Adversarial Detection Avoidance Attacks: Evaluating the robustness of perceptual hashing-based client-side scanning

Shubham Jain*, Ana-Maria Creţu*, and Yves-Alexandre de Montjoye†

Department of Computing and Data Science Institute, Imperial College London

Abstract

End-to-end encryption (E2EE) by messaging platforms enable people to securely and privately communicate with one another. Its widespread adoption however raised concerns that illegal content might now be shared undetected. Following the global pushback against key escrow systems, client-side scanning based on perceptual hashing has been recently proposed by tech companies, governments and researchers to detect illegal content in E2EE communications. We here propose the first framework to evaluate the robustness of perceptual hashing-based client-side scanning to detection avoidance attacks and show current systems to not be robust. More specifically, we propose three adversarial attacks–a general black-box attack and two white-box attacks for discrete cosine transform-based algorithms–against perceptual hashing algorithms. In a large-scale evaluation, we show perceptual hashing-based client-side scanning mechanisms to be highly vulnerable to detection avoidance attacks in a black-box setting, with more than 99.9% of images successfully attacked while preserving the content of the image. We furthermore show our attack to generate diverse perturbations, strongly suggesting that straightforward mitigation strategies would be ineffective. Finally, we show that the larger thresholds necessary to make the attack harder would probably require more than one billion images to be flagged and decrypted daily, raising strong privacy concerns. Taken together, our results shed serious doubts on the robustness of perceptual hashing-based client-side scanning mechanisms currently proposed by governments, organizations, and researchers around the world.

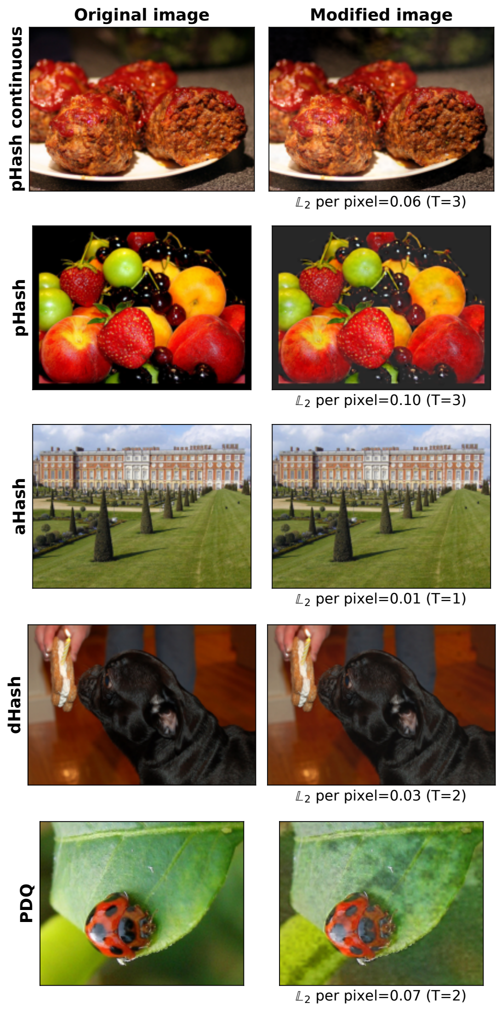
Original image
Modified image
pHash continuous
$\mathbb{L}_2$ per pixel=0.06 (T=3)
pHash
$\mathbb{L}_2$ per pixel=0.10 (T=3)
aHash
$\mathbb{L}_2$ per pixel=0.01 (T=1)
dHash
$\mathbb{L}_2$ per pixel=0.03 (T=2)
PDQ
$\mathbb{L}_2$ per pixel=0.07 (T=2)

Figure 1: Examples of original and modified images for different hashing functions. All four modified images evade detection for the threshold T while maintaining a small $\mathbb{L}_2$ perturbation per pixel and high visual similarity to the original image.

1 Introduction

More than two billion people across the world use end-to-end encryption (E2EE)-enabled platforms such as Signal and

*The first two authors contributed equally and are listed in random order.
† Corresponding author at demontjoye@imperial.ac.uk

WhatsApp [6, 53], exchanging more than 100 billion messages daily on WhatsApp alone [52]. E2EE provides a strong privacy protection to users, preventing governments, hackers, and platform providers themselves to access the content of their communications. Governments and organizations have however raised concerns that E2EE is preventing the detection of illegal content [39, 43] such as child sexual abuse media and terrorism-related content [41]. Governments have also recently been discussing bills looking at online safety including the countering of disinformation and misinformation online [17].

Following strong concerns from security experts and former national security officials alike [18, 31], key escrow systems providing governments an encryption "backdoor" have been abandoned. Instead, client-side scanning has been proposed to detect the sharing of illegal content on E2EE-enabled platforms by tech companies [4], researchers [22, 35, 45], and policy makers [11]. Here, a signature of a visual media (image, video) would be computed on the user's device and then compared against the database of signatures of known illegal images. If a match is found, the user would e.g. be flagged and/or the unencrypted content automatically shared for further review. Designed to be robust to small changes to the media, as well as transformations like rotation and rescaling, perceptual hashing algorithms would be used to generate the signature. Several variations of this scheme have been proposed. For instance, Apple's recent proposal sends a cryptographic voucher containing the encrypted images and the results of the match. If the number of successful matches reach a predefined threshold, the system is then able to decrypt all the matched images [4].

A large literature has developed recently in adversarial machine learning. The input of a machine learning classifier is adversarially modified to fool the model. Slightly modified but visually similar variants of an image can be designed resulting in a misclassification by the model [32,42,49]. Some of the most famous examples include an image of panda being misclassified as a gibbon [20] or a road works traffic sign to be misclassified as a give way traffic sign [42]. A wide range of techniques in both white and black-box setup have been developed, including [34] and [51].

Our contribution. We here propose the first framework to evaluate the robustness of perceptual hashing-based client side scanning to a novel *detection avoidance attack*. Specifically, our framework assesses the threat posed by adversarial attacks aiming to minimally modify an image so as to avoid detection while preserving its content. Our framework also takes into account the diversity of modifications produced by the attack in terms of distances between signatures. The more diverse the perturbation, the harder it is for the system to mitigate by expanding the database with the modified images.

We propose three adversarial attacks against perceptual hashing algorithms, a general black-box attack, inspired by previous work in adversarial ML, e.g., [24, 48, 59], and two novel and optimal white-box attacks exploiting the linearity and orthogonality of discrete cosine transform (DCT)-based perceptual hashing algorithms. The attacks are designed to minimally perturb the image while avoiding detection. In particular, they produce a wide range of perturbations preventing easy mitigation strategies such as expanding the database with modified images.

We evaluate the robustness of perceptual hashing-based client-side scanning mechanisms and show them to be highly vulnerable to detection avoidance attacks. We perform a large-scale extensive evaluation of five commonly used perceptual hashing algorithms: pHash (continuous and discrete) [12, 62], dHash, aHash and PDQ [27] and show that, for all of them, our attack manages avoid detection with modified images very similar to the original. Taken together, our results strongly suggest that perceptual hashing-based client-side scanning is highly vulnerable to small modifications in all scenarios considered, and that simple mitigation strategies like expanding the database would be ineffective. This sheds serious doubts on the robustness of currently proposed client-side scanning mechanisms based on perceptual hashing [11, 22, 35, 45].

2 Perceptual hashing-based client-side scanning

Perceptual hashing algorithms. Perceptual hashing algorithms compute a signature of a visual media (e.g., image, video) without having to share it. Perceptual hashes are different from cryptographic hashes in that the former changes gradually as the image changes, while the latter changes significantly as soon as a single pixel changes. Importantly, perceptual hashes are designed to detect instances of a visual media that are visually similar (e.g., a resized version) without being exact copies [38]. To achieve this, they extract features that remain invariant under small modifications, such as resizing, noise addition, format change and rotation. For example, pHash and Facebook's PDQ use the discrete cosine transform (DCT) to extract image features relating to lower frequencies in the DCT output, that remain invariant to image modifications such as blurring, resizing and watermarking [55].

Formally, a perceptual hashing algorithm $h : \mathcal{I} \to \mathcal{O}^l$ is a deterministic function mapping a multimedia $X \in \mathcal{I}$ to a fixed-size vector representation, the *hash*, usually consisting of bits ($\mathcal{O} = \{0,1\}$) or real numbers ($\mathcal{O} = \mathbb{R}$). The similarity between two media $X, X' \in \mathcal{I}$ is quantified by computing the *distance* between the hashes according to a metric, henceforth denoted by d. For bit-valued hashes, d is usually the Hamming distance, while for real-valued hashes a typical choice for d is the Euclidean distance.

Client-side scanning. Perceptual hashing-based client-side scanning (PH-CSS) for illegal image detection consists of a database $\mathcal{D} = \{X_1, \dots, X_N\}$ of N images $X_i \in \mathcal{I}, \forall 1 \leq i \leq N$, a perceptual hashing algorithm h, a distance d, and a threshold

$T > 0$. Given an image $X \in I$, the detection system computes the distance between the hash $h(X)$ and the hashes of images in the database $h(X_i)$. The image is flagged if there exists $1 \leq i \leq N$ such that $d(h(X_i), h(X)) \leq T$. We note that our framework is applicable more broadly to visual media (e.g., videos).

The detection system performance can be measured using the false positive and false negative rates [62].

False positive rate. For a threshold T and database $\mathcal{D}$, the *false positive rate* $FPR(T, \mathcal{D})$ is defined as the probability that a media visually different from those in the database ($X \not\sim \mathcal{D}$) is detected:

$$FPR(T, \mathcal{D}) = P(\exists 1 \leq i \leq N \,:\, d(h(X), h(X_i)) \leq T | X \not\sim \mathcal{D})$$

False negative rate. For a threshold T and database $\mathcal{D}$, the *false negative rate* $FNR(T, \mathcal{D})$ is defined as the probability that a media visually similar to medias in the database ($X \sim \mathcal{D}$) is wrongly rejected:

$$FNR(T, \mathcal{D}) = P(\forall 1 \leq i \leq N \,:\, d(h(X), h(X_i)) > T | X \sim \mathcal{D})$$

For both the FPR and FNR, the randomness is taken over the distribution of media shared by users.

The threshold T modulates the trade-off between the false positive and negative rates. Intuitively, as the threshold T decreases the system will wrongly flag fewer media outside the database. At the same time, it would wrongly reject more media visually similar to those in the database.

3 Attack model

3.1 Detection avoidance attack

We here propose an adversarial attack against perceptual hashing-based client-side scanning, which we call *detection avoidance attack*.

Attack model. We assume that a malicious agent, *the attacker*, is in possession of a image from the database, henceforth denoted *original image* $X \in \mathcal{D}$. The attacker's goal is to minimally modify X into X' such that its content is preserved while avoiding detection. More specifically, the attacker's goal is to modify X via an additive perturbation δ such that the modified image: (1) is valid, i.e., $X' = X + \delta \in I$ meaning that no pixel goes out of bounds, (2) evades detection, i.e., $d(h(X), h(X+\delta)) > T$, where T is the threshold used by the detection system and (3) the perturbation is minimal in terms of visual dissimilarity. We assume that the attacker knows the distance d and the threshold T. We revisit these assumptions later.

Assuming that the attacker quantifies the visual similarity between images using a metric v (with smaller values for higher visual similarity), the attacker's goal can alternatively be written as follows.

$$\begin{aligned} \text{Minimize: } & v(X, X+\delta) && (1)\\ \text{s.t.: } & d(h(X), h(X+\delta)) > T && (2)\\ & X + \delta \in I && (3) \end{aligned}$$

The objective function quantifies the visual similarity between the original image X and the modified image $X' = X + \delta$. The attacker seeks to minimize this objective in order to preserve the image content. The first constraint requires that the image should avoid detection by not being matched with the original image. However, it is possible the modified image might still be flagged because its hash is close ($\leq T$) to other images in the database. Finally, the second constraint requires that the modified image should be valid (within the image bounds).

Perturbation diversity requirement. Additionally, the attack should be resistant to simple defenses that the system could implement, such as expanding the database with hashes of modified images. This assumption is realistic as the detection system might gain knowledge of the attack. The attack should therefore produce a wide range of random perturbations that cannot be predicted and added to the database.

Attack intuition. We provide three intuitions for why perceptual hashing-based client-side scanning could be vulnerable to the proposed detection avoidance attack.

First, by design, the perceptual hash of an image changes gradually as the image changes, opening up the possibility to find a minimally modified image X' such that $d(h(X), h(X')) > T$ while X and X' remain visually similar.

Second, mitigation strategies like increasing the threshold T or expanding the database with hashes of modified images could lead to an increase in the detection system's false positive rate, rendering it unsuitable for the use case.

Third, the hash space likely contains many valid image hashes that are at least T away from the original image hash. Formally, the d-ball of radius T around the hash of the original image $B(X, T) = \{X' \in I \,:\, d(h(X), h(X')) \leq T\}$ is such that its complement $\overline{B(X, T)}$ contains potentially many images X' for typical distances (Hamming, Euclidean) and suitable threshold values. Furthermore, common perceptual hashing algorithms are non-injective, meaning that multiple inputs can lead to the same output. This suggests that it may be possible to obtain several and possibly many different perturbations of the same image such that $d(h(X), h(X')) > T$.

Finally, PH-CSS is similar but different from a classification model. It is a threshold-based detection system for matches in a database. This poses unique challenges: first, as the threshold T can be quite large, it is not trivial to see why adversarial perturbations can be found that still preserve the original image content, and second, the detection system could expand the database with hashes of adversarial images produced using our attack, making perturbation diversity a core requirement.

3.2 Attacker access to the perceptual hashing algorithm

We consider two levels of attacker access to the perceptual hashing algorithm.

Black-box access. Unless otherwise specified, we assume that the attacker has black-box access to the perceptual hashing algorithm. In a black-box setting, the attacker can provide an input image X to retrieve its corresponding hash $h(X)$ but does not know how the algorithm works. We believe this assumption to be realistic in the context of perceptual hashing-based client-side scanning context. Most of the proposed implementations of PH-CSS, including Apple's recent proposal, would compute the image hashes on the device [4, 8]. Our black-box assumption is further supported by the fact that Apple's model was made accessible in a recent iOS version, and by subsequent statements by Apple that this was "expected behavior" [13].

We further assume that the attacker can retrieve output hashes for as many inputs as needed, without the hashes being uploaded to the server. This can be made possible for example by automating the image upload through the app and creating a honeypot to catch all the requests, read the hash sent in the request and send it back to the attacker.

White-box access. In some cases, the attacker could have full knowledge of the algorithm used by the detection system. This can for example be the result of reverse-engineering work and the limited number of available perceptual hashing algorithms available. This knowledge consists of the rules or transformations used by the algorithm to map the input to the output hash. In this paper, we develop two novel attacks for DCT-based perceptual hashing algorithms such as the popular pHash and Facebook's PDQ (see Sec. 4.3). These attacks provide a strong attack targeting DCT as they are optimal in the sense of minimally modifying the input to evade detection. They also provide theoretical insights into the vulnerability of existing approaches.

4 Attack methodology

We present the methodology for a general black-box attack against perceptual hashing-based client side scanning and two novel attacks against DCT-based hashes.

4.1 Notation

General. We consider real-valued images $I = [0,1]^n$ and define an image as an element $X \in I$. The image size n is equal to the number of pixels in the image. For $1 \leq i \leq n$, we use X^i to denote the i-th element of the (flattened) image X. Given an image $X \in I$, the space of valid perturbations is denoted by $I - X = \{\delta \in [-1,1]^n \, : \, 0 \leq \delta^i + X^i \leq 1, \forall 1 \leq i \leq n\}$

Distance from original image. All our attacks aim to modify an original image X into $X' = X + \delta$ such that the distance between their respective hashes is larger than a given threshold T. We denote by $f_X : I \to O'$ the function mapping a perturbation δ to the distance between the hashes of $X' = X + \delta$ and X: $f_X(\delta) = d(h(X), h(X+\delta))$. For example, for real-valued hashes $O' = \mathbb{R}$, while for bit-valued hashes $O' \subset \mathbb{N}$ (for details see Section 5). For convenience and because each image is attacked separately, we drop the subscript and use $f := f_X$ when there is no ambiguity.

Visual similarity. We quantify the visual similarity between original image X and a modified image X' using the $\mathbb{L}_p$ norm of $X - X'$, where $\mathbb{L}_p : X \in \mathbb{R}^n \to ||X||_p = \big(\sum_{i=1}^{n} |X^i|^p\big)^{1/p}$. The Euclidean ($p=2$), $\mathbb{L}_1$ and $\mathbb{L}_\infty$ norms are common choices in the adversarial ML literature to measure visual similarity. To compare the visual similarity between original and modified images independently of the image size, we will use the $\mathbb{L}_p$ perturbation per pixel, defined as $\mathbb{L}_{p,\text{pixel}} : X \in \mathbb{R}^n \to \big(\frac{1}{n}\sum_{i=1}^{n} |X^i|^p\big)^{1/p}$.

4.2 Black-box attack

Approach. Our black-box attack attempts to maximize the distance $f(\delta)$ between the hashes of original and modified images X and $X' = X + \delta$ while ensuring the perturbation $||\delta||_p$ is smaller than a fixed constant ε. We take this approach instead of directly minimizing the perturbation under the constraint that $f(\delta) > T$ as we found it impractical to enforce the constraint on f in a black-box setting.

More specifically our attack seeks a solution to the following optimization problem:

$$\text{Find: } \max_{\delta} \min(T, f(\delta)) \tag{4}$$

$$\text{s.t.: } ||\delta||_p \leq \varepsilon \tag{5}$$

$$\delta \in I - X \tag{6}$$

The objective is either T or $f(\delta)$; when it is equal to the former, the program stops, while when it is equal to the latter, i.e., $f(\delta) \leq T$, we perform gradient ascent in search for a better solution. The first constraint requires that the perturbation's $\mathbb{L}_p$ norm does not exceed ε. The second constraint requires that the modified image remains valid.

To ensure that the perturbation is as small as possible, we start with a very small admissible perturbation ε and gradually increase it when the above program fails to find a solution within a reasonable amount of steps.

Gradient estimation. We use zero-order gradient estimation via Natural Evolutionary Strategies (NES) [48, 59] to estimate the gradient of the distance function f with respect to the perturbation δ and perform gradient ascent. This is a popular technique for optimization under black-box assumptions and has already been used to obtain adversarial perturbations against image classification models [24].

Algorithm 1 GRAD: Gradient estimation for $f(\delta)$

```
1: Inputs:
       $f$: Function whose gradient is to be estimated.
       $X$: Original image to be attacked, of size $n$.
       $\delta$: Point at which the gradient is to be estimated.
       $d'$: Number of Gaussian samples (should be even).
       $\sigma$: Scaling factor for Gaussian samples $\sim N(0, I_n)$.
2: Output:
       $grad$: $\nabla_\delta E[f(\delta)]$, estimate of the gradient of $f(\delta)$
3: Initialize:
       $\theta_i \leftarrow N(0, I_n)$, for $i \in \{1, ..., \frac{d'}{2}\}$
       $\theta_i \leftarrow -\theta_{d'-i+1}$, for $i \in \{(\frac{d'}{2}+1), ..., d'\}$
       $grad \leftarrow 0$
4: for $i = 1$ to $d'$ do
5:     $\theta'_i \leftarrow \max(\min(1, X+\delta+\sigma\theta), 0) - X - \delta$
6:     $grad \leftarrow grad + f(\delta + \theta'_i) * \theta'_i * \frac{1}{\sigma d'}$
7: end for
```

Algorithm 2 UPDATE: Estimating δ_{t+1} via gradient ascent

```
1: Inputs:
       $\delta_t$: Perturbation after the $t^{th}$ iteration.
       $grad_t$: $\nabla_\delta E[f(\delta_t)]$, estimate of the gradient of $f(\delta_t)$.
       $prev\_grad_t$: Weighted sum of previous gradients.
       $X$: Original image to be attacked.
       $\epsilon$: Upper bound on $||\delta||_p$.
       $\mu$: Momentum parameter.
       $\eta$: Step size to update $\delta$ in each iteration.
2: Output:
       $\delta_{t+1}$: Perturbation after the $(t+1)^{th}$ iteration.
       $prev\_grad_{t+1}$: Updated weighted sum of gradients.
3: $grad_t \leftarrow \mu * prev\_grad_t + (1-\mu) * grad_t$
4: $\delta_{t+1} \leftarrow \delta_t + \eta * \text{sign}(grad_t)$
5: $\delta_{t+1} \leftarrow \min(\max(X + \delta_{t+1}, 0), 1) - X$
6: if $||\delta_{t+1}||_p > \epsilon$ then
7:     $\delta_{t+1} \leftarrow \delta_{t+1} * \epsilon / ||\delta_{t+1}||_p$
8: end if
9: $prev\_grad_t \leftarrow grad_t$
```

The NES-based strategy for gradient estimation can be described as a special case of estimation using finite-differences on a random Gaussian basis. We estimate the gradient of f using the following equation:

$$\nabla_\delta E[f(\delta)] \approx \frac{1}{\sigma d} \sum_{i=1}^{d'} \delta_i f(\delta + \sigma\theta_i) \tag{7}$$

where $\theta_i \sim N(0, I_n), 1 \leq i \leq d'$ are samples from a standard multivariate normal distribution over $\mathbb{R}^n$. Nesterov and Spokoiny [40] showed through theoretical analysis that the number of samples d' required to estimate the gradient scales linearly with the input dimension n.

Alg. 1 details the black-box gradient estimation procedure. To reduce the variance of our estimate, we use antithetic sampling: we sample Gaussian noise θ_i for $i \in \{1, ..., \frac{d'}{2}\}$ and set $\theta_j = -\theta_{d'-j+1}$ for $j \in \{(\frac{d'}{2}+1), ..., d'\}$ with d' an even number (line 3). This optimization has been empirically shown to improve performance of NES [48]. To satisfy the image bound constraints for $X + \delta + \sigma\theta$ which will be given as input to the hashing function (in order to compute $f(\delta + \sigma\theta)$), we replace each noise sample with a clipped version (line 5).

Perturbation update. Alg. 2 details the perturbation update. At each step t, we use the sign of the estimated gradient, $\text{sign}(grad)$ along with momentum μ to update δ_t (lines 3-4). Sign gradients have been previously used by Goodfellow et al. to develop adversarial perturbations against image classification models [20]. We clip the resulting image $X + \delta_{t+1}$ to ensure it is within the image bounds (line 5), and we enforce the visual similarity constraint on the updated δ_{t+1} by projecting it onto the ball of $\mathbb{L}_p$ norm ϵ (lines 6-7).

Perturbation bound. To ensure that the visual similarity is comparable across images of different sizes, the bound on the perturbation norm (ϵ) is derived from a constant independent of the image size (denoted ϵ_{norm}) for each image attacked. Perturbing each pixel by 1% in the same image but with different sizes would result in a visually similar perturbation but different values of $||\delta||_p$. In the attack, we increment ϵ_{norm} starting from small values, and the corresponding actual perturbation bound used in the attack (dependent on the image size) will be $\epsilon = \epsilon_{\text{norm}} * n^{\frac{1}{p}}$.

Making the attack more efficient. Because the number of samples d' required to estimate the gradient scales linearly with the image size [40], we attack a small-dimension resized and grayscaled version of the original image, which we denote by $\bar{X}$ (of size $n' < n$). The attack produces a perturbation $\bar{\delta}$, which we map to a final perturbation δ in the original space using Alg. 3. Our strategy, which we show to work in practice, is informed by three insights: (1) most perceptual hashing algorithms are known to be invariant to grayscaling and resizing of the image [56]. This means that, for an image X, $h(X)$ does not change much even when X is recolored or resized. (2) both grayscaling, i.e., converting a three-channel RGB image to single-channel image, and resizing to a smaller sized image reduce the size of the original image X and (3) optimizing the perturbation for the grayscale image and mapping it to a corresponding perturbation in the RGB space leads to smoother perturbations than directly optimizing for the RGB image.

Alg. 3 details our algorithm for inverting the perturbation. Using a resize transformation, we map the single-channel perturbation $\bar{\delta}$ (of size n') to a corresponding perturbation of same single-channel size as the original image X ($n/3$), denoted by δ_{gray}. The final pixel value δ^i_c for channel c is derived from δ_{gray} such that (1) δ^i_c is proportional to X^i_c, the intensity for pixel i in the original image, and (2) the modified pixel $X_c^i + \delta_c^i$ is within the image bounds.

Algorithm 3 InverseDelta: Calculate perturbation δ for original image X from perturbation $\bar{\delta}$

```
1:  Inputs:
        $X$: Original image to be attacked of size $n$.
        $\bar{\delta}$: Optimal perturbation for $\bar{X}$.
2:  Output:
        $\delta$: Perturbation for $X$.
3:  Initialize:
        $X_c \leftarrow$ channel $c$ of $X$, for $c \in \{R,G,B\}$
        $\text{mean}(X_c) \leftarrow \frac{X_R+X_G+X_B}{3}$
        $n_{gray} \leftarrow n/3$
4:  $\delta_{gray} = \text{resize}(\bar{\delta}, n_{gray})$
5:  for $c$ in $\{R,G,B\}$ do
6:      for $i = 1$ to $n_{gray}$ do
7:          if $\delta_{gray}^i \leq 0$ then
8:              $\delta_c^i = \delta_{gray}^i \frac{X_c^i}{\text{mean}(X_c^i)}$
9:          else
10:             $\delta_c^i = \delta_{gray}^i \frac{1-X_c^i}{1-\text{mean}(X_c^i)}$
11:         end if
12:         $\delta_c^i = min(max(X_c^i + \delta_c^i, 0), 1) - X_c^i$
13:     end for
14: end for
```

Complete attack. The complete attack for an image and X of size n and threshold T is described in Alg. 4. The image-independent perturbation bound is initialized to ε_0. We grayscale and resize X to obtain $\bar{X}$ of smaller size n'. In each iteration t, the algorithm estimates the gradient and performs a gradient update. If the distance $f_{\bar{X}}(\bar{\delta})$ reaches a plateau as quantified by the finite difference Δf (and the number of previous iterations for which $\Delta f \leq 0$), we consider that the current ε is too small to allow for $f(\bar{\delta}) > T$. We thus increase the image-independent perturbation bound $\varepsilon_{\text{norm}}$ by a fixed step η_ε and update ε accordingly. As soon as the distance $f_{\bar{X}}(\bar{\delta}) > T$, δ is computed from $\bar{\delta}$. The algorithm stops if $f_X(\delta) > T$ as well, in which case the attack is considered successful. If after m iterations the distance $f_X(\delta)$ does not exceed T, the attack is considered unsuccessful.

Diversity of perturbations. For the attack to be resistant against simple mitigation strategies like expanding the database with hashes of the modified image X', the attack should generate multiple random perturbations for an image, resulting in different and unpredictable output hashes. To achieve this objective, we make two modifications to our black-box attack. First, we initialize our perturbation $\bar{\delta}_0$ to be a valid random non-zero perturbation such that $||\bar{\delta}_0||_p = \varepsilon_{start} * n'^{\frac{1}{p}}$ for some small value of ε_{start}. More specifically, we uniformly sample $\bar{\delta}_0$ from a set of valid perturbations, scale it to have an $\mathbb{L}_p$-norm of ε_{start} and clip to limit it

Algorithm 4 Black-box attack against perceptual hashing algorithms

```
1:  Inputs:
        $X$: Original image to be attacked of size $n$.
        $h$: Perceptual hashing algorithm to be attacked.
        $d$: Distance metric corresponding to perceptual hashing
        algorithm $h$.
        $T$: Threshold to be attacked.
        $n'(< n)$: Size of the resized and grayscaled image $\bar{X}$
        $\varepsilon_0$: Starting value for the maximum perturbation allowed.
        $\eta_\varepsilon$: Step size for $\varepsilon$.
        $k$: Number of previous iterations used to detect
        plateauing of $f_{\bar{X}}$.
        $\mu$, $\eta$: Parameters required for updating the gradient.
        $d'$, $\sigma$: Number of samples and scale for Gaussian
        samples used in gradient estimation.
        $s$: Seed defined by the attacker.
        $m$: Maximum number of iterations.
2:  Output:
        $\delta$: Perturbation for original image X.
3:  Initialize:
        $\bar{X} \leftarrow \text{resize}(\text{rgb2gray}(X), n')$
        $\bar{\delta}_0 \leftarrow 0$
        $prev\_grad_0 \leftarrow 0$
        $seed \leftarrow (X, s)$
        $\varepsilon_{norm} \leftarrow \varepsilon_0$
        $\varepsilon \leftarrow \varepsilon_{norm} * n^{\frac{1}{p}}$
4:  setseed($seed$)
5:  for $t = 0$ to $m-1$ do
6:      $grad_t \leftarrow \text{Grad}(f_{\bar{X}}, \bar{X}, \bar{\delta}_t, d', \sigma)$
7:      $\bar{\delta}_{t+1}, prev\_grad_{t+1} \leftarrow \text{Update}($
            $\bar{\delta}_t, grad_t, prev\_grad_t, \overline{X}, \varepsilon, \mu, \eta)$
8:      $\Delta f_t \leftarrow f_{\bar{X}}(\bar{\delta}_{t+1}) - f_{\bar{X}}(\bar{\delta}_t)$
9:      if Count($[\Delta f_{t-k+1}, ..., \Delta f_t] \leq 0) > \frac{k}{2}$ then
10:         $\varepsilon_{norm} \leftarrow \varepsilon_{norm} + \eta_\varepsilon$
11:         $\varepsilon \leftarrow \varepsilon_{norm} * n'^{\frac{1}{p}}$
12:     end if
13:     if $f_{\bar{X}}(\bar{\delta}_{t+1}) > T$ then
14:         $\delta = \text{InverseDelta}(X, \bar{\delta}_{t+1})$
15:         if $f_X(\delta) > T$ then
16:             break
17:         end if
18:     end if
19: end for
```

within bounds.

$$\bar{\delta}_0 \sim U(-\bar{X}, 1-\bar{X})$$
$$\bar{\delta}_0 = \varepsilon_{start} * n'^{\frac{1}{p}} * \frac{\bar{\delta}_0}{||\bar{\delta}_0||_p} \tag{8}$$
$$\bar{\delta}_0 = \min(\max(\bar{\delta}_0, 0), 1)$$

Second, we observe that requiring the perturbation to be smaller than the constant used in the default black-box attack can lead to less diverse outputs, even when different starting points are used in the optimization. So we allow for a larger maximum perturbation.

The two algorithm modifications lead to more diverse hashes, at a small cost to the visual similarity.

Reproducibility. In order to ensure our attack results are reproducible, we initialize the attack using both the image X and the attacker-defined seed s (line 4 in Alg. 4). This ensures that the directions θ being probed for gradient estimation are not the same for all images even when the attacker-defined seed s remains the same. For diversity, using different attacker-defined seeds leads to diverse perturbations for the same image. An attacker can either manually define the seed, or set it based on processor clock to generate a diverse perturbation everytime.

4.3 White-box attacks for DCT-based hashes

We develop a principled attack to devise a minimum perturbation for DCT-based perceptual hashing algorithms. The Discrete Cosine Transform (DCT) [2] is a very popular image compression algorithm. DCT maps a discrete signal to linear combinations of the original signal and cosines of different frequencies. The signal can then be compressed by noting that coefficients corresponding to lower frequencies encode the most important features. This property makes DCT suitable for JPEG image compression [58] and several perceptual hashing algorithms like pHash [12] and Facebook's PDQ [14].

Attack intuition. Our attack exploits the linearity and orthogonality of DCT. By using the Euclidean distance to measure both the input perturbation and the distance between original and modified hashes, we show that minimal perturbations can be found as linear combinations of eigenvectors derived from the linear transform. We further show that the minimal perturbation needed to exceed the threshold T is equal to T exactly.

Overview of DCT-based perceptual hashing. In what follows, we assume that the DCT operates on an image X of size $k \times k$. DCT-based perceptual hashing algorithms indeed typically work by applying a set of transformations to the original image, such as converting to grayscale, resizing, or blurring resulting in a smaller $k \times k$ image. The 2-dimensional DCT (see below) is applied to the $k \times k$ image, resulting in an output of the same size. Next, the dimensionality of the output is reduced by keeping a submatrix of pixels and discarding the rest. For the pHash algorithm, $k = 32$ and the 8×8 submatrix of pixels from the intersection of rows and columns 2 to 9 is preserved, while for PDQ $k = 64$ and the 16×16 submatrix of pixels from the intersection of rows and columns 2 to 17 is preserved.

DCT transform. The DCT transform for an image $k \times k$ is defined as follows:

$$h_{DCT} : X \longrightarrow MXM^T$$

$$M \in \mathbb{R}^{k \times k}, M_{ij} = \sqrt{\frac{2}{k}} \Lambda_i \cos\left[\frac{\pi}{k}\left(j + \frac{1}{2}\right) i\right], 1 \leq i, j \leq k \quad (9)$$

$$\Lambda_i = \begin{cases} \frac{1}{\sqrt{2}}, & \text{if } i = 0 \\ 1, & \text{otherwise} \end{cases} \quad (10)$$

M is orthogonal, i.e., $MM^T = I_k$.

The submatrix extraction step applied in perceptual hashing can be written as $h'_{DCT} : X \rightarrow (MXM^T)_{a:b,a:b}$, containing pixels from rows a to b and columns a to b, where $1 \leq a \leq b \leq k$ are parameters chosen when developing a perceptual hashing algorithm. a is typically small to extract lower frequencies while b controls the hash size. We rewrite the mapping as $h'_{DCT} : X \rightarrow M'XM'^T$, with $M' = M_{a:b,1:k} \in \mathbb{R}^{(b-a+1) \times k}$, i.e., M' is the matrix obtained by extracting rows a to b from M. Similarly to above, $M'M'^T = I_{b-a+1}$. For more conciseness we set from now on $c = b - a + 1$.

Linearity of DCT. It is straightforward to show that h'_{DCT} is a linear transformation mapping a $k \times k$-dimensional input to a $c \times c$-dimensional output. Simplifying the notation, h'_{DCT} can be rewritten as the linear transform of a vector (the flattened input matrix X):

$$h'_{DCT} : \begin{cases} I \longrightarrow \mathbb{R}^{c^2} \\ X \longrightarrow AX \end{cases}$$

$$\begin{gathered} A \in \mathbb{R}^{c^2 \times k^2} \\ A_{c_1 \times c + c_2, k_1 \times k + k_2} = M'_{1+c_1, 1+k_1} \times M'_{1+c_2, 1+k_2} \\ \text{for } 0 \leq c_1, c_2 \leq c - 1, 0 \leq k_1, k_2 \leq k - 1 \end{gathered} \quad (11)$$

An important property of A, which we exploit later, is that $AA^T = I_{c^2}$ (see Appendix). This is due to the orthonormality of the rows of M'.

Perturbation bound for a given threshold. We prove that when $f(\delta) \geq T$ then necessarily $||\delta||_2 \geq T$. Using d as the Euclidean distance, we can write $f_{DCT}(\delta) = ||h_{DCT}(X) - h_{DCT}(X + \delta)||_2 = ||A\delta||_2 = \sqrt{\delta^T A^T A \delta}$. Since $A^T A$ is symmetric, it is diagonalizable in a real orthonormal basis of eigenvectors $(\delta_i)_{1 \leq i \leq k^2}$ by the spectral theorem. The eigenvalues are non-negative because $A^T A$ is positive semidefinite. Since $AA^T = I_{c^2}$, it follows that the eigenvalues can only be 1 or 0. We prove in the Appendix that the multiplicities of eigenvalues are c^2 for 1 and $k^2 - c^2$ for 0. Without loss of generality, we reorder the eigenvectors by decreasing eigenvalue. Finally, if $\delta = \sum_{i=1}^{k^2} \alpha_i \delta_i$, we can write $f(\delta) = \sqrt{\sum_{i=1}^{c^2} \alpha_i^2} \leq \sqrt{\sum_{i=1}^{k^2} \alpha_i^2}$. Equality holds if and only if the perturbation δ belongs to the vector space spanned by eigenvectors $(\delta_i)_{1 \leq i \leq c^2}$.

We provide two practical ways to obtain perturbations δ such that $f(\delta) \geq T$.

Attack as an optimization program. We frame the attack as the following non-convex optimization program:

$$\begin{aligned} \text{Maximize} : & ||A\delta||_2^2 \\ \text{s. t.} : & ||\delta||_2^2 \leq T^2 \\ & \delta \in I - X \end{aligned} \tag{12}$$

The first constraint requires that the $\mathbb{L}_2$ norm of the input perturbation is no larger than T. The second constraint is linear and requires that the perturbed image is valid. We use the Disciplined Convex Concave Programming toolbox [50] to find a solution.

Attack using sampling. This attack samples valid perturbations of norm T *uniformly at random*, using rejection sampling. Alg. 5 describes our procedure to sample perturbations of norm T along eigenvectors that are not canceled by the linear mapping $(\delta_i)_{1\leq i\leq c^2}$ until a valid perturbation is found meaning that $\delta \in I - X$. Given a threshold T, we sample uniformly at random a vector α on the $\mathbb{L}_2$-sphere of radius T in c dimensions (lines 5-6). We set the perturbation to be $\delta = \sum_{i=1}^{c} \alpha_i \delta_i$ (line 7). Due to the orthonormality of the eigenvector basis, indeed $||\delta||_2^2 = \sum_{i=1}^{c} \alpha_i^2 = T^2$.

Algorithm 5 White-box attack using rejection sampling

```
1: Inputs:
       $T$: Threshold to be attacked.
       $(\delta_i)_{1\leq i\leq c}$: Orthonormal eigenvectors of $A^TA$ for
       eigenvalue 1.
2: Output:
       $\delta$: Minimal perturbation (of $\mathbb{L}_2$ norm $T$) such that
       $f_{DCT}(\delta) = T$.
3: while $\delta \notin I - X$ do
4:     $\alpha \sim N(0, I_c)$
5:     $\alpha \leftarrow T\frac{\alpha}{||\alpha||_2}$
6:     $\delta \leftarrow \sum_{i=1}^{c} \alpha_i \delta_i$
7: end while
```

5 Experimental setup

In this section, we describe the perceptual hashing algorithms used to instantiate our attack and how we implemented it.

5.1 Perceptual hashing algorithms

The perceptual hashing algorithms evaluated in this paper are pHash [12, 62], aHash, dHash [1], and Facebook's PDQ [27]. They are commonly used algorithms for image deduplication or retrieval.

[1]https://hackerfactor.com/blog/index.php%3F/archives/432-Looks-Like-It.html

pHash, aHash and dHash outputs a 64-bit hash while PDQ outputs a 256-bit hash for any input image. The Hamming distance is used to compare the outputs of the hashing algorithms. Each algorithm applies a sequence of transformations such as grayscaling and resizing followed by image feature extraction (e.g. using DCT), and finally a bit discretization step. To experiment with an algorithm with real-valued outputs (of size 64), we remove the discretization step of the pHash algorithm to obtain another algorithm, which we call pHash continuous. The Euclidean distance is used to compare the outputs of pHash continuous.

pHash and pHash continuous apply grayscaling, box blurring and resizing resulting in a 32×32 image. They then apply the DCT as explained in Sec. 4.3, with parameters $a = 1, b = 8, k = 32$ to get a 64-sized vector. We use this real-valued vector as the output of pHash continuous. pHash assigns bits larger than the median to 1 and 0 otherwise to obtain the final hash.

dHash and aHash apply grayscaling followed by resizing to get a 9×8 and 8×8 image, respectively. aHash outputs a 64-bit hash by flattening the image, and setting the pixels above mean value to 1 and 0 otherwise. dHash computes a difference between consecutive columns of the image, flattens the computed difference, sets the values greater than 0 to 1, and 0 otherwise to report a 64-bit hash.

PDQ is inspired by pHash and outputs a 256-bit hash. We will not attempt to describe the algorithm in detail, but to provide an overview. PDQ applies grayscaling to the image followed by a series of image transformations to get a 64×64 image. It then applies DCT as explained in Sec. 4.3 with $a = 1, b = 16, k = 64$, followed by bit quantization by setting bits larger than the median to 1, and 0 otherwise to finally obtain a 256-bit hash.

5.2 Attack parameters

Black-box attack. We use the following parameters to run our black-box attack. All the images are converted to grayscale and resized to $n' = 64 \times 64 = 4,096$ dimensions. We use the $\mathbb{L}_2$ norm to quantify visual similarity and $\varepsilon_0 = \eta_\varepsilon = 1/255$. The plateauing behavior of the objective function, i.e., when it stops increasing steadily, is detected based on the last $k = 10$ iterations. The number of Gaussian samples used for gradient estimation in each iteration is $d' = 800$. We run the attack for a maximum number of iterations $m = 10,000$, although in practice only a few hundred iterations are required for most images. We use a momentum $\mu = 0$ for all hashing algorithms but pHash continuous for which we use $\mu = 0.8$. The values for the scale of Gaussian noise σ and learning rate η are: $\sigma = 0.001, \eta = 0.001$ for pHash continuous, $\sigma = 0.1, \eta = 0.01$ for pHash, $\sigma = 0.1, \eta = 0.001$ for aHash, $\sigma = 0.1, \eta = 0.001$ for dHash and $\sigma = 0.1, \eta = 0.01$

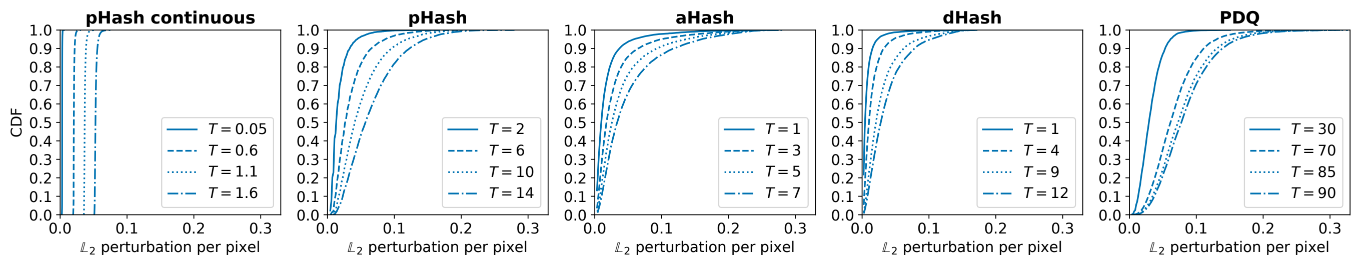

Figure 2: Cumulative distribution function (CDF) for the $\mathbb{L}_2$ perturbations per pixel for different algorithms and thresholds and all successfully attacked images over 10 experiments. A lower perturbation indicates higher visual similarity between the modified and original images. The perturbation increases slowly with the threshold, but remains small in all cases.

for PDQ. An attacker might want to adjust these parameters for each image, e.g., in order to run the attack more efficiently. For simplicity, we here use the same parameters for all the images. We chose the values of these parameters for each perceptual hashing algorithm based on early experiments on a handful of images from the Stanford Dogs dataset [28].

Diversity. We use the algorithm with modifications for diversity as detailed in Section 4.2. We set $\varepsilon_0 = 0.25$ and $\eta_0 = 0.01$. Our results on $\mathbb{L}_2$ perturbations per pixel for different algorithms and different thresholds in Fig. 2 show that pHash continuous requires only a small perturbation to generate successful perturbations, while for the other algorithms we need a threshold-specific ε_{start}. With these observations, we set $\varepsilon_{start} = 0$ for pHash continuous for all thresholds. For aHash and dHash we set $\varepsilon_{start} = 0.02, 0.03, 0.04, 0.05$, and for pHash and PDQ we set $\varepsilon_{start} = 0.04, 0.06, 0.08, 0.1$ for four respective thresholds.

Implementation details. We implement the framework and the perceptual hashing algorithms in Python 3.7 [44] using NumPy [23], OpenCV [7], and SciPy [57]. We further use DCCP [50] for white-box optimization and Faiss [25] for efficiently matching hashes against the database.

6 Results

We use the ImageNet dataset from ILSVRC 2012 challenge [47] for empirical evaluation of our attacks. For privacy reasons we discard the images containing faces as detected by Yang et al. [61]. At the end, we get a dataset with 1,187,974 images in total. We assume that no two images from ImageNet are visually similar.

For each repetition ($R = 10$), we sample two mutually exclusive sets of images, of sizes N and M, from ImageNet uniformly without replacement. The images in the first set are used to create the database $\mathcal{D}$ of size N (by default, we use $N = 100$k images) and the images in the second set are used as images visually different to those in the database, to estimate the false positive rate (we use $M = 500$k images). We select four thresholds for each perceptual hashing algorithm, shown in Table 1 (see Discussion for an analysis). We run the black-box attack on $N' = 1{,}000$ randomly sampled images from the database D. While the threshold will vary depending on the use case, the first two thresholds are values that would be typically used in practice (see e.g., [27] for PDQ). The third and in particular the fourth threshold would probably generate too many false positives in practice. We include them to test our attack in more challenging scenarios. Given a threshold T and N' images to be attacked, we compute the *attack effectiveness* as the proportion of images that are successfully attacked, i.e., a perturbation is found satisfying $f_X(\delta) > T$.

Algorithm	Thresholds
pHash continuous	0.05, 0.6, 1.1, 1.6
pHash	2, 6, 10, 14
aHash	1, 3, 5, 7
dHash	1, 4, 9, 12
PDQ	30, 70, 85, 90

Table 1: Thresholds selected for the detection system.

In our experiment, our black-box attack manages to successfully attack all images ($N' \times R = 10{,}000$) for all perceptual hashing algorithms but one, reaching an attack effectiveness of 99.9%. More specifically, we found that one image could not be attacked at the thresholds considered when using aHash for one particular instantiation of our attack . All other images were all successfully attacked for all perceptual hashing algorithms and thresholds considered.

Fig. 1 shows examples of images successfully perturbed using our attack along with the threshold used and resulting $\mathbb{L}_2$ perturbation per pixel. The modified image always preserves the visual content of the original image with PDQ seemingly requiring the most visually perceptible modifications. Values of up to $\mathbb{L}_{2,\text{pixel}} \approx 0.10$ resulting in very small changes are typically needed for most images even with the largest thresholds considered. We chose images where the subject is in focus so that the effect of perturbations can be clearly seen and we refer the reader to the Appendix section of the extended

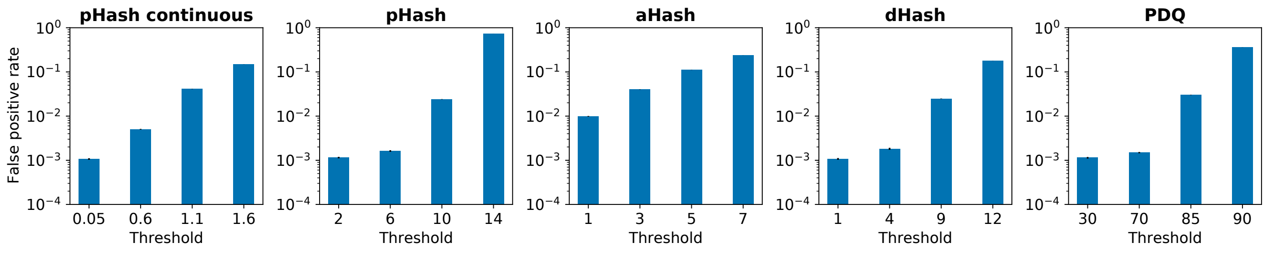

Figure 3: False Positive Rate (FPR) for different algorithms and thresholds. The database size is $N = 100$k, the number of images used to estimate the FPR is $M = 500$k and the number of attacked images is $N' = 1$k. We show the mean with error bars for the standard deviation (which is very small) over 10 repetitions.

version of our paper [2] for the complete results of our attack against these images for all perceptual hashing algorithms and thresholds.

Fig. 2 shows that a small $\mathbb{L}_2$ perturbation per pixel is enough to successfully attack most of the images for all hashing algorithms. We obtain similar results using the popular Learned Perceptual Image Patch Similarity (LPIPS) distance [64] (see Fig. 9 in the Appendix) that measures the perceptual distance between images using intermediate layers of a pre-trained image classification model. We can see that the perturbation increases overall with the threshold for all perceptual hashing algorithms. This is expected as a larger perturbation might be needed to push an image at least T away from the original. However, even for the largest thresholds, we observe that an $\mathbb{L}_2$ perturbation per pixel of 0.10 is enough to successfully attack with imperceptible modification 95.0% of images for dHash (respectively 100.0%, 83.7%, 87.2% and 73.3% for pHash continuous, pHash, aHash and PDQ). While results cannot be directly compared across algorithms as threshold are algorithm-specific, we observe interesting difference in the shape of the various algorithm's CDF. pHash continuous seems to requires a similar amount of perturbation for all images (at a given threshold) with, e.g., $\mathbb{L}_2$ for $T = 1.6$ only ranging between 0.050 and 0.077. pHash, dHash, and aHash have similar distributions while PDQ seems to always require some changes to the image (resulting in a CDF shifted to the right). We hypothesize the behavior we observe for pHash continuous to be due to two factors: (1) the attack mostly targets the DCT step where it finds an almost optimal $\mathbb{L}_2$ perturbation of $T = 1.6$ for an image size of 32×32, yielding an $\mathbb{L}_2$ perturbation per pixel of $\approx 1.6/32 = 0.05$ and (2) the other transformations from original to resized images (and from resized to original image to invert the perturbation) roughly preserve the $\mathbb{L}_2$ perturbation per pixel.

Fig. 3 shows that a detection system would have a very large False Positive Rate (FPR), even for the lowest threshold considered. We here empirically estimate FPR as the fraction of images among the M images outside the database flagged by the system. Even for the smallest threshold considered, the false positive rate is larger than 0.1%. While seemingly small, such values of FPR would, in practice, result in a large number of images being detected by the system and having e.g. to be sent unencrypted to be analyzed. In 2017, 4.5B images were shared daily on Whatsapp alone [5]. For a prevalence rate of illegal content of $p = 10^{-4}$ and a database size of $N = 100$k, this would result in >4M false positive images having to be shared unencrypted daily (see Table 2), raising very significant privacy concerns. For the largest thresholds, the false positive rate increases to rates between 14.8% for pHash continuous to 73.0% for pHash resulting in hundreds of millions of images being incorrectly flagged and shared daily. We show in the Appendix how this number does not vary much as a function of prevalence rates.

Hash	**Rough estimate of the number of wrongly detected images daily / Threshold T**			
pHash continuous	~ 5M $T = 0.05$	~ 23M $T = 0.6$	~ 184M $T = 1.1$	~ 665M $T = 1.6$
pHash	~ 5M $T = 0.2$	~ 7M $T = 6$	~ 108M $T = 10$	~ 3.3B $T = 14$
aHash	~ 44M $T = 1$	~ 179M $T = 3$	~ 498M $T = 5$	~ 1.1B $T = 7$
dHash	~ 5M $T = 1$	~ 8M $T = 4$	~ 112M $T = 9$	~ 798M $T = 12$
PDQ	~ 5M $T = 30$	~ 7M $T = 70$	~ 135M $T = 85$	~ 1.6B $T = 90$

Table 2: Rough estimate of the number of images that would be wrongly detected daily on WhatsApp alone for different perceptual hashing algorithms and thresholds. We here consider a database size of $N = 100$k, 4.5B images being shared daily [5], and a prevalence rate of illegal content of 10^{-4}.

Diversity. Contrary to the ML case, ensuring that an attack generates diverse enough perturbations every time it is run is essential to make it a practical concern for client-side scanning. Deterministic attacks could indeed easily be thwarted by expanding the database to include the original images and its modifications. To assess the ability of our at-

[2] https://arxiv.org/abs/2106.09820

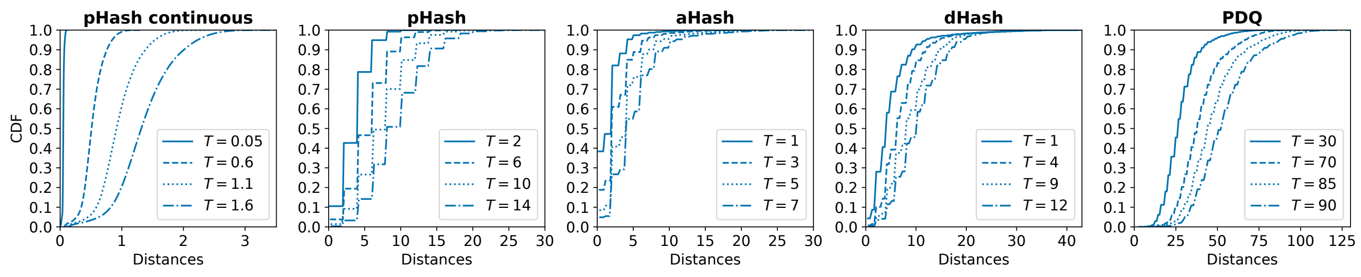

Figure 4: Pairwise distances between hashes of the multiple modified images generated for the same image, for different algorithms and thresholds. $D = 50$ modified images are generated for each ($N' = 100$) original image.

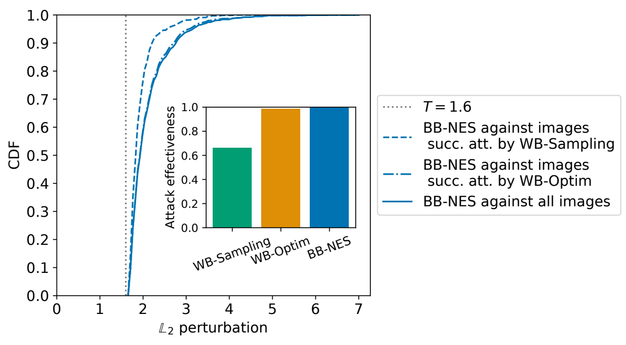

Figure 5: Cumulative distribution function (CDF) for the $\mathbb{L}_2$ norm of perturbations using our black-box attack (BB-NES) against the DCT step and a threshold $T = 1.6$ (gray dotted line) for $N' = 1{,}000$ images. The inset shows the attack effectiveness for the three attacks.

tack to produce different perturbations, we attack $N'' = 100$ images sampled uniformly at random without replacement from the database. Each image is attacked $D = 50$ times, using the diversity-focused version of the attack and different random initializations (seeds) (see Section 4.2). Fig. 10 (see Appendix) shows the $\mathbb{L}_2$ perturbation per pixel is slightly larger than in the default black-box attack. Yet, most perturbations are imperceptible and have $\mathbb{L}_2$ perturbation per pixel ≤ 0.15 for all thresholds and algorithms.

Fig. 4 shows the pairwise distances between hashes of modified images generated from the same image. The distances between modified images are roughly similar to the threshold used for all the perceptual hashing algorithms evaluated. This suggests that the modified images are well distributed around the ball centered on the original image and not clustered in one part of the space. This also strongly suggests that simple mitigation strategies like expanding the database would not be sufficient to counter the attack.

Finally, we assess the capacity of the black-box approach to produce optimal perturbations against the DCT step. We compare the results of the black-box attack to those of the white-box attacks and to our theoretical lower-bound (for a threshold T and the Euclidean distance, the $\mathbb{L}_2$-norm $\geq T$, see Sec. 4.3). We attack $N' = 1{,}000$ images randomly sampled from ImageNet after resizing them to $k \times k$ with $k = 32$ and use the black-box attack parameters from the DCT-based pHash continuous. For the WB-Sampling approach, we generate up to 1M samples and we consider the attack successful as soon as $\delta \in I - X$ (see Alg. 5 for details). For the WB-Optim approach, we run the optimization program for a maximum of 1,000 iterations and for up to 10 times, until a solution is found; if none is found, the attack is considered unsuccessful. Fig. 5 shows that the black-box attack achieves an input perturbation no larger than 2 times the theoretical limit 96% of the time and no more than 0.5 above the limit 74% of the time. Interestingly, in this experiment, while the black-box attack always succeeds in finding a perturbation, the white-box approaches are less successful. WB-Optim and WB-Sampling achieve an attack effectiveness of only 98.7% and 66.3% respectively. We believe that WB-Sampling is less successful because the sampled perturbations are more likely to go out of the image bounds for some images, resulting in no successful perturbation. Finally, we note that the $\mathbb{L}_2$ perturbation is smaller for the images successfully attacked by WB-Sampling, perhaps because it easier to devise small perturbations for these images.

7 Discussion

Our results and, in particular, the FPRs for specific thresholds are estimated using ImageNet. While ImageNet already contains more than 1M images, results might differ in different and, in particular, larger datasets. We randomly sample P images from ImageNet and, for each hashing algorithm, compute the distances between image hashes for up to 1M pairs. We find the distribution of distances between images to be roughly stable when the number of images varies for each algorithm. Similarly, we run the analysis on another dataset, Stanford Dogs [28], and show the distribution of distances to be similar to that of ImageNet and stable. This suggest that our FPRs can be safely extrapolated to larger and different

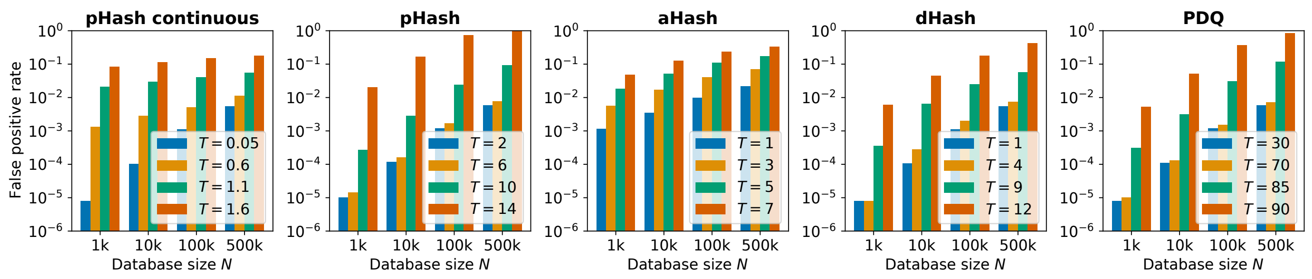

Figure 6: False Positive Rate (FPR) for each algorithm, threshold, and database size N. The number of images visually different from those in the database that are used to estimate the FPR is $M = 500$k.

datasets. For the figure supporting this analysis, we refer the reader to the Appendix of the extended version of our paper [3].

We have, throughout this work, assumed that the attacker knows the threshold and distance function used by the system to flag images. We believe this assumption to be reasonable as a) pairwise distances are fairly similar across datasets and stable, allowing an attacker to reasonably estimate acceptable thresholds and b) our attack works well, producing images very similar to the original one, even with very high thresholds. We also believe the assumption that the attacker knows the distance function used to be reasonable, as an attacker is likely to be able to infer the distance being used based on the output values, especially with the Euclidean and Hamming distance being the most commonly used.

While we report results on a database size of $N = 100$k, larger databases are often used in practice (e.g., the National Center for Missing and Exploited Children in the US reported a database size of 47.2M [1] and the Global Internet Forum for Counter Terrorism reported a database of size 250k [19]). Fig. 6 shows that the FPR of a system increases with the database size N. Even for the lowest threshold, the FPR for $N = 500$k reaches 0.55% for pHash continuous ($T = 0.05$), 0.59% ($T = 2$) for pHash, 2.13% ($T = 1$) for aHash, 0.55% ($T = 1$) for dHash and 0.59% ($T = 30$) for PDQ. This strongly suggest that the FPRs we report are a lower bound on the FPRs and therefore the privacy risk in practice.

We have so far considered our attack to be successful when the modified image is at a distance $> T$ from the original image. In practice, however, a modified image might be at a distance $> T$ from the original image but still be at a distance $\leq T$ from another image in the database and be (correctly, even if for the wrong reason) flagged. To evaluate the impact this has on the detection avoidance capability of our attack, we compute the False Negative Rate (FNR), the fraction of modified images incorrectly rejected by the system. Fig. 7 shows that the FNR stays extremely high for all perceptual hashing algorithms at reasonable thresholds, indicating that very few modified images get flagged for being at a distance $\leq T$ from another image in the database. As expected, FNR decreases (and FPR increases) when databases becomes large and a large threshold is used, leading to a significant fraction of the space being considered "close" to at least one illegal image. For instance, for a database size $N = 500$k and the highest threshold, 2.2% of modified images would still be flagged for pHash and 23.3% for PDQ.

Our black-box attack relies on a heuristic to minimize the perturbation while increasing the distance to the original image above a threshold T. Although we produce imperceptible perturbations for most of the attacked images, our results are only an upper estimate of the actual minimum perturbation. Prior work studying the robustness of machine learning classification models to adversarial perturbations showed that finding minimal perturbation is a hard problem [26] and even lower perturbations might be achieved in future work.

We focus, in our white-box attack, on DCT-based perceptual hashing algorithms, exploiting the linearity of the DCT transform. Our attack is likely to be extendable to other linear image transforms, e.g. image scaling [60], opening the door to future attacks as well as potentially novel theoretical insights.

Countermeasures. We discuss several countermeasures that a detection system could implement to thwart our attack.

First, the system could use a larger threshold. We showed in Sec. 6 that the larger thresholds are not only ineffective in detecting adversarially modified images but also lead to a significant increase in false positives.

Second, one could use our attacks to generate hashes and add them to the database. We show in Sec. 6 that our attack is able to generate diverse perturbations so as to prevent such countermeasures.

Third, systems (such as Apple's protocol [4]) could flag users only after the number of matches exceed a predefined threshold k, rather than on a single match. Using a simple model, we show that such a measure is not a trivial countermeasure to our attack.

More specifically, we assume that both offenders and non-offenders send 1000 images to the server. However, while a) non-offenders send 0 illegal images b) offenders send exactly 100 illegal images (10%). For each type of user and each threshold, we compute the probability for the system to flag $1 \leq k \leq N$ images using the estimated FPR and FNR for a

[3] https://arxiv.org/abs/2106.09820

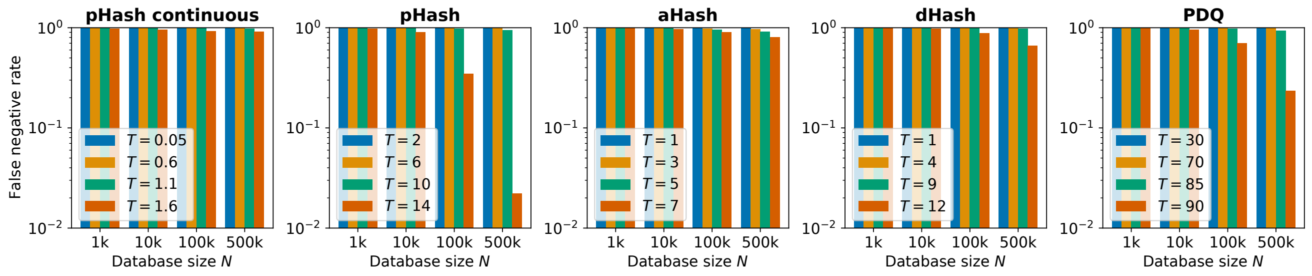

Figure 7: False Negative Rate (FNR) for each algorithm, threshold, and database size N. The number of attacked images used to estimate the FNR is $N' = 1,000$.

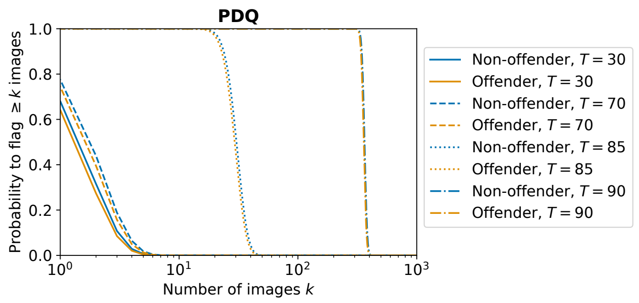

Figure 8: Probability that a detection system flags at least k images shared by non-offenders (sharing 0 illegal images) and offenders (sharing 100 illegal images). Both offenders and non-offenders share a total of 1,000 images.

database size of 100k (see Appendix for the exact formulas). Fig 8 shows that offenders and non-offenders are similarly likely to have at least k images flagged. We obtain similar results for the other perceptual hashing algorithms. This can be attributed to the fact that the probability for a non-illegal image to get flagged (FPR) is comparable to the probability for an adversarially modified illegal image to be flagged (true positive rate or $TPR = 1 - FNR$). These results suggest that flagging a user with at least k matches is not a trivial countermeasure against our attack. We leave a deeper analysis of such a mechanism for future work.

Fourth, the detection system could apply additional image transformations before computing the perceptual hash of the image. Such modifications would however be part of the black-box and attacked by our algorithm. Simple transformations are thus unlikely to prevent our adversarial detection avoidance attack. Similarly, modifying the perceptual hashing algorithm for instance by using a secret set of DCT coefficients that is different from the set used by pHash or PDQ is unlikely to prevent our attack. These would furthermore make the algorithm less robust as the coefficients extracted by pHash and PDQ represent the most important features of the image.

Fifth, increasing the length of the hashes is likely to help better distinguish between different images and reduce the false positive rate. This might allow the system to increase what are realistic thresholds and make our attack harder. We here use hashes of size 64 (and 256 for PDQ [27], see Sec. 5.2). Most perceptual hashing algorithms can be adapted to provide hashes of different sizes. For instance, pHash, aHash and dHash can all output hashes of size 256 [4]. While further analysis is required, we note here that a) our attack works well even for large thresholds and b) that longer hashes encode more information about the image. This raises strong privacy concerns e.g. reversal attacks [33] and, in the extreme, defeats the purpose of perceptual hashing.

Implications. Our research shows that current perceptual hashing (PH) algorithms are not robust to black-box detection avoidance attacks and that no straightforward mitigations strategies exist. We believe PH algorithms, which are designed to produce hashes that change gradually as the image changes, might be inherently vulnerable to attacks. Our results, combined with the concerns PH-CSS and in particular cryptographic-enhanced PH-CSS such as Apple's [4] raise on the "illegal" content being searched for, led us to believe that even the best PH-CSS proposals today are not ready for deployment.

8 Related work

Robustness of perceptual hashing algorithms. Previous work have shown perceptual hashing algorithms to be robust to small standard image transformations like resizing, recoloring, watermarking, cropping, and blurring [16, 63]. They evaluate the performances of perceptual hashing algorithms against modifications in duplicate image detection setup, i.e. matching against a single image. They however focus on standard image transformations and do not consider adversaries with more sophisticated tools at their disposal.

Dolhansky and Ferrer [15] studied collision (false positive) attacks in perceptual hashing algorithms under white-box assumptions. More specifically, their algorithm modifies an image such that its hash is same as the hash of a given target image. They conclude that perceptual hashing algorithms should remain secret. In this work, we instead focus on the

[4]https://github.com/thorn-oss/perception

client-side use case proposed recently by researchers and policy makers [11, 36] where the attacker has a black-box access to the algorithm. Furthermore, our attack is a false negative attack, aiming at finding minimal diverse perturbations such that distance between the hashes of modified and input image is greater than a given threshold.

Adversarial attacks against ML models. Adversarial attacks have been widely studied against ML models [46, 49] under both white-box [9, 20, 30, 54] and black-box assumptions [32, 42]. Image classification models have been particularly found to be vulnerable to adversarial attacks [3] leading the image to be misclassified by the ML model. Our attack leverages previous work in adversarial ML including Natural Evolutionary Strategies (NES) (see Sec. 4.2).

Adversarial attacks on ML models in a black-box setting assume that the attacker has an access to query the model and get the output. A commonly observed approach is to train a substitute model to emulate the target model, and then attack the surrogate model [42, 51]. This approach implicitly assumes that the surrogate model has enough parameters to provide a good approximation of the target model. This would imply that not all substitute models would work for all the target models and hence the attack might not always work.

Another approach is to iteratively perturb the image and query the model to update the perturbation in each iteration [21]. These methods often use gradient estimation to estimate the perturbation update in each iteration. Natural evolutionary strategies (NES) for gradient estimation is one of the state-of-the-art methods in black-box adversarial ML [10, 24, 37]. Our black-box attack also uses NES-based strategy for gradient estimation and perturbation update. Lastly, we note that generating multiple perturbations for the same image for diversity in the output space is not a requirement in the adversarial ML setup, while it is an important requirement for adversarial attacks against PH-CSS.

9 Conclusion

In this paper, we introduced the first framework to evaluate the robustness of perceptual hashing-based client-side scanning against adversarial attacks. We proposed a general black-box attack and showed that $> 99.9\%$ of images can be successfully modified while preserving the image content. We also show our attack to generate diverse perturbations preventing straightforward mitigation strategies such as expanding the database with modified images. We finally propose two white-box attacks, providing a theoretical basis for attacks.

Taken together, our results shed strong doubt on the robustness to adversarial black-box attacks of perceptual hashing-based client-side scanning as currently proposed. The detection thresholds necessary to make the attack harder are likely to be very large, probably requiring more than one billion images to be wrongly flagged daily, raising strong privacy concerns.

Acknowledgements

We thank the Computational Privacy Group and, in particular, Florimond Houssiau and Ali Farzanehfar for their helpful feedback and comments on the manuscript. We also would like to thank the reviewers, chairs, and our shepherd for their useful feedback and help improving the paper.

References

[1] The internet investigation report march 2020. https://www.iicsa.org.uk/document/internet-investigation-report-march-2020, 2020. Accessed on June 8, 2021.

[2] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transform. *IEEE TRANSACTIONS ON COMPUTERS*, page 4, 1974.

[3] N. Akhtar and A. Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *Ieee Access*, 6:14410–14430, 2018.

[4] Apple. Expanded protections for children. https://www.apple.com/child-safety/. Accessed on Sep 18, 2021.

[5] WhatsApp blog. Connecting one billion users every day. https://blog.whatsapp.com/connecting-one-billion-users-every-day, 2017. Accessed on June 6, 2021.

[6] Whatsapp Blog. Two billion users – connecting the world privately. https://blog.whatsapp.com/two-billion-users-connecting-the-world-privately/, 2020. Accessed on June 6, 2021.

[7] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[8] J. Callas. Thoughts on mitigating abuse in an end-to-end world. page 16, Jan 2020.

[9] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[10] M. Cheng, T. Le, P.-Y. Chen, H. Zhang, J. Yi, and C.-J. Hsieh. Query-efficient hard-label black-box attack: An optimization-based approach. In *International Conference on Learning Representations*, 2018.

[11] European Commission. Technical solutions to detect child sexual abuse in end-to-end encryption communications, 2020.

[12] B. Coskun and B. Sankur. Robust video hash extraction. In *Proceedings of the IEEE 12th Signal Processing and Communications Applications Conference, 2004.*, page 292–295, Apr 2004.

[13] J. Cox, L. Franceschi-Bicchieral, and S. Cole. Apple defends its anti-child abuse imagery tech after claims of 'hash collisions'.

[14] A. Davis and G. Rosen. Open-sourcing photo- and video-matching technology to make the internet safer, 2019. https://about.fb.com/news/2019/08/open-source-photo-video-matching/.

[15] B. Dolhansky and C. C. Ferrer. Adversarial collision attacks on image hashing functions. *arXiv:2011.09473*, 2020.

[16] A. Drmic, M. Silic, G. Delac, K. Vladimir, and A. S. Kurdija. Evaluating robustness of perceptual image hashing algorithms. In *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, 2017.

[17] Department for DCMS. Draft online safety bill. https://www.gov.uk/government/publications/draft-online-safety-bill. Accessed on Sep 21, 2021.

[18] L. Franceschi-Bicchierai. Former nsa chief: I 'would not support' encryption backdoors. https://www.vice.com/en/article/8qxwda/former-nsa-chief-strongly-disagrees-with-current-nsa-chief-on-encryption, 2015.

[19] GIFCT. Gifct transparency report july 2020. https://gifct.org/wp-content/uploads/2020/10/GIFCT-Transparency-Report-July-2020-Final.pdf, 2020. Accessed on June 9, 2021.

[20] I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[21] C. Guo, J. Gardner, Y. You, A. G. Wilson, and K. Weinberger. Simple black-box adversarial attacks. In *International Conference on Machine Learning*, pages 2484–2493. PMLR, 2019.

[22] H. Gupta and H. Taneja. Whatsapp has a fake news problem—that can be fixed without breaking encryption. https://www.cjr.org/tow_center/whatsapp-doesnt-have-to-break-encryption-to-beat-fake-news.php, Aug 2018. Accessed on June 6, 2021.

[23] C. R. Harris et al. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[24] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, page 2137–2146. PMLR, Jul 2018.

[25] J. Johnson, M. Douze, and H. Jégou. Billion-scale similarity search with gpus. *arXiv:1702.08734*, 2017.

[26] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International Conference on Computer Aided Verification*. Springer, 2017.

[27] John Kerl. The TMK+PDQF video-hashing algorithim and the PDQ image hashing algorithm. https://github.com/facebook/ThreatExchange/blob/master/hashing/hashing.pdf, 2020. Accessed on June 7, 2021.

[28] A. Khosla, N. Jayadevaprakash, B. Yao, and F. Li. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011.

[29] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in neural information processing systems*, 2012.

[30] A. Kurakin, I. Goodfellow, S. Bengio, et al. Adversarial examples in the physical world, 2016.

[31] I. Levy and C. Robinson. Principles for a more informed exceptional access debate. https://www.lawfareblog.com/principles-more-informed-exceptional-access-debate, 2018.

[32] Y. Liu, X. Chen, C. Liu, and D. Song. Delving into transferable adversarial examples and black-box attacks. *arXiv:1611.02770*, 2016.

[33] Nick Locascio. Black-box attacks on perceptual image hashes with gans. https://towardsdatascience.com/black-box-attacks-on-perceptual-image-hashes-with-gans-cc1be11f277, 2018. Accessed on June 9, 2021.

[34] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[35] Jonathan Mayer. Content moderation for end-to-end encrypted messaging. *Princeton University*, 2019.

[36] P. Melo, J. Messias, G. Resende, K. Garimella, J. Almeida, and F. Benevenuto. Whatsapp monitor: A fact-checking system for whatsapp. *Proceedings of the International AAAI Conference on Web and Social Media*, 13:676–677, Jul 2019.

[37] L. Meunier, J. Atif, and O. Teytaud. Yet another but more efficient black-box adversarial attack: tiling and evolution strategies. *arXiv:1910.02244*, 2019.

[38] M. K. Mıhçak and R. Venkatesan. New iterative geometric methods for robust perceptual image hashing. In *ACM Workshop on Digital Rights Management*, pages 13–21. Springer, 2001.

[39] NCMEC. Ncmec's statement regarding end-to-end encryption. https://www.missingkids.org/blog/2019/post-update/end-to-end-encryption, Mar 2019. Accessed on June 6, 2021.

[40] Y. Nesterov and V. Spokoiny. Random gradient-free minimization of convex functions. *Foundations of Computational Mathematics*, 17(2):527–566, 2017.

[41] Home Office. Interim code of practice on online child sexual exploitation and abuse (accessible version). https://www.gov.uk/government/publications/online-harms-interim-codes-of-practice/interim-code-of-practice-on-online-child-sexual-exploitation-and-abuse-accessible-version, Dec 2020. Accessed on June 6, 2021.

[42] N. Papernot et al. Practical black-box attacks against machine learning. In *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pages 506–519, 2017.

[43] P. Patel, W. Barr, P. Dutton, A. Little, B. Blair, India, and Japan. Internation statement: End-to-end encryption and public safety. https://www.gov.uk/government/publications/international-statement-end-to-end-encryption-and-public-safety, Oct 2020. Accessed on June 6, 2021.

[44] Python Core Team. Python: A dynamic, open source programming language. https://www.python.org/, 2021.

[45] J. C. S. Reis, P. Melo, K. Garimella, and F. Benevenuto. Can whatsapp benefit from debunked fact-checked stories to reduce misinformation? *Harvard Kennedy School Misinformation Review*, 2020.

[46] K. Ren, T. Zheng, Z. Qin, and X. Liu. Adversarial attacks and defenses in deep learning. *Engineering*, 6(3):346–360, 2020.

[47] O. Russakovsky et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[48] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv:1703.03864*, 2017.

[49] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 acm sigsac conference on computer and communications security*, pages 1528–1540, 2016.

[50] X. Shen, S. Diamond, Y. Gu, and S. Boyd. Disciplined convex-concave programming. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 1009–1014. IEEE, 2016.

[51] Y. Shi, S. Wang, and Y. Han. Curls & whey: Boosting black-box adversarial attacks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6519–6527, 2019.

[52] M. Singh. Whatsapp is now delivering roughly 100 billion messages a day. https://social.techcrunch.com/2020/10/29/whatsapp-is-now-delivering-roughly-100-billion-messages-a-day/, Oct 2020. Accessed on June 6, 2021.

[53] M. Singh. Signal's brian acton talks about exploding growth, monetization and whatsapp data-sharing outrage. https://social.techcrunch.com/2021/01/12/signal-brian-acton-talks-about-exploding-growth-monetization-and-whatsapp-data-sharing-outrage/, Jan 2021. Accessed on June 6, 2021.

[54] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv:1312.6199*, 2013.

[55] Thorn. Benchmarking. https://perception.thorn.engineering/en/latest/examples/benchmarking.html. Accessed on June 6, 2021.

[56] Thorn. Testing different image hash functions. https://content-blockchain.org/research/testing-different-image-hash-functions/. Accessed on June 8, 2021.

[57] P. Virtanen et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[58] G. K. Wallace. The jpeg still picture compression standard. *IEEE Transactions on Consumer Electronics*, 38(1):xviii–xxxiv, Feb 1992.

[59] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber. Natural evolution strategies. *The Journal of Machine Learning Research*, 15(1):949–980, Jan 2014.

[60] Q. Xiao, Y. Chen, C. Shen, Y. Chen, and K. Li. Seeing is not believing: Camouflage attacks on image scaling algorithms. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 443–460, 2019.

[61] K. Yang, J. Yau, L. Fei-Fei, J. Deng, and O. Russakovsky. A study of face obfuscation in imagenet. *arXiv:2103.06191*, 2021.

[62] C. Zauner. Implementation and benchmarking of perceptual image hash functions. Master's thesis, 2010. https://www.phash.org/docs/pubs/thesis_zauner.pdf.

[63] C. Zauner, M. Steinebach, and E. Hermann. Rihamark: perceptual image hash benchmarking. In *Media watermarking, security, and forensics III*, volume 7880. International Society for Optics and Photonics, 2011.

[64] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

A Proofs for the DCT analysis

Proposition 1. *$AA^T = I_{c^2}$, where A is defined as follows:*

$$\begin{gathered}A \in \mathbb{R}^{c^2 \times k^2}\\ A_{c_1\times c+c_2, k_1\times k+k_2} = M'_{1+c_1,1+k_1} \times M'_{1+c_2,1+k_2}\\ \textit{for } 0 \le c_1, c_2 \le c-1, 0 \le k_1, k_2 \le k-1\end{gathered}$$

Proof. Let $c = c_1 \times c + c_2$ and $c' = c'_1 \times c + c'_2$. We develop $(AA^T)_{c,c'}$ as follows:

$$\begin{aligned}(AA^T)_{c,c'} &= \sum_{k_1=0}^{k-1}\sum_{k_2=0}^{k-1} A_{c_1\times c+c_2,k_1\times k+k_2} \times A_{c'_1\times c+c'_2,k_1\times k+k_2}\\ &= \sum_{k_1=0}^{k-1}\sum_{k_2=0}^{k-1} M'_{1+c_1,1+k_1} \times M'_{1+c_2,1+k_2} \times M'_{1+c'_1,1+k_1} \times\\ &\times M'_{1+c'_2,1+k_2}\\ &= (\sum_{k_1=0}^{k-1} M'_{1+c_1,1+k_1} \times M'_{1+c'_1,1+k_1})\times\\ &\times (\sum_{k_2=0}^{k-1} M'_{1+c_2,1+k_2} \times M'_{1+c'_2,1+k_2})\\ &= (M'M'^T)_{1+c_1,1+c_2} \times (M'M'^T)_{1+c'_1,1+c'_2}\end{aligned}$$

Because $M'M'^T = I_c$, it follows that $(A\,A^T)_{c,c'} = 1$ if $c = c'$ and 0 otherwise. Indeed, $c = c'$ if and only if $c_1 = c'_1$ and $c_2 = c'_2$, meaning that the terms being multiplied in the last equality can both be non-zero only when $c = c'$, in which case they are both equal to 1. □

Proposition 2. *The eigenvalues of A^TA are 1 with multiplicity c^2 and 0 with multiplicity $k^2 - c^2$.*

Proof. Let λ be an eigenvalue of A^TA and $x \neq 0$ such that $A^TAx = \lambda x$. We multiply by A to the left and obtain $\lambda Ax = A(A^TAx) = (AA^T)Ax = Ax$. We distinguish two cases: (1) $Ax = 0$, which implies that $\lambda = 0$ and (2) $Ax \neq 0$, which implies that $\lambda = 1$.

Therefore the eigenvalues of AA^T can only be 0 or 1.

We denote by $m(\lambda)$ the multiplicity of eigenvalue λ.

It follows from the above analysis that $dim(Ker(A)) \le m(0)$ and that $rank(A) \le m(1)$. By summation we obtain $k^2 \le m(0) + m(1) = k^2$, therefore $m(1) = rank(A)$.

We show that the rank of A is c^2. Indeed $rank(A) \le min(c^2, k^2) = c^2$. If it were the case that $rank(A) < c^2$, then there would be $x \in \mathbb{R}^{c^2}$ such that $x^TA = 0$ but $x \neq 0$. Since $AA^T = I_{c^2}$, by multiplying with A^T on the right we obtain $x^T = 0$, a contradiction.

We thus conclude that $m(1) = c^2$ and $m(0) = k^2 - c^2$. □

B Visual similarity

Fig. 9 shows the perceptual similarity between the original and modified images as quantified by the Learned Perceptual Image Patch Similarity distance [64] evaluated using AlexNet [29].

C Prevalence rate

Table 3 shows the number of wrongly detected images for different perceptual hashing algorithms and threshold, for a prevalence rate of 10^{-7} for illegal content among 4.5B images shared daily. The results are comparable in order of magnitude to those of Table 2 in the main paper.

D Impact of diversity on visual similarity of perturbed images

Fig 10 shows that our modifications to black-box algorithm leads to more diverse perturbations, but it also leads to relatively higher $\mathbb{L}_2$ perturbation per pixel compared to black-box algorithm without modifications. However we also observe that even for the highest thresholds, 99.9% of diverse perturbed images generated for dHash (respectively 100.0%, 80.4%, 91.8% and 65.2% for pHash continuous, pHash, aHash and PDQ) of perturbed images are within the $\mathbb{L}_2$ perturbation per pixel of 0.13.

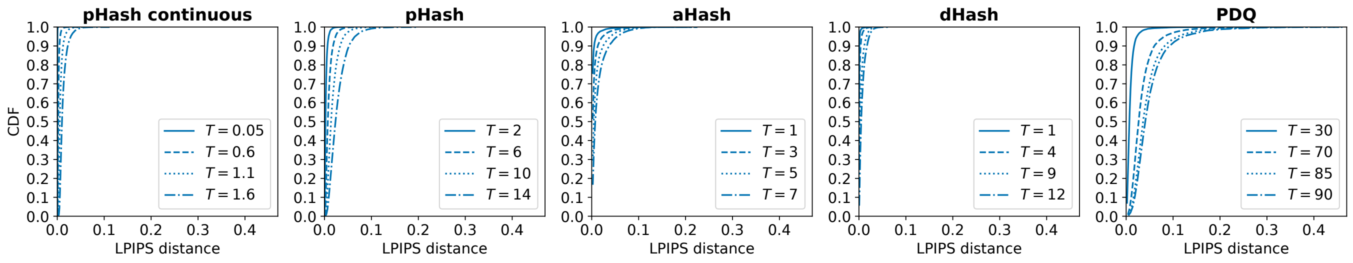

Figure 9: Cumulative distribution function (CDF) for the LPIPS distance between the original and the perturbed image for different algorithms and thresholds T, and all successfully attacked images over 10 experiments. A smaller distance indicates higher visual similarity between the modified and original images. The distances increase slowly with the threshold but remain small in all cases.

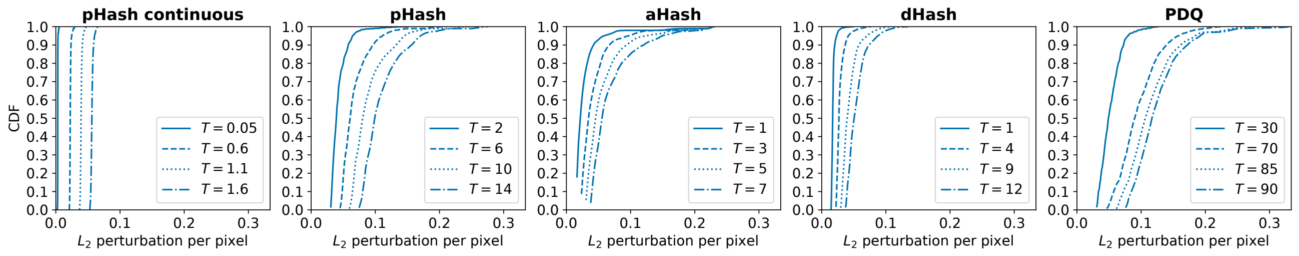

Figure 10: Pairwise distances between hashes of the multiple perturbations generated for the same image. The number of perturbations generated for each image is $D = 50$ and the number of images attacked is $N' = 100$.

Hash	**Rough estimate of the number of wrongly detected images daily / Threshold T**			
pHash continuous	$\sim$ 5M $T = 0.05$	$\sim$ 23M $T = 0.6$	$\sim$ 184M $T = 1.1$	$\sim$ 665M $T = 1.6$
pHash	$\sim$ 5M $T = 0.2$	$\sim$ 7M $T = 6$	$\sim$ 108M $T = 10$	$\sim$ 3.3B $T = 14$
aHash	$\sim$ 44M $T = 1$	$\sim$ 179M $T = 3$	$\sim$ 498M $T = 5$	$\sim$ 1.1B $T = 7$
dHash	$\sim$ 5M $T = 1$	$\sim$ 8M $T = 4$	$\sim$ 112M $T = 9$	$\sim$ 798M $T = 12$
PDQ	$\sim$ 5M $T = 30$	$\sim$ 7M $T = 70$	$\sim$ 135M $T = 85$	$\sim$ 1.6B $T = 90$

Table 3: Rough estimate of the number of images that would be wrongly detected daily on WhatsApp alone for different perceptual hashing algorithms and thresholds. We here consider a database size of $N = 100$k, 4.5B images being shared daily [5], and a prevalence rate of illegal content of 10^{-7}.

E Probability to flag k images

We consider a simple model with two types of users each sending N images independently from one another. The first type, which we call non-offender, sends only non-illegal images. The second type, which we call offender, sends l illegal images that are adversarially modified using our attack and $N - l$ non-illegal images. The probability for the detection system to flag an image is FPR for non-illegal images and $1 - FNR$ for illegal images adversarially modified using our attack.

Under this model, the number of images flagged for a non-offender follows a binomial distribution $N_1 \sim Bin(N, FPR)$. Similarly, the number of images flagged for an offender can be written as the sum of two binomial random variables $N_2 \sim Bin(l, 1 - FNR) + Bin(N - l, FPR)$. The two follow the same distribution if $1 - FNR = FPR$, i.e., if the adversarial images are indistinguishable from natural images to the detection system.

This allows us to compute, for each type of user, the probability that $0 \leq k \leq N$ of their images are flagged:

$$P(N_1 = k) = \binom{N}{k} FPR^k (1 - FPR)^{N-k}$$

$$P(N_2 = k) = \sum_{i=0}^{min(l,k)} \binom{l}{i} (1 - FNR)^i FNR^{l-i} \times \binom{N-l}{k-i} FPR^{k-i} (1 - FPR)^{N-l-k+i}$$