Breaking ALASKA: Color Separation for Steganalysis in JPEG Domain

Yassine Yousfi, Jan Butora, Jessica Fridrich
Binghamton University
Department of Electrical and Computer Engineering
Binghamton, NY 13902-6000
yyousfi1,jbutora1,fridrich@binghamton.edu

Quentin Giboulot
Troyes University of Technology
Laboratory for System Modelling and Dependability, ICD, UMR 6281 CNRS
Troyes, France
quentin.giboulot@utt.fr

ABSTRACT

This paper describes the architecture and training of detectors developed for the ALASKA steganalysis challenge. For each quality factor in the range 60–98, several multi-class tile detectors implemented as SRNets were trained on various combinations of three input channels: luminance and two chrominance channels. To accept images of arbitrary size, the detector for each quality factor was a multi-class multi-layered perceptron trained on features extracted by the tile detectors. For quality 99 and 100, a new "reverse JPEG compatibility attack" was developed and also implemented using the SRNet via the tile detector. Throughout the paper, we explain various improvements we discovered during the course of the competition and discuss the challenges we encountered and trade offs that had to be adopted in order to build a detector capable of detecting steganographic content in a stego source of great diversity.

CCS CONCEPTS

• **Security and privacy → Cryptanalysis and other attacks**.

KEYWORDS

Steganography, steganalysis, JPEG, deep learning, ALASKA competition, color

ACM Reference Format:
Yassine Yousfi, Jan Butora, Jessica Fridrich and Quentin Giboulot. 2019. Breaking ALASKA: Color Separation for Steganalysis in JPEG Domain. In *ACM Information Hiding and Multimedia Security Workshop (IH&MMSec '19), July 3–5, 2019, Paris, France.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Steganography is the art of covert communication when secrets are hidden in ordinary looking cover objects. The goal is to make steganographic communication indistinguishable from regular exchange of information during which no secrets are passed between communicating parties. Digital media, such as images, are particularly suitable cover objects because of their ubiquity and because they can be slightly modified without changing their appearance, potentially thus able to hold large messages. The task of detecting the presence of embedding changes is complicated by the fact that images contain an indeterministic component, the acquisition noise, and by the immense diversity and complexity introduced during acquisition, development from the RAW capture, post-processing, editing, and sharing. When designing steganalysis detectors, researchers thus usually consider a rather sand-boxed environment : a known steganographic scheme, known payload, and a known cover source typically consisting of grayscale images of a fixed size.

The purpose of the ALASKA competition was to have researchers face more realistic conditions that are closer to what a steganalyst might have to deal with in real life. In this paper, we only mention those aspects of the competition that are relevant for the material presented here while referring the reader to [8] for a more detailed description of the competition setup and interpretation of the final results. The participants were given a set of 5,000 JPEG images, some of which were cover images and some embedded with secrets. We will call this set 'ALASKArank' because the detection results achieved on this set determined the ranking of the competing teams. Four JPEG steganographic schemes were used to produce the stego images: J-UNIWARD [16], UED-JC [14], EBS [23], and nsF5 [11] with priors 0.4, 0.3, 0.15, and 0.15, respectively, according to the embedding script shared by the organizers. All four embedding methods were adjusted to hide in color JPEG files by embedding in chrominance channels a fraction of the payload determined by the JPEG quality factor (see Section 2.3 in [8]). The size of the embedded payload was determined by the cover image development history (starting with a RAW sensor capture), which was again randomized. It involved four different choices for demosaicking, resizing by a randomly selected factor in the range [0.6, 1.3], a version of source-preserving cropping called the 'smart crop' [12] to $A \times B$ pixels with $A, B \in \{512, 640, 720, 1024\}$, sharpening, denoising, and micro-contrast enhancement whose parameters were again randomized, and final JPEG compression with quality factor between 60 and 100 selected at random according to a prior that the organizers computed by analyzing a large number of JPEG images uploaded to the image sharing portal Flickr. The payload w.r.t. image size was scaled according to the square root law [19] to obtain an approximately constant statistical detectability across different sizes. We note that the smallest and largest sizes were 512×512 and 1024×1024, respectively.

The embedding code for all four steganographic schemes was given to the participants as was the script for developing a RAW image to a JPEG cover. This allowed the participants to generate their training sets without worrying about the cover source mismatch, at least up to possible differences in the source of RAW images.

The organizers claimed that ALASKArank did not include stego images created with other embedding algorithms. Thus, the competition followed what is recognized as the "closed-set problem."

IH&MMSec '19, July 3–5, 2019, Paris, France
2019. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

The information that was *not* revealed to the competitors included:

(1) The percentage of stego images in ALASKArank and per each quality factor.
(2) The priors for all four stego schemes per each quality factor, possibly thus introducing an unknown stego-source mismatch.
(3) The source of RAW images for ALASKArank, possibly thus creating a cover-source mismatch.

The competitors were permitted one submission per four hours per team. The submission was a text file with file names from ALASKArank ordered from the most likely stego to the least likely stego. This allowed the organizers to draw an ROC curve and report three quantities on ALASKA leaderboard: the missed detection rate at 5% false alarm, MD5, the minimum average total error under equal priors, P_{E}, and the false-alarm rate for 50% detection, FA50:

$$\mathrm{MD5} = P_{\mathrm{MD}}(P_{\mathrm{FA}} = 0.05) \tag{1}$$

$$P_{\mathrm{E}} = \min_{P_{\mathrm{FA}}} \frac{1}{2}\left(P_{\mathrm{FA}} + P_{\mathrm{MD}}(P_{\mathrm{FA}})\right) \tag{2}$$

$$\mathrm{FA50} = P_{\mathrm{FA}}(P_{\mathrm{MD}} = 0.5) \tag{3}$$

The quantity MD5 was used for the final ranking.

In the next section, we describe the detector we built for the competition. Due to limited resources and time, this detector was built only for the most populous quality factors in ALASKArank different from 99 and 100 since for these two quality factors, we developed a new "reverse JPEG compatibility attack" with much larger detection accuracy than conventional approaches. Section 3 contains the results of all investigations conducted during the competition that motivated our approach and the effect of various choices on the performance of our detectors. In Section 5, we analyze false alarms of our detectors across JPEG quality factors, sensors, and embedding algorithms. The paper is concluded in Section 5.

2 DETECTORS AND THEIR TRAINING

The final structure of our detector described in this paper was necessarily affected by the available resources and limited time. The competition required us to address a spectrum of diverse challenges that each ideally should be investigated in a separate paper: steganalyzing images of arbitrary size, steganalysis of color JPEGs, detection in diversified stego source, variable payload, and a wide spectrum of quality factors.

Since the beginning of the competition, we committed to the strategy to build a detector for each quality factor (QF) as it is unlikely that a single detector, whether built as a neural network or with rich models, would provide the best performance. It remains to be seen whether this strategy is scalable in the real world because many digital cameras as well as editing software use customized quantization matrices. The obvious remedy here would be to steganalyze images with non-standard tables with the detector trained for the closest quantization table in some suitable metric. We stress that in our quest, we did not address this issue and fully focused on building detectors for each quality factor that occurred in ALASKArank.

The detectors for QFs 60–98 were built as multi-layered multi-class perceptrons (MLPs) trained on features in the form of four moments of 512 feature maps from up to five different SRNets [4] trained on various combinations of the three channels that comprise color JPEG images: luminance Y and chrominances C_r, C_b. Due to the limited memory of our GPUs (11–12GB), in order to use a reasonable size minibatch these network detectors were first trained on small 256×256 tiles. The front part (before the fully-connected segment of the network) of these tile detectors was used as a "feature extractor" to convert an input image of arbitrary size to 4×512 moments on which a multi-class MLP was trained for the final detector.

For quality factors 99 and 100, we discovered a new attack, which we call the reverse JPEG compatibility attack. In a nut shell, we basically trained SRNets on rounding errors when decompressing an image to the spatial domain. The remarkable accuracy of these detectors is fundamentally due to the fact that the block discrete cosine transform (DCT) applied during JPEG compression is applied to an integer-valued signal.

2.1 Detector architecture

All detectors were built around the same deep residual neural network called SRNet [4]. This detector was developed in-house and we had the most experience with it. Also, based on the comparisons with competing architectures [24, 25] reported in [4], at the time of publishing this work SRNet achieved the best overall results for steganalysis in the JPEG domain. Moreover, this network is rather large, it contains 4, 781, 157 learnable parameters, which we felt might be important when detecting steganography in such greatly diversified cover and stego sources. We note that the selection-chanel-aware version of SRNet could not be used because the stego source contained images embedded with four different methods, some of which were adaptive to content (J-UNIWARD, UED, and EBS), while others (nsF5) were non-adaptive.

The SRNet uses residual skip connections with 3×3 filters. All convolutional layers use batch normalization and ReLU activation. The first eight convolutional blocks are unpooled because average pooling can be seen as a low-pass filter, whereas steganalysis is mostly interested in high frequency content where the stego signal resides. The first eight layers can thus be loosely viewed as noise residual extractors. The next convolutional blocks are pooled using a 3×3 averaging layer with stride 2, as well as strided 1×1 convolutions in the skip connections. The SRNet applies global average pooling in the last pooled layer to 512 feature maps. In the original SRNet, this 512-dimensional "feature vector" of global feature map averages is fed into a fully-connected (FC) layer with two outputs when training a binary classifier.

To be used for steganalysis of JPEG images, the SRNet inputs are JPEG images decompressed to the spatial domain without rounding to integers or clipping. For color steganalysis (or multi-channel inputs in general), the SRNet was modified by changing the 3×3 kernels in the first layer to $c \times 3 \times 3$ kernels, where c is the number of input channels, without any other modifications to its architecture.

2.2 Training dataset

A total of 50,000 full size RAW images made available by the ALASKA organizers were used to prepare our training sets for each quality factor, which required modifying the developing script to compress using a desired quality factor instead of randomly sampling according to the Flickr distribution. The developing script supplied by the organizers as well as the embedding script were used to generate the training set of 50,000 cover images and 50,000 stego images for each embedding method (thus, the training set contained 5×50,000 images). All JPEG images were obtained using Python's PIL library.

Since the SRNet requires images of size 256×256 to fit a reasonable size minibatch into the memory of our GPUs (12GB), we first created 50,000 cover "tiles" all of size 256×256. This also required modifying the developing script to always select a smart crop of this size. The embedding script was then used to create two sets of 4×50,000 stego images: 'TILEbase' and 'TILEdouble' with stego images embedded with payload scaled to the smaller size as prescribed in the embedding script and with the same script embedding double this payload, respectively. First, the SRNet was trained on TILEdouble and then used as a seed for training on TILEbase. This had to be done because training directly on the base payload may not always converge or produce the best detector. This is especially true for large quality factors that appeared to be harder to steganalyze due to the specific payload scaling. The detector trained on 256×256 tiles from TILEbase will be referred to as the "tile detector."

Similarly, we created a database of arbitrary sized images 'ARBITRARYbase' used to train the arbitrary size detector as described in Section 2.4.

The training set (TRN), validation set (VAL), and test set (TST) contained respectively 42,500, 3,500, and 3,500 cover images (around 500 cover images were not used because they were corrupted or failed the processing pipeline). The TRN, VAL, and TST sets were created for each quality factor and each stego scheme in TILEdouble, TILEbase, and ARBITRARYbase. The TST set was used solely to produce all experimental results for this paper and was not used for building the detectors.

For internal development purposes, we replicated the ALASKArank set locally by selecting 3,500 JPEG images from the TRN JPEGs made available at the Alaska website developed, processed, and embedded by the organizers. We believed that forming this "replica" of ALASKA rank would give us a set with similar properties in terms of the mixture of quality factors, sizes, and stego images. We will refer to this set as 'mixTST.' Based on evaluating the outputs of our detectors on ALASKArank (especially the detectors for quality factors 99 and 100 see Section 3.9.1), it appeared that it contained only 10% stego images and 90% cover images. Thus, when forming mixTST, we selected 350 stego images and 3,150 covers.

2.3 Detector form

The form of the detector used for the ALASKA competition was inspired by the results reported in [5], where the authors investigated steganalysis of multiple stego algorithms in the spatial domain using three different strategies with the SRNet: binary detector (cover vs. all stego), multi-class detector, and the so-called bucket detector, where K binary detectors are trained to distinguish between covers and a specific stego method (out of K methods), then their last activations before the FC layer are merged into a MLP trained as a binary or a multi-class detector. It was shown experimentally that the best strategy in terms of accuracy of classifying a cover image as cover and any stego image as stego was the multi-class SRNet with the bucket detector performing by far the worst. After an initial study on QF 75 (see Section 3), we selected the multi-class detector for the ALASKA challenge as well.

Denoting the training set of cover images as $\mathcal{S}_0$ and the sets of images embedded with stego algorithm $k \in \{1, 2, 3, 4\}$ as $\mathcal{S}_k$, each minibatch of $2N_B$ images $\mathcal{B}$ was formed by randomly selecting N_B cover images and pairing each cover $x \in \mathcal{B}$ with the corresponding stego image $y \in \mathcal{S}_k$, where the stego class $k \in \{1, 2, 3, 4\}$ was selected with the stego class priors mentioned in Section 1. This multi-class detector uses five soft-max output neurons (with soft outputs $q_k(x)$, $k = 0, \ldots, 4$) and minimizes the multi-class cross-entropy loss function[1]

$$L(\mathcal{B}) = -\frac{1}{|\mathcal{B}|} \sum_{x \in \mathcal{B}} \sum_{k=0}^{K} p_k(x) \log q_k(x), \tag{4}$$

where $p_k(x) = 1$ when $x \in \mathcal{S}_k$, $k \in \{0, \ldots, 4\}$ and $p_k(x) = 0$ otherwise. The cover-stego pair constraint is important when iteratively training detectors for steganalysis because it helps find the gradients separating the classes. Using Tensorflow's Estimators API [7] together with the Datasets API [2] allowed us to implement the SRNet in a cleaner and more efficient fashion with a minibatch size $N_B = 32$, which is twice as big as what was used in [4]. A larger minibatch is highly beneficial when training on diversified stego sources so that the optimizer sees more stego images from each embedding method in each minibatch.

During training, data augmentation was also applied to the batch using flips and rotations. Note that the random selection of the embedding scheme can also be viewed as data augmentation – one cover image x may be paired up with the corresponding stego image y embedded with any of the four embedding schemes through the epochs.

2.4 Arbitrary size detector

The tile detector explained above accepts small tiles on its input. The input to the FC layer of the SRNet, however, is independent of the input image size because it is a 512-dimensional vector of global means of all 512 feature maps. Technically it is thus possible to use this "feature vector" extracted by the tile detector and only retrain a simple non-linear classifier, such as a MLP, on features extracted from ARBITRARYbase images. Following the work of Tsang et al. [12] on steganalyzing images of arbitrary size, we extracted additional three moments from the 512 feature maps – the variance, minimum, and maximum since order statistics supply additional information about the original image resolution.

The arbitrary size detector was trained on ARBITRARYbase using the same TRN/VAL split.[2]

We experimented with MLPs with one and two hidden layers each with double the dimensionality of the input feature vector and ReLU activations. For example, when using features from a single SRNet, 4×512 moments are fed into the MLP with two hidden layers each with 8×512 neurons. Based on experiments (Section 3) two hidden layers provide better performance than a single hidden

[1]When $K = 2$, the detector is a simple binary detector (stego/cover).

[2]Note that training on the same split can be done here without the risk of over training because the ARBITRARYbase image properties are very different from the TILEbase. Keeping both training sets disjoint necessitates a smaller training set and did not lead to any noticeable generalization improvement.

layer, and training the MLP as multi-class again provides better performance than training it as a binary cover vs. all stego classifier.

2.5 Color separation

The most straightforward and arguably the simplest way to extend the SRNet to accept images with more than one channel (c channels) is to replace the 3×3 filters in the first convolutional layer with $c \times 3 \times 3$ filters. The rest of the architecture can be kept unchanged. While there certainly exist other options, such as keeping the three channels separated up to a certain depth within the architecture [26] and only then allowing them to merge, we felt that there simply was not enough time to properly research alternative architectures since many other challenges had to be addressed.

Early in the competition, we built all our tile detectors as three-channel SRNets ($YCrCb$-SRNet) trained on color JPEG images represented as three channels: decompressed luminance Y, decompressed chrominance C_r, and decompressed chrominance C_b without any rounding or clipping. Later on, we discovered that training additional SRNet tile detectors only on luminance and only on chrominance and merging their "feature vectors" provided a significant boost. This may be due to the way we introduce color to SRNet – the three channels are merged on the input to the second convolutional layer. We hypothesize that when supplying all three channels, it is possible that the SRNet focuses on leveraging embedding inconsistencies between the luminance and the two chrominances, ignoring possibly useful but perhaps comparatively weaker signals that exist within each channel that are left "untapped" when training the YC_rC_b-SRNet. Training the SRNet only on one chrominance may force the network to leverage different types of embedding artifacts. Extending this idea of "color separation" even further, we trained (on the most populous QFs in ALASKArank) five versions of tile detectors: YC_rC_b-SRNet, Y-SRNet, C_rC_b-SRNet, C_r-SRNet, and C_b-SRNet. When used as feature extractors for training the detector (multi-class MLP) for arbitrary image size, their concatenated feature vectors had the dimensionality $5 \times 4 \times 512$ (five networks, four moments, 512 feature maps from each).

We fully acknowledge that addressing color in this fashion is likely suboptimal, and also perhaps cumbersome, and that a single alternative architecture with the colors kept separate to a certain depth may be able to achieve the same performance. This is postponed to future research.

2.6 Quality factors 99 and 100

These two quality factors were treated separately because during the course of the competition, we discovered a new, extremely reliable "compatibility attack" on JPEG steganography applicably only to these two largest quality factors. Since the authors are currently preparing a separate journal manuscript detailing this attack, in this paper, we mention only briefly the main idea for JPEG quality 100.

Let us assume that the DCT is applied to an integer-valued signal x_{ij}, such as luminance or chrominance. After the transform, the DCT coefficients c_{ij} are rounded to integers $d_{ij} = [c_{ij}]$. Modeling the rounding error in the DCT domain, $c_{ij} - [c_{ij}]$, as a random variable uniformly distributed on the interval $(-1/2, 1/2]$, due to the orthonormality of the inverse DCT, the difference between the original uncompressed pixel values x_{ij} and the same pixel value z_{ij} in the decompressed JPEG image follows a Gaussian distribution (due to central limit theorem) with variance $s = 1/12$, $z_{ij} \sim \mathcal{N}(0, s)$, the variance of the rounding error in the DCT domain. Even though the uncompressed pixel value x_{ij} is not available to the detector, the rounding error $e_{ij} = z_{ij} - [z_{ij}]$ follows $\mathcal{N}(0, s)$ "folded" to the interval $[-1/2, 1/2]$:

$$\nu(x; s) = \frac{1}{\sqrt{2\pi s}} \sum_{n \in \mathbb{Z}} \exp\left(-\frac{(x+n)^2}{2s}\right). \quad (5)$$

If the DCT coefficients d_{ij} are subjected to steganographic embedding changes, the combined "noise" due to rounding and embedding will translate to a larger noise variance $s' > s$ in the JPEG domain and thus a larger variance of pixels in the (non-rounded and non-clipped) decompressed pixels from the stego image. What makes the attack work really well is the fact that the folded Gaussian distribution is very sensitive to the variance s and rather quickly converges to a uniform distribution as s increases. Figure 1 shows the folded Gaussian distribution (5) for various values of the variance s.

While a scalar statistic in the form of the variance of the rounding errors of the decompressed image can achieve a respectable performance for quality 100, an even better performance, especially for quality 99, can be achieved when simply training an SRNet on the rounding errors e_{ij}. We experimentally determined that training only on rounding errors of luminance gave in fact slightly better results than when training a three-channel SRNet on rounding errors of luminance and both chrominance channels. The detectors were also built by first training a tile detector on 256×256 tiles, and then an inner-product (IP) layer was retrained on 512 global means extracted by the front part of the tile detector for images of arbitrary size, similar to the procedure outlined above. We note that replacing the FC layer with a MLP with hidden layers did not lead to any performance improvement, and neither did adding other moments than means.

The detectors trained on rounding errors achieved detection accuracy of 94% and 99% on our TST sets where the stego classes were represented with the priors mentioned in Section 1. The detection accuracy on individual stego schemes for J-UNIWARD, nsF5, EBS, and UED were: 0.9985, 0.7945, 0.9810, and 0.9885. The false alarm rate for this detector is 0.0007.

When training the detector as multi-class, on 773 QF 100 images from ALASKArank we detected 701 covers and 27, 9, 11, and 25 stego images from J-UNIWARD, nsF5, EBS, and UED, respectively, which approximately corresponds to the priors of all four embedding schemes.

2.7 Ordering

ALASKArank contains images with a wide range of JPEG quality factors. When training a separate detector for each quality factor, we had to sort the images for a submission file, which required merging the outputs from all detectors. While it seems natural to use the soft outputs for this task, it is important to realize that, despite the fact that soft-outputs are non-negative and sum to one, they are often incorrectly called "probabilities." This is because they usually lack an important property of a probability estimate: being

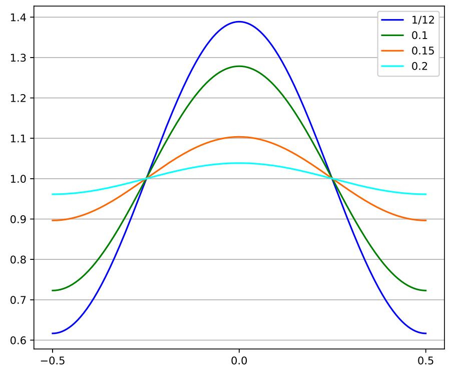

Figure 1: Folded Gaussian distribution $v(x; s)$ for noise variance in the DCT domain $s = 1/12, 0.1, 0.15, 0.2$. Note how rapidly $v(x; s)$ converges to a uniform distribution with increased s.

a representative of the true correctness likelihood. This property is often referred to as "calibration" in the statistical and machine learning community. Calibration is important for the ALASKA challenge when sorting the images from ALASKArank as the test statistics from all network detectors should represent comparable confidence levels.

Calibration is usually visualized using confidence plots (also called calibration plots) where the expected fraction of positives (stego) is plotted as a function of the soft outputs. To be truly representative of the correctness likelihood, confidence plots should be approximately diagonal, i. e., a soft output of 0.8 should reflect 80% in expectation of samples belonging to the positive (stego) class..

In practice, the expected fraction of positives is estimated by binning the outputs into M intervals of the same size and calculating the fraction of positives within each bin. As shown experimentally in [13], soft outputs from deep neural architectures are not well calibrated. The authors suggest using a plug-in post processing technique called temperature scaling to correct this mis-calibration.

It is also interesting to point out that the deeper an architecture is, the less calibrated the output is likely to be. This is coherent with the fact that logistic regression (also seen as a single-layer MLP) is one of the best classifiers in terms of calibration [1].

In our case, the final detector was a MLP trained as multi-class. Thus, for an input image x it outputs five numbers that add to 1: $q_k(x), k \in \{0, \dots, 4\}$ with $q_0(x)$ associated with the cover class. We experimented with several ways to convert these five soft outputs to a scalar for ordering ALASKArank. The simplest is to order according to $1 - p_0(x) = \sum_{k=1}^{4} p_k(x)$. Being an output from a MLP with only two hidden layers, its output was already approximately calibrated. Thus, there was no need to calibrate our detectors.

Figure 2 shows the confidence plot for quality factor 95, highlighting the difference between the soft-outputs of a YC_rC_b-SRNet tile detector and a single-hidden-layer MLP (the arbitrary size detector for QF 95) with a single soft output $1 - p_0(x)$. This shows that

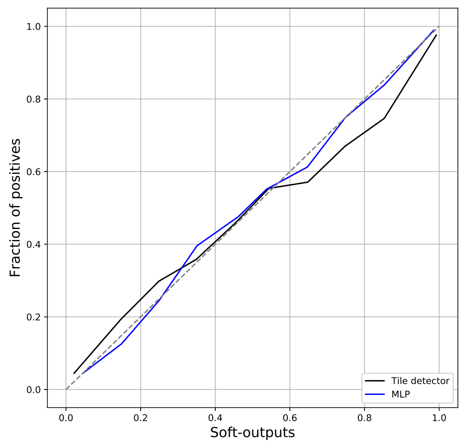

Figure 2: Calibration plot for the tile detector and the arbitrary size detector for JPEG quality 95.

Table 1: Detection performance for YC_rC_b-SRNet trained as binary and multi-class for QF 75 on TILEbase.

	Binary	Multi-class
P_E	8.10	7.13
MD5	11.41	9.60

the use of a simple MLP for arbitrary size images helps improve calibration. This trend was observed across all quality factors.

3 EXPERIMENTS

In this section, we report the results of multiple experiments whose purpose is to justify the detector architecture explained above. In reality, since the final architecture emerged slowly over six months, our experiments may appear somewhat "spotty," which is an unfortunate consequence of having to submit this paper right after the competition end . Nevertheless, the results do provide useful insight into what motivated our choices and reveal numerous interesting lessons-learned that are likely to spur additional research.

All experiments were performed using four NVIDIA Titan, four NVIDIA Titan X, four NVIDIA Titan Xp, and three NVIDIA GeForce RTX 2080Ti GPUs. We report the results in terms of MD5 (the ALASKA score) as well as P_E as we noticed that sometimes detectors with approximately the same P_E may exhibit vastly different MD5.

3.1 Detector form

Our initial study was performed for the quality factor 75. The purpose was to determine the best form of the detector. In particular, we compared detection using multi-class vs. one-against-all types of classifiers. Table 1 shows the advantage of multi-class detectors both in terms of P_E and MD5. As discussed in [5] multi-class detectors also learn differences between different stego schemes, which improves their detection performance.

3.2 Accuracy across image size

The payload scaling in the embedding script is made to follow the square root law of imperfect steganography in order to keep the detectability at a constant level across different crop sizes. Figure 3 shows the performance of theYC_rC_b-SRNet for QF 80 on ten image sizes. Notice that, technically, there are $4 \times 4 = 16$ different image sizes (Section 1) but rectangular images $A \times B$ and $B \times A$ have the same number of pixels, hence there are only ten unique sizes in terms of the number of pixels. This figure shows that larger crops are generally more difficult to steganalyze. This may be due to the fact that initially training on 256×256 tiles inherently penalizes the detector on larger images. Another reason for this, however, may be the payload size scaling in the embedding script. The square root law does not apply to the payload size but to the number of embedding changes. When optimal embedding simulators are used, as is the case of all four embedding schemes in ALASKA, the relationship between payload size and the number of embedding changes is non-linear. Instead of making the payload proportional to $\sqrt{N}$, where N is the number of cover elements, it should be asymptotically proportional to $\sqrt{N} \times \log N$ [17, 18], which may have contributed to the observed decrease of accuracy of our detectors with increased crop size.

3.3 Accuracy across quality factors

Next, we show how our detectors fared w.r.t. the quality factor. Figure 4 shows P_E and MD5 for theYC_rC_b-SRNet across JPEG quality factors 75–98 on TILEbase, TILEdouble, and on ARBITRARYbase. Note that the tile detector for double payload was trained only for multiples of 5 since curriculum learning [3] via the quality factor was used to obtain the remaining tile detectors directly for the base payload (Section 3.7.5). The general trend here is that the detection becomes harder towards larger quality factors. This, again, is most likely due to the payload size scaling w.r.t. quality factor in the embedding script. At this point, we wish to point out that when fixing the relative payload either in terms of bits per non-zero AC DCT or in terms of bpp, modern embedding schemes, such as J-UNIWARD and UED, which form 70% of stego images in ALASKArank tend to be harder to steganalyze until QF$\approx$ 96 − 98 after which their security starts decreasing [6].

Also note that the increase of the detection error from TILEbase to arbitrary images is already commented upon in the previous section.

3.4 Accuracy across stego schemes

In this section, we discuss the accuracy of our detectors on individual stego schemes. Figure 5 shows P_E and MD5 for theYC_rC_b-SRNet across JPEG quality factors 75–98 on ARBITRARYbase when tested on cover-stego pairs from one specific stego method. While the content-adaptive schemes are approximately detected with the same accuracy, nsF5 is markedly harder to detect. This is probably due to mis-scaled payload for nsF5 combined with a small prior of nsF5 stego images in minibatches – our detectors probably "sacrificed" the detection performance on nsF5 in favor of improved detection of stego methods occurring with larger priors.

Table 2: Detection performance for Y-SRNet, Y-DCTR and Y-GFR for QF 95 on ARBITRARYbase.

	SRNet		DCTR		GFR	
	YC_rC_b	Y	YC_rC_b	Y	YC_rC_b	Y
P_E	24.47	36.48	25.23	39.16	26.37	40.03
MD5	48.12	73.06	62.83	79.54	60.27	79.67

3.5 The surprising performance of rich models

Realizing the loss of detection accuracy for QFs in high 90's, midway through the competition, we performed another investigation to further improve our detectors. The study was executed for quality factor 95. First, we looked at the performance of the DCTR features [15] with the low-complexity linear classifier [9]. Since the feature computed from each channel had dimensionality of 8,000, the final feature representation of a color JPEG image was 24,000. The scalar test statistic obtained as the projection of the feature vector onto the weight vector of the linear classifier will be referred to as YC_rC_b-DCTR. Similarly, when trained only on channel $X \in \{Y, C_r, C_b\}$ this scalar is denoted X-DCTR.

Figure 6 shows the ROC curves of the YC_rC_b-SRNet and YC_rC_b-DCTR on ARBITRARYbase. While the detectors perform surprisingly close in terms of P_E, the network detector is much more accurate for low false alarms. In fact, all our detectors showed highly non-Gaussian ROC curves with low MD5 scores. Figure 7 shows the evolution of the MLP training using the Adamax optimizer [20]. The training starts by optimizing the detection for high true positive rates followed by optimizing the detection for low false alarm rates.

The original publication on SRNet [4] clearly demonstrated the superiority of the SRNet over rich models (SCA-GFR [10]) in terms of P_E, especially for large quality factors and small payloads. It was thus rather surprising that the P_E for the SRNet and DCTR on ARBITRARYsize was comparable. We believe this is due to improper payload scaling in chrominance channels. To better understand why, we trained both detectors only on the luminance channel: Y-SRNet, Y-DCTR, and Y-GFR [21]. Table 2 showing P_E and MD5 of all three detectors indicates that removing the chrominances channels enlarged the gap between SRNet and both rich models. The drop of performance when restricting the detectors to luminance means that the payload embedded in the chrominance channels is too large. Since chrominance is a "residual type of signal" with a narrower dynamic range, it is easier to detect embedding there even for weaker detectors and thus the SRNet provides comparatively smaller advantage than when analyzing grayscale images.

3.6 Channel separation

The key innovation that allowed us to further substantially improve the detection accuracy of the network detector was to train additional SRNet tile detectors on TILEbase when separating the channels and training only on luminance Y and only on C_r, C_b. As already discussed in Section 2.5, by separating the channels in this fashion, we hypothesize that we force the network to utilize embedding artifacts that may not be utilized when stronger embedding artifacts exist, for example, between luminance and the chrominance channels. Table 3 shows the effect of adding the features

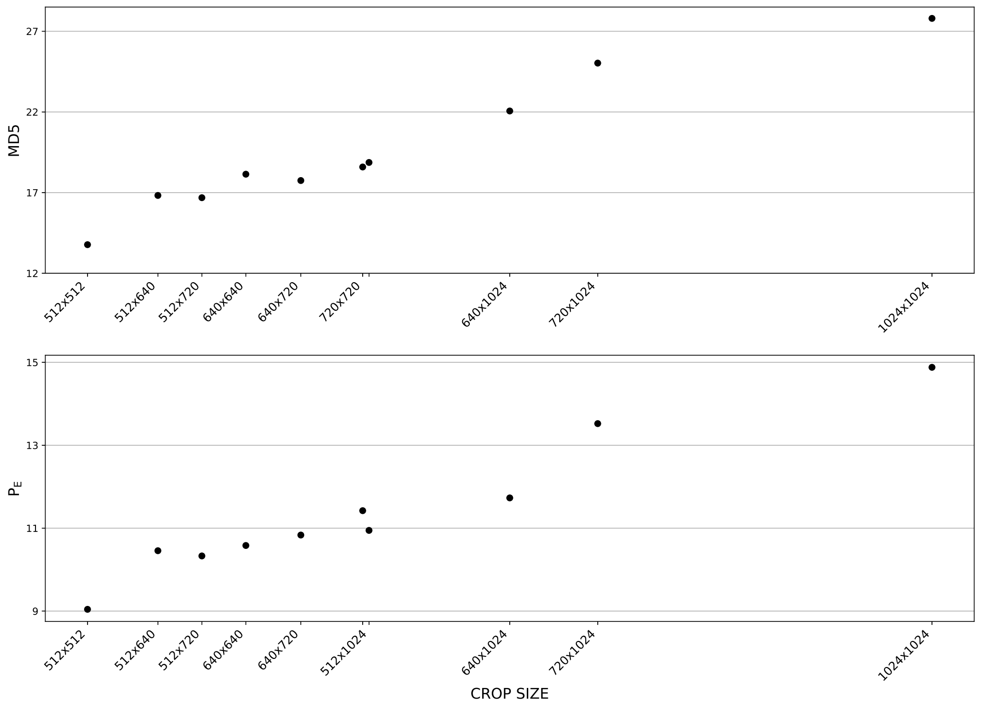

Figure 3: P_{E} and MD5 across various image sizes for JPEG quality 80.

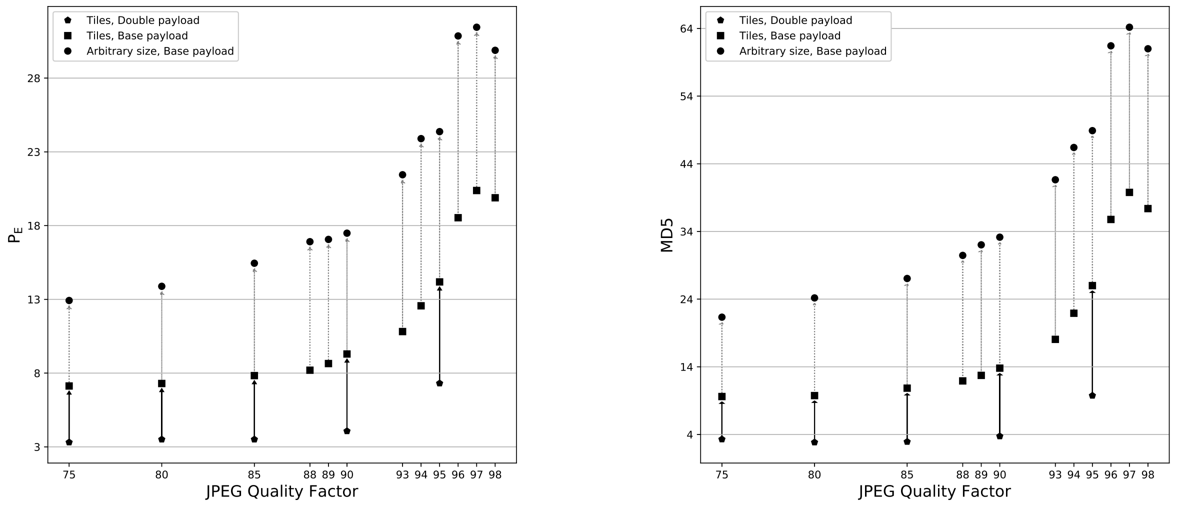

Figure 4: MD5 for the tile detector (YC_rC_b-SRNet) trained on TILEdouble, TILEbase, and the MLP on ARBITRARYbase across quality factors.

extracted using Y-SRNet and C_rC_b-SRNet (column '+Y, C_rC_b'), adding features from C_r-SRNet and C_b-SRNet (column '+C_r, C_b'), and even adding a single scalar – the projection of the DCTR feature on the weight vector determined by the low-complexity linear classifier. Additionally, the table also shows the effect of the number of hidden layers in the MLP for arbitrary size (one hidden layer vs. two hidden layers in column 'MLP'), training the MLP as a binary or multi-class classifier (column 'B/MC'), and including the four

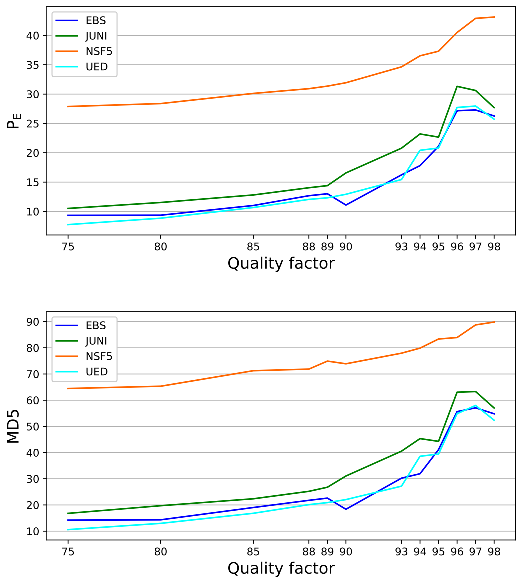

Figure 5: P_E and MD5 for YC_rC_b-SRNet on ARBITRARYbase for each stego scheme

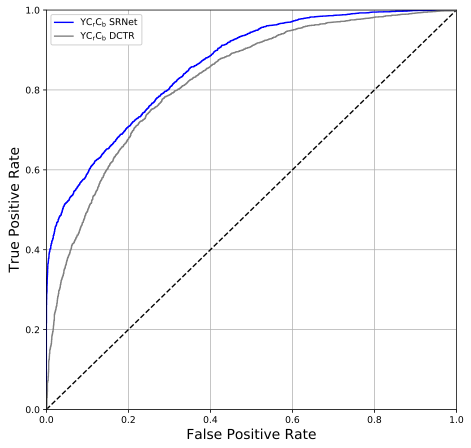

Figure 6: ROC curves of YC_rC_b-SRNet and YC_rC_b-DCTR on ARBITRARYbase for QF 95.

moments of feature maps or just their global means (column 'Moments'). While the table does not show all possible combinations, its inspection tells us that:

(1) The detector with the largest complexity – multi-class, two hidden layers in MLP, with four moments, and features from five versions of SRNet gave the best performance.
(2) By far the biggest improvement is due to adding the feature maps from the channel-separated SRNets. While the YC_rC_b-SRNet with a single hidden MLP layer gave MD5 = 0.481, the error dropped to MD5 = 0.407 after adding features from Y-SRNet and C_rC_b-SRNet. An additional boost of about 2% is observed when adding the feature maps from the C_r-SRNet and C_b-SRNet.
(3) The effect of adding DCTR is rather small (rows 7 and 8).
(4) Overall, two-hidden layers in the MLP and multi-class perform better than a single-hidden-layer MLP and a binary classifier.

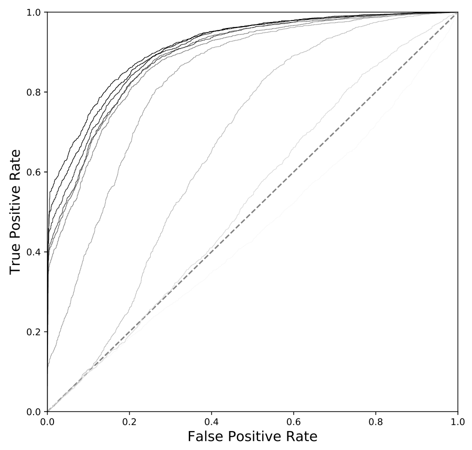

Figure 7: ROC curves for TILEbase detector for QF 95. Each curve corresponds to 10 iterations of optimization with the lightest shade corresponding to the onset of training.

At this point, we feel that it is important to mention that the organizers of the ALASKA challenge mistakenly embedded larger payload in the chrominance channels (and a smaller payload in luminance) than prescribed in the original work on color JPEG steganography [22] (see Section 2.3 in [8] for more details). The question remains whether the observed benefit of color separation demonstrated in this section also occurs when the payload is split among the luminance and the two chrominance channels correctly. To this end, we executed a limited experiment on the same datasets and with the same detector architectures but with stego images embedded as described in [22] with the parameter "beta," which controls the payload split, equal to 0.3. Comparing the detection performance of YC_rC_b-SRNet (row 9 in the table) and the final detector (row 12), we in fact observed an even larger gain of 15% in terms of MD5. Thus, the beneficial effect of color separation should not be attributed to the incorrect split of the payload among the color channels in ALASKA stego images.

3.7 Bag of tricks

In this section, we explain a few additional tricks that helped improve the detection performance.

3.7.1 Weighting soft outputs. As mentioned in Section 2.7, for the final submission, the outputs from detectors trained for quality factors other than 99 and 100 were ordered by $1 - p_0(x) = \sum_{k=1}^{4} p_k(x)$,

Table 3: Detection performance for different configurations of the detector for QF 95 ARBITRARYbase.

Row	B/MC	MLP	Moments	YC_rC_b	+DCTR	$+Y$	$+C_rC_b$	$+C_r, C_b$	MD5	FA50	P_E	Note
1	B	2	No	Yes	No	Yes	Yes	No	42.86	2.26	21.45	
2	B	2	Yes	Yes	Yes	Yes	Yes	No	42.70	2.41	20.62	
3	B	2	No	Yes	Yes	Yes	Yes	No	41.97	2.26	20.13	
4	B	2	Yes	Yes	No	Yes	Yes	No	41.32	2.20	20.99	
5	MC	2	No	Yes	Yes	Yes	Yes	No	40.70	2.02	20.06	
6	MC	2	Yes	Yes	Yes	Yes	Yes	No	40.44	1.66	19.58	
7	MC	1	Yes	Yes	No	Yes	Yes	No	40.83	1.87	20.45	
8	MC	1	Yes	Yes	Yes	Yes	Yes	No	40.67	1.71	19.71	
9	MC	1	Yes	Yes	No	No	No	No	48.13	3.85	24.51	YC_rC_b-SRNet
10	MC	2	Yes	Yes	No	Yes	Yes	No	40.70	2.08	20.48	
11	MC	2	Yes	No	No	No	Yes	No	48.47	4.57	24.08	C_rC_b-SRNet
12	MC	2	Yes	Yes	Yes	Yes	Yes	Yes	**38.31**	**1.38**	**19.25**	

where $p_0(x)$ is the output of the multi-class MLP for the cover class and $p_k(x)$, $k = 1, \dots, 4$, the outputs for stego classes. Based on error analysis in Section 4.2, we discovered that slightly better results are obtained by weighting the outputs of stego classes: $\sum_{k=1}^{4} w_k p_k(x)$, where w_k are suitably selected weights. Experimentally, we determined that $\mathbf{w} = (1, 1.1, 1.1, 1)$ for J-UNIWARD, UED, EBS, and nsF5 gave a slight improvement in our result.

3.7.2 Data augmentation at inference. Data augmentation in the form of flips and rotations is commonly applied during training because it effectively enlarges the training set. The learned convolutional kernels, however, do not necessarily exhibit any symmetries, which means that flipping and rotating a given test image usually produces slightly different feature maps from the tile detectors and consequently different soft-outputs from the MLP. To leverage these differences, we extracted eight (all rotation/flip combinations) soft-outputs from each image in the test set and averaged them. This provided a consistent boost over all quality factors of about 1% in terms of MD5 and 0.5% in terms of P_E.

3.7.3 Dropping the learning rate in the first few iterations . For larger quality factors, it appears that dropping the learning rate to 10^{-4} for the first 20,000 iterations then following the same learning rate schedule as in [4] helps with speeding up the convergence. We used this trick for all quality factors when training the tile detectors from scratch.

3.7.4 Out-of-range DCT coefficients. Texts on JPEG compression emphasize that the dynamic range of quantized DCT coefficients in a JPEG image is $[-1024, 1023]$. While this is a true statement, for any given quality factor and DCT mode, the dynamic range can be narrower. For example, the DC term can never be larger than 1016, as will be seen below.

Current JPEG steganographic schemes assign the so-called "wet costs," a very large value, to modifications that would change a DCT coefficient outside of the interval $[-1024, 1023]$. Thus, they fail to comply with the narrower dynamic range across DCT modes. This can introduce "impossible" values of DCT coefficients into the stego image and a hard evidence of stego embedding. While this does not happen often, when it does happen (and it will if the covert communication is sustained), it has grave consequences for the steganographer, who will be identified with certainty.

Given pixel values x_{ij} in an 8×8 block of an uncompressed image, the DCT coefficients before quantization are, $0 \le k, l \le 7$:

$$c_{kl} = \frac{1}{4} w_k w_l \sum_{i,j=0}^{7} s_{ij} \cos\frac{(2i+1)k\pi}{16} \cos\frac{(2j+1)l\pi}{16}, \qquad (6)$$

where $s_{ij} \in \{-128, \dots, 127\}$ are shifted pixel values, $s_{ij} = x_{ij} - 128$, and $w_0 = \frac{1}{\sqrt{2}}$, $w_k = 1$ for $k \neq 0$. From here,

$$|c_{kl}| \le \frac{1}{4} 128 \times 64 = 2^{11}. \qquad (7)$$

Therefore, after quantization (and rounding to integers), c_{kl} requires at most 12 bits. By making the bound tighter for each mode (k, l), we can show that not all values representable by 12 bits can be achieved in JPEG files.

Using the Iverson bracket $[P]_I = 1$ when P is true and $[P]_I = 0$ otherwise, we define

$$C_{ij}^{kl} = \cos\frac{(2i+1)k\pi}{16} \cos\frac{(2j+1)l\pi}{16} \qquad (8)$$

and

$$D_{ij}^{kl}(+) = 255 \cdot [C_{ij}^{kl} > 0]_I - 128 \qquad (9)$$

$$D_{ij}^{kl}(-) = 255 \cdot [C_{ij}^{kl} < 0]_I - 128. \qquad (10)$$

The upper and lower bounds on the coefficients are

$$c_{kl} \le \frac{1}{4} w_k w_l \sum_{i,j=0}^{7} C_{ij}^{kl} D_{ij}^{kl}(+) \qquad (11)$$

$$c_{kl} \ge \frac{1}{4} w_k w_l \sum_{i,j=0}^{7} C_{ij}^{kl} D_{ij}^{kl}(-). \qquad (12)$$

Denoting M_{kl} and m_{kl} the maximum and minimum attainable value of DCT coefficients in mode (k, l), when quantized with quantization matrix $\mathbf{Q}$, the maximum and minimum values of the quantized coefficients are

$$\mathbf{M}(\mathbf{Q}) = [\mathbf{M}./\mathbf{Q}] \qquad (13)$$

$$\mathbf{m}(\mathbf{Q}) = [\mathbf{m}./\mathbf{Q}], \qquad (14)$$

where $'./'$ is elementwise division.

Table 4: Learning rate schedule in terms of iterations for tile detectors during curriculum learning via payload and quality factor.

Pre-training	[0, 20000]	[20000, 170000]	[170000, 400000]
Learning rate	10^{-4}	10^{-3}	10^{-4}
Fine-tuning	[0, 170000]		[170000, 300000]
Learning rate	10^{-3}		10^{-4}

Table 5: Tile detector performance on TILEbase with and without payload curriculum learning for quality factor 75 **and** 95

	Without CL		With CL	
	MD5	P_E	MD5	P_E
75	15.69	10.09	9.60	7.12
95	95.00	50.00	13.80	9.29

The embedding simulator for J-UNIWARD assumes that the maximum value of coefficients is 1024 and therefore could in theory produce 'impossible' values, such as 1017 for the DC term (M_{00} = 1016). In our ARBITRARYbase set, we identified 69 stego images with out-of-range (OOR) coefficients, which corresponds to the probability of 0.0014 of a stego image violating the constraints. In ALASKArank, we found only one image with OOR coefficient – the DC term – for a quality-60 image '2221.jpg', which shows a fuzzy teddy bear. Given the fact that only about 10% of images in ALASKArank were stego, this is compatible with the expected number of stego images $500 \times 0.0014 = 0.7$.

3.7.5 Curriculum learning. Curriculum Learning (CL) [3] over payload and quality factor was used to speed up the training and overcome problems with convergence. To do so, training is split into two steps: pre-training and fine-tuning.

(1) Payload curriculum: Training a multi-class SRNet on 'TILEdouble' and then used it as an initialization point (seed) to train on 'TILEbase.'
(2) Quality factor curriculum: For the same payload size, seeding with a detector trained trained on a close quality factor.

Both curriculum strategies followed the learning rate schedule shown in Table 4.

3.8 Final detector structure

Due to limited time and resources, the following final detector structure was used at the end of the ALASKA challenge to generate the winning submission.

For the most populated quality factors in ALASKArank, 75, 80, 85, 90, 93, 94, 95, 96, 98, the detector was trained as in row 11 of Table 3. For quality factors 70, 88, 89, 91, 92, 93, 97, the detector shown in row 6 was trained (without C_r-SRNet and C_b-SRNet). For quality factors 99 and 100, we used the reverse JPEG compatibility attack explained in Section 2.6. For all remaining quality factors (60, 71, 72, 73, 74, 76, 78, 79, 81, 82, 83, 84, 86, 87) only the MLP was retrained with a feature extractor YC_rC_b-SRNet from the closest QF (row 9). Thus, for example, QF 75 was done with feature maps as in row 11 of the table, while the detector for QF 87 was done with the feature extractor from QF 88, etc.

3.9 Timeline

3.9.1 First submission. Our first submission to the ALASKA website done with detectors based on the SRNet was based on merely 7 detectors for quality factors 75, 80, 85, 90, 95, 99, and 100 with the 99 and 100 already covered by the reverse JPEG compatibility attack. The near 100% accuracy of the compatibility attack for quality 100 also gave us another piece of the puzzle – the fact that ALASKArank contained many more cover images than stego images. If the split between covers and stego images extended to other quality factors, ALASKArank contained only ≈ 10% (500) stego images.

Since we knew the detectors for 99 and 100 were considerably more reliable than the other detectors, and because the other quality factors were not covered by any detectors, we split the submission into three parts:

- S_1: 99 and 100: ordered by soft-outputs corresponding to the stego class
- S_2 : 75, 80, 85, 90, and 95 ordered by $1 - p_0(x)$ as explained in Section 2.7
- S_3: All other QFs randomly ordered, because at this time, we only had 7 detectors trained

Each part produces two orderings corresponding to the predicted cover and stego images, which we arranged in the following manner: $S_1(stego)$, $S_2(stego)$, S_3, $S_2(cover)$, $S_1(cover)$.

This arrangement requires making hard decisions on top of the ordering. This was done by using the default threshold for S_1, and grouping all stego schemes to a stego super-class for S_2 i. e., $\min(1, \widehat{y})$, where $\widehat{y}$ is the predicted class (0 for the cover class, and $\geqslant 1$ for the stego classes).

Even with this rather incomplete detector, we achieved MD5 = 0.4442 on November 8, 2018.

3.9.2 Improvements. In our next submissions we stopped randomly ordering S_3 and started using the detector from the closest QF. This did not lead to much improvement, especially in MD5, since S_3 is located in the middle of the ordering.

Our submission on February 1, 2019 marked a substantial improvement thanks to the channel separation (row 6). The next improvement on March 13, 2019 was due to additional channel separation (row 12). All other small improvements were mainly due to covering more quality factors, which inevitably improved our score as fewer images were ordered using mismatched detectors, as well as applying data augmentation at inference (Section 3.7.2). As of February 13, 2019 we stopped using the detector from the closest QF and started using (semi) dedicated detectors trained on feature maps computed using the tile detector from the closest QF.

4 ERROR ANALYSIS

In this section, we analyze our results on the mixTST dataset' (Section 2.2) using our final strategy.

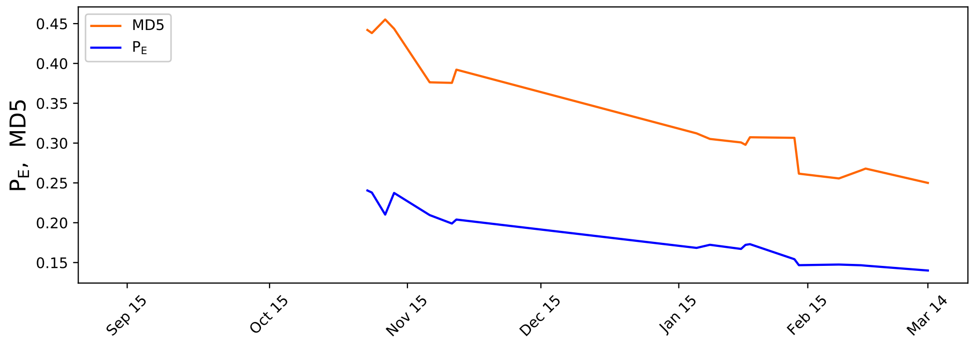

Figure 8: ALASKA scores (MD5 and P_E) over time.

Table 6: Final scores on mixTST and ALASKArank.

	MD5	P_E	FA50
mixTST	18.55	11.50	0.09
ALASKArank	25.2	14.63	0.77

Table 7: Distribution of false alarms in mixTST across predicted classes ('CA' is JPEG compatibility attack).

Predicted class	JUNI	nsF5	UED	EBS	CA
Portion of false alarms	42.7%	31.3%	19.8%	5.3%	0.8%

4.1 Performance of the final strategy

On mixTST, all three performance metrics are consistently better than on 'ALASKArank' when submitted online as shown in Table 6. As revealed by the organizers [8], this loss is mainly due to a cover source mismatch introduced by including 400 images in ALASKArank that were prepared from decompressed JPEGs instead of RAW images and developed using a script modified to skip the demosaicking.

4.2 False alarm analysis

In this section, we analyze all 131 false alarms observed on mixTST when using the default threshold of each detector. In Figures 9 and 7, we show the distribution of these false alarms across JPEG quality factors, and embedding algorithms.

The following was inferred from this error analysis :

(1) Larger quality factors (other than 99 – 100) introduce more false alarms, which is consistent with the results reported in Section 3.3.
(2) Most false alarms are predicted as J-UNIWARD embedded stego images with a portion very close to the initial J-UNIWARD prior in mixTST (0.4), followed by nsF5 with more than double of the corresponding prior. This is again consistent with the results in Section 3.4 showing that nsF5 is the least detectable embedding algorithm in the mixture.
(3) Only 19.8% and 5.3% of false alarms are predicted as UED and EBS, respectively, while the results in Section 3.4 show that these embedding algorithms are the most detectable. This is what gave us a hint that slightly more UED and EBS predicted images should be put in front of the ordering to improve the detection. This was done by weighting the soft outputs as described in Subsection 3.7.

5 CONCLUSIONS

As Neale Donald Walsch said, "Life begins at the end of your comfort zone." ALASKA definitely pushed all competitors to the next level and pay attention to aspects that get usually ignored in academic publications. When departing typical idealistic conditions, new problems arise and unexpected and sometimes contradictory results are obtained. As we reflect on the past six months, we first provide feedback regarding the competition itself and then lay out a condensed view of lessons learned together with a list of future directions.

ALASKA was designed with the motto "into the wild." While overall designed impressively well while paying attention to details, certain aspects of the competition were hardly "real life," such as images processed with a developing chain that was not very realistic. Some images in ALASKArank were extremely noisy, most likely due to excessive sharpening applied to an image acquired with a high ISO. Such unrealistic images, which are essentially noise with mere traces of content, are extremely unlikely to be encountered in practice, and probably also impossible to steganalyze. The second flaw was the payload distribution across color channels. Too large a payload was embedded in the chrominance channels to the point that restricting the detectors to just the chrominance would decrease the detection performance only little. The competitors thus trained on essentially faulty embedding schemes.

The approach chosen by our team was a natural progression of our previous research on deep learning architectures for steganalysis. When facing a multitude of embedding schemes, it is better to train a multi-class detector than one-against-all. With increased number of stego methods in the stego source, it is important to train with a sufficiently large minibatch to prevent noisy gradients. We addressed this by using Tensorflow's Estimators API, which allowed us to train with double the batch size than when training without them. The networks had to be trained via payload curriculum learning on double payload first since training directly on the base payload would not always produce good detectors or convergent networks.

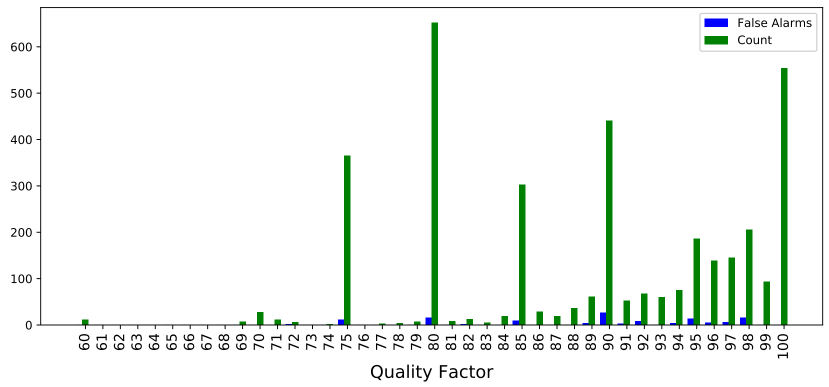

Figure 9: Distribution of false alarms in mixTST across JPEG quality factors.

The SRNet was also modified to accept multiple-channel input and several versions were trained by separating the color channels, which turned out the key ingredient that allowed us to further improve our detectors beyond their initial "off-the-shelf" form. This provided a significant boost over simply training a three-channel SRNet. This may be due to the way the channels were merged in the network. We plan to research alternative architectures that will keep the channels separate for a certain depth to remove the rather cumbersome training of five different versions of the same network.

While we trained a separate detector for each quality factor, this is not a scalable approach to cover, for example, non-standard quantization tables. Achieving scalability is among the tasks left for future research. Also, it is not clear if the way we approached the detection of arbitrary image sizes will scale to much larger images. Finally, ALASKA was a closed-set challenge and questions remain as to how well our detectors would detect stego images generated by previously unseen embedding schemes.

6 ACKNOWLEDGMENTS

The work on this paper was supported by NSF grant No. 1561446 and DARPA under agreement number FA8750-16-2-0173. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation there on. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied of DARPA or the U.S. Government.

REFERENCES

[1] Probability calibration. https://scikit-learn.org/stable/modules/calibration.html, 2019. [Online; accessed 07-Feb-2019].
[2] Tensoflow datasets for estimators. https://www.tensorflow.org/guide/datasets_for_estimators, 2019. [Online; accessed 07-Feb-2019].
[3] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.
[4] M. Boroumand, M. Chen, and J. Fridrich. Deep residual network for steganalysis of digital images. *IEEE Transactions on Information Forensics and Security*, 14(5):1181–1193, May 2019.
[5] J. Butora and J. Fridrich. Detection of diversified stego sources using CNNs. In A. Alattar and N. D. Memon, editors, *Proceedings IS&T, Electronic Imaging, Media Watermarking, Security, and Forensics 2019*, San Francisco, CA, January 14–17, 2019.
[6] J. Butora and J. Fridrich. Effect of jpeg quality on steganographic security. In R. Cogranne and L. Verdoliva, editors, *The 7th ACM Workshop on Information Hiding and Multimedia Security*, Paris, France, July 3–5, 2019. ACM Press.
[7] Heng-Tze Cheng, Zakaria Haque, Lichan Hong, Mustafa Ispir, Clemens Mewald, Illia Polosukhin, Georgios Roumpos, D Sculley, Jamie Smith, David Soergel, et al. Tensorflow estimators: Managing simplicity vs. flexibility in high-level machine learning frameworks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1763–1771. ACM, 2017.
[8] R. Cogranne, Q. Giboulot, and P. Bas. The ALASKA steganalysis challenge: A first step towards steganalysis âinto the wildâ. In R. Cogranne and L. Verdoliva, editors, *The 7th ACM Workshop on Information Hiding and Multimedia Security*, Paris, France, July 3–5, 2019. ACM Press.
[9] R. Cogranne, V. Sedighi, T. Pevný, and J. Fridrich. Is ensemble classifier needed for steganalysis in high-dimensional feature spaces? In *IEEE International Workshop on Information Forensics and Security*, Rome, Italy, November 16–19, 2015.
[10] T. Denemark, M. Boroumand, and J. Fridrich. Steganalysis features for content-adaptive JPEG steganography. *IEEE Transactions on Information Forensics and Security*, 11(8):1736–1746, August 2016.
[11] J. Fridrich. Feature-based steganalysis for JPEG images and its implications for future design of steganographic schemes. In J. Fridrich, editor, *Information Hiding, 6th International Workshop*, volume 3200 of Lecture Notes in Computer Science, pages 67–81, Toronto, Canada, May 23–25, 2004. Springer-Verlag, New York.
[12] C. Fuji-Tsang and J. Fridrich. Steganalyzing images of arbitrary size with CNNs. In A. Alattar and N. D. Memon, editors, *Proceedings IS&T, Electronic Imaging, Media Watermarking, Security, and Forensics 2018*, San Francisco, CA, January 29–February 1, 2018.
[13] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1321–1330. JMLR. org, 2017.
[14] L. Guo, J. Ni, and Y. Q. Shi. Uniform embedding for efficient JPEG steganography. *IEEE Transactions on Information Forensics and Security*, 9(5):814–825, May 2014.
[15] V. Holub and J. Fridrich. Low-complexity features for JPEG steganalysis using undecimated DCT. *IEEE Transactions on Information Forensics and Security*, 10(2):219–228, February 2015.
[16] V. Holub, J. Fridrich, and T. Denemark. Universal distortion design for steganography in an arbitrary domain. *EURASIP Journal on Information Security, Special Issue on Revised Selected Papers of the 1st ACM IH and MMS Workshop*, 2014:1, 2014.
[17] A. D. Ker. On the relationship between embedding costs and steganographic capacity. In M. Stamm, M. Kirchner, and S. Voloshynovskiy, editors, *The 5th ACM Workshop on Information Hiding and Multimedia Security*, Philadelphia, PA, June 20–22, 2017. ACM Press.
[18] A. D. Ker. The square root law of steganography. In M. Stamm, M. Kirchner, and S. Voloshynovskiy, editors, *The 5th ACM Workshop on Information Hiding and Multimedia Security*, Philadelphia, PA, June 20–22, 2017. ACM Press.
[19] A. D. Ker, T. Pevný, J. Kodovský, and J. Fridrich. The Square Root Law of steganographic capacity. In A. D. Ker, J. Dittmann, and J. Fridrich, editors, *Proceedings of the 10th ACM Multimedia & Security Workshop*, pages 107–116, Oxford, UK, September 22–23, 2008.

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, 2014. http://arxiv.org/abs/1412.6980.

[21] X. Song, F. Liu, C. Yang, X. Luo, and Y. Zhang. Steganalysis of adaptive JPEG steganography using 2D Gabor filters. In P. Comesana, J. Fridrich, and A. Alattar, editors, *3rd ACM IH&MMSec. Workshop*, Portland, Oregon, June 17–19, 2015.

[22] T. Taburet, L. Filstroff, P. Bas, and W. Sawaya. An empirical study of steganography and steganalysis of color images in the JPEG domain. In *International Workshop on Digital Forensics and Watermarking (IWDW)*, Jeju, South Korea, 2018.

[23] C. Wang and J. Ni. An efficient JPEG steganographic scheme based on the block–entropy of DCT coefficents. In *Proc. of IEEE ICASSP*, Kyoto, Japan, March 25–30, 2012.

[24] G. Xu. Deep convolutional neural network to detect J-UNIWARD. In M. Stamm, M. Kirchner, and S. Voloshynovskiy, editors, *The 5th ACM Workshop on Information Hiding and Multimedia Security*, Philadelphia, PA, June 20–22, 2017.

[25] J. Zeng, S. Tan, B. Li, and J. Huang. Large-scale JPEG image steganalysis using hybrid deep-learning framework. *IEEE Transactions on Information Forensics and Security*, 13(5):1200–1214, 2018.

[26] J. Zeng, S. Tan, G. Liu, Bin Li, and J. Huang. WISERNet: Wider separate-then-reunion network for steganalysis of color images. *CoRR*, abs/1803.04805, 2018.