscaled thumbnail by its difference with the image. The difference image could be also used as an attention mechanism. Systematic explorations of these directions are future work.

The experiments were limited to grayscale images, which was chosen to prevent color components from contributing additional side information, and to a single quality factor. A general limitation is that capturing a leaked cover thumbnail is a rather strong assumption.

VIII. Conclusion

This paper investigates to what extent a leaked cover thumbnail aids steganalysis. Cover thumbnails can be divided into pre-compression and post-compression thumbnails based on their generation steps. For post-compression cover thumbnails, we quantify the information left and demonstrate that reproducing the thumbnail leads to near-perfect detection of steganography. For pre-compression cover thumbnails, we demonstrate how an EfficientNet detector can benefit from this additional side information.

Acknowledgment

We thank Patrick De Smet (NICC, Belgium) for pointing us to thumbnails as a potential source of side information. Moreover, we thank Verena Lachner, Nora Hofer, and Alexander Schlögl for their valuable support and feedback. 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). The code is available at github.com/uibk-uncover/steganalysis-using-thumbnails.

References

[1] L. Guo, J. Ni, W. Su, C. Tang, and Y. Shi, "Using Statistical Image Model for JPEG Steganography: Uniform Embedding Revisited," *IEEE TIFS*, vol. 10, no. 12, pp. 2669–2680, 2015.

[2] V. Holub, J. Fridrich, and T. Denemark, "Universal Distortion Function for Steganography in an Arbitrary Domain," *EURASIP JINS*, vol. 2014, no. 1, pp. 1–13, 2014.

[3] J. Fridrich, T. Pevný, and J. Kodovský, "Statistically Undetectable JPEG Steganography: Dead Ends Challenges, and Opportunities," in *MMSEC*. ACM, 2007, pp. 3–14.

[4] H. Farid, "Detecting Steganographic Messages in Digital Images," Dartmouth College, Tech. Rep. TR2001-412, 2001.

[5] R. Cogranne, Q. Giboulot, and P. Bas, "ALASKA#2: Challenging Academic Research on Steganalysis with Realistic Images," in *WIFS*. IEEE, 2020, pp. 1–5.

[6] M. Goljan and J. Fridrich, "CFA-Aware Features for Steganalysis of Color Images," in *MWSF*, vol. 9409. SPIE, 2015, pp. 279–291.

[7] M. Kirchner and R. Böhme, "'Steganalysis in Technicolor' – Boosting WS Detection of Stego Images from CFA-Interpolated Covers," in *ICASSP*. IEEE, May 2014, pp. 3982–3986.

[8] W. Tang, H. Li, W. Luo, and J. Huang, "Adaptive Steganalysis Against WOW Embedding Algorithm," in *IH&MMSEC*. ACM, 2014, pp. 91–96.

[9] M. Boroumand, M. Chen, and J. Fridrich, "Deep Residual Network for Steganalysis of Digital Images," *IEEE TIFS*, vol. 14, no. 5, pp. 1181–1193, 2018.

[10] J. Ye, J. Ni, and Y. Yi, "Deep Learning Hierarchical Representations for Image Steganalysis," *IEEE TIFS*, vol. 12, no. 11, pp. 2545–2557, 2017.

[11] S. McKeown, G. Russell, and P. Leimich, "Fast Forensic Triage Using Centralised Thumbnail Caches on Windows Operating Systems," *ADFSL JDFSL*, vol. 14, no. 3, 2020.

[12] I. Kuksov, "Do Your Online Photos Respect Your Privacy?" 2023, accessed on 22 May 2023. [Online]. Available: https://www.kaspersky.co.uk/blog/exif-privacy/7893/

[13] L. Newman, "Some Photo-Cropping Apps Are Exposing Your Secrets," 2023, accessed on 17 May 2023. [Online]. Available: https://www.wired.com/story/acropalyse-google-markup-windows-photo-cropping-bug/

[14] T. Denemark, V. Sedighi, V. Holub, R. Cogranne, and J. Fridrich, "Selection-Channel-Aware Rich Model for Steganalysis of Digital Images," in *WIFS*. IEEE, 2014, pp. 48–53.

[15] T. Denemark, M. Boroumand, and J. Fridrich, "Steganalysis Features for Content-Adaptive JPEG Steganography," *IEEE TIFS*, vol. 11, no. 8, pp. 1736–1746, 2016.

[16] Q. Li, G. Feng, Y. Ren, and X. Zhang, "Embedding Probability Guided Network for Image Steganalysis," *IEEE Signal Processing Letters*, vol. 28, pp. 1095–1099, 2021.

[17] V. Holub and J. Fridrich, "Phase-Aware Projection Model for Steganalysis of JPEG Images," in *MWSF*, vol. 9409. SPIE, 2015, pp. 259–269.

[18] X. Song, F. Liu, C. Yang, X. Luo, and Y. Zhang, "Steganalysis of Adaptive JPEG Steganography Using 2D Gabor Filters," in *IH&MMSEC*, 2015, pp. 15–23.

[19] M. Chen, V. Sedighi, M. Boroumand, and J. Fridrich, "JPEG-Phase-Aware Convolutional Neural Network for Steganalysis of JPEG Images," in *IH&MMSEC*, 2017, pp. 75–84.

[20] S. Murdoch and M. Dornseif, "Hidden Data in Internet Published Documents," 21st Chaos Communication Congress, 2004.

[21] K. Cohen, "Digital Still Camera Forensics," *SSDDF*, vol. 1, no. 1, pp. 1–8, 2007.

[22] E. Kee and H. Farid, "Digital Image Authentication from Thumbnails," in *EI: Media Forensics and Security II*, vol. 7541. San Jose, CA, USA: SPIE-IS&T, 2010, pp. 139–148.

[23] S. Morris and H. Chivers, "An Analysis of the Structure and Behaviour of the Windows 7 Operating System Thumbnail Cache," in *EAI ICDF2C*. University of Strathclyde, Glasgow, 2011.

[24] R. Böhme, *Advanced Statistical Steganalysis*. Springer, 2010.

[25] R. Gonzales and P. Wintz, *Digital Image Processing*. Addison-Wesley Longman Publishing Co., Inc., 1987.

[26] C. Pasquini and R. Böhme, "Information-Theoretic Bounds for the Forensic Detection of Downscaled Signals," *IEEE TIFS*, vol. 14, no. 7, pp. 1928–1943, 2018.

[27] J. Fridrich, *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge University Press, 2009.

[28] Japan Electronics and Information Technology Industries Association, "Design Rule for Camera File system: DCF Version 2.0," Camera & Imaging Products Association, Tokyo, Japan, Standard, Apr. 2010.

[29] Y. Yousfi, J. Butora, J. Fridrich, and C. Fuji Tsang, "Improving EfficientNet for JPEG Steganalysis," in *IH&MMSEC*. ACM, 2021, pp. 149–157.

[30] R. Cogranne, Q. Giboulot, and P. Bas, "The ALASKA Steganalysis Challenge: A First Step Towards Steganalysis," in *IH&MMSEC*, 2019, pp. 125–137.

[31] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a Similarity Metric Discriminatively, with Application to Face Verification," in *CVPR*, vol. 1. IEEE, 2005, pp. 539–546.

Appendix A
JPEG Thumbnail

The JPEG thumbnail is carried inside Exchangeable Image File Format (EXIF) metadata, and as such is limited by the EXIF format. JPEG thumbnail parameters are further defined in Design rule for Camera File system (DCF) standard [28] issued by the Camera & Image Products Association (CIPA). The DCF standard is a guideline for compatibility, and is not obligatory. Table III shows a comparison of EXIF and DCF.

TABLE III: Parameters of JPEG thumbnails defined by the EXIF format and by the DCF standard.

	EXIF	**DCF**
Format	JPEG/TIFF/HEIC	JPEG
Size	Up to 64 kB	Not specified
Resolution	Any	160×120
Aspect ratio	Any	4:3
Chroma sampling	Any	4:2:2
Color space	Any	sRGB
No. thumbnails	Multiple	At most 1