An Overview of Perceptual Hashing

Hany Farid

Abstract. It is said that what happens on the internet stays on the internet, forever. In some cases this may be considered a feature. Reports of human rights violations and corporate corruption, for example, should remain part of the public record. In other cases, however, digital immortality may be considered less desirable. Most would agree that terror-related content, child sexual abuse material, non-consensual intimate imagery, and dangerous disinformation, to name a few, should not be so easily found online. Neither human moderation nor artificial intelligence is currently able to contend with the spread of harmful content. Perceptual hashing has emerged as a powerful technology to limit the redistribution of multimedia content (including audio, images, and video). We review how this technology works, its advantages and disadvantages, and how it has been deployed on small- to large-scale platforms.

1 Introduction

In March 2019, a terrorist killed 51 people and injured another 40 at two mosques in Christchurch, New Zealand. This attack was the deadliest shooting in modern New Zealand history. In addition to sharing a 74-page hate-filled, anti-immigrant manifesto on Twitter and 8chan, the terrorist livestreamed the attack on Facebook. Both the video and manifesto were subsequently banned in New Zealand and Australia.

According to Facebook, the livestream was watched by fewer than 200 viewers (Sonderby 2019). In the 24 hours following the attack, however, Facebook removed 1.5 million videos of the attack, 1.2 million of which were blocked at upload (it remains unclear how long the other 300,000 remained online or how many times they were viewed). The video also circulated on other platforms including YouTube, Reddit, and Twitter. YouTube, for example, saw an unprecedented volume of re-upload attempts, peaking at one attempt per second (YouTubeInsider 2019).

Two months after the attack, New Zealand's Prime Minister Jacinda Ardern initiated the Christchurch Call to Action Summit, which aimed to "bring together countries and tech companies in an attempt to bring to an end the ability to use social media to organise and promote terrorism and violent extremism" (Ardern 2019). World leaders from 48 countries,[1] along with technology companies—including Facebook, Google, Microsoft, Twitter, and YouTube—pledged to eliminate terrorist and violent extremist content online.

1. The United States, under President Trump, declined to attend the summit. In May 2021, President Biden announced the United States would join the Christchurch Call.

doi:10.54501/jots.v1i1.24

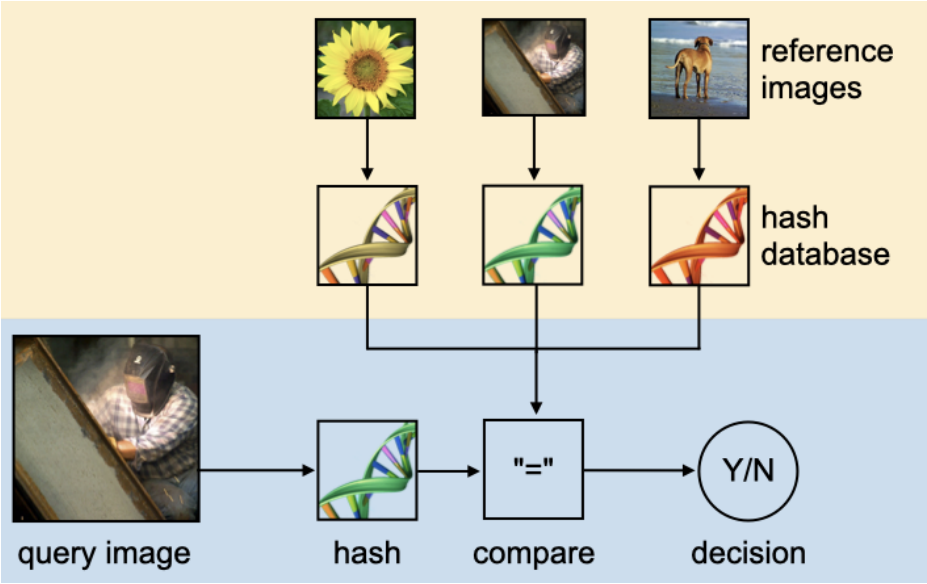

Figure 1: An overview of image hashing consisting of two primary stages: (1) an offline stage (upper) in which a hash is extracted from each offending reference image (sunflower, welder, dog); and (2) an online stage (lower) in which a hash is extracted from a query image (welder) at the point of upload and compared against all database hashes—any matched image is blocked, reviewed, reported, or subjected to whatever policy a service provider initiates.

More than two years after the attack and subsequent global call to action, videos of the horrific Christchurch attack are still available on Facebook and other video-sharing platforms (Solon 2019). In the years since the March 2019 attack, social media has continued to be weaponized by extremist groups including, for example, the radicalization, planning, and execution of the January 6, 2021, insurrection at the US Capitol.

These incidents highlight the challenges faced by online trust and safety teams in contending with real-world violence stemming from online behaviors. Platforms face additional challenges when grappling with a myriad of other abuses, including child sexual abuse material (CSAM), illegal sex trade, illegal drugs and weapons, dangerous disinformation and conspiracies, and more. Here we will focus on technologies to limit the redistribution of various forms of illegal, dangerous, and hateful content, with a focus on multimedia content (audio, image, and video).

Hashing is arguably the most widely used technology for limiting the spread of previously identified multimedia content. This technology is used to contend with everything from terror-related content to CSAM (Farid 2018a) and copyright infringement. After an offending audio, image, or video is identified (either manually or automatically), a distinct digital signature is extracted from the content (the digital signature is often referred to as a fingerprint, message digest, hash value, hash code, hash sum, or, simply, hash). The same hash is extracted from each future upload and compared against a database of offending hashes (see Figure 1). Any matched content can, for example, be automatically blocked from upload or subjected to any policy a service provider initiates.

The terms hash and hashing originated from the culinary term describing the process of chopping a mixture of ingredients to yield, for example, a hash of potatoes, peppers, and tomatoes. In the digital realm, hashing refers to chopping a data file into small pieces and combining them to yield a concise numeric value that can be used to identify the original

data file. Although we are discussing hashing in the context of the modern internet, the importance of hashing dates back to the early years of the electronic computer, the process having first been described in 1953 (Luhn 1953), with the term hashing first appearing in 1967 (Hellerman 1967).

In the broadest use of the term, there are many applications of hashing, including efficient data storage and retrieval, data integrity, and data matching. We will focus on this latter use, and in particular the application of hashing to digital media files.

In order to be most effective at internet scale with billions of daily uploads, an effective hash should have several properties:

1. **Distinct.** The extracted hashes for two distinct pieces of content should themselves be distinct; this ensures that innocuous content is not incorrectly matched.
2. **Resilient.** The extracted hash for a piece of content should be resilient to simple content modifications that do not fundamentally alter the underlying content; this ensures that perceptually similar but not identical pieces of content will be matched.
3. **Deterministic.** The extracted hash for a piece of content is always the same; this ensures that a piece of content can always be identified based on its hash.
4. **Efficient.** The computational cost of extracting a hash and comparing hashes should be low; this ensures the technology can be deployed on even the largest social media sites.
5. **Non-reversible.** It should be practically impossible to reconstruct the original content from only the hash; this allows for hashes of illegal content (e.g., CSAM) to be shared across different organizations.

Different flavors of hashing comply with one or more of the above five criteria, but not all five. Hard-hashing techniques, for example, are highly distinct but not resilient to even the smallest change in the underlying media file (hard hashing is also referred to as fingerprinting). In contrast, perceptual hashing can be resilient but is less distinct (perceptual hashing is also referred to as robust hashing, fuzzy hashing, or content-based image retrieval [CBIR]). In the next two sections, we will discuss these two classes of hashing: how they work, their advantages and disadvantages, and how they have been deployed to counter online extremism, CSAM, non-consensual intimate imagery, and copyright infringement.

2 Hard Hashing

The most widely used hard-hashing algorithms are arguably MD5 (Rivest 1992) and SHA-1 (Eastlake and Jones 2001), along with their variants (MD2, MD4, MD6, and SHA-0, SHA-2, SHA-3). The MD5 (message digest) algorithm takes as input an arbitrary data file and generates a 128-bit hash, typically represented as a sequence of 32 hexadecimal characters (hexadecimal corresponds to a base 16 alphanumeric system consisting of the values `0–9` and `a–f` [or `A–F`]). A simple text file containing the text "An Overview of Perceptual Hashing," for example, yields the MD5 hash `7b2cc72c5a57c2d6878a9d897b21e9cd`. The MD5 algorithm first partitions the data file into 512-bit blocks, from which a per-block hash is computed by manipulating the data in each block alongside the hash from the previous block. The final hash is generated in the final data block.

MD5 was initially designed as a cryptographic hash with additional security features beyond the basic hash features described above, including that a message cannot be

generated to yield a specific hash and that a small change in the data file leads to a large and uncorrelated change in the hash (the so-called avalanche effect). Although these cryptographic properties were found not to hold (Wang and Yu 2005), MD5 hashing continues to be used for basic data integrity checks and data matching.

Shown below is the MD5 hash of five text files, each containing a single line of text.

```
MD5("Perceptual")                          = 4fcd92bbcf33b0b4a1eac76a673e1f33
MD5("Perceptual Hashing")                  = c6eedea9b3db1e5ff9e80153dc629dcc
MD5("An Overview of Perceptual Hashing") = 7b2cc72c5a57c2d6878a9d897b21e9cd
MD5("an Overview of Perceptual Hashing") = 6df033f24a309e7eca6c70834d34bd2a
MD5("xn Overview of Perceptual Hashing") = 9a87d6c846de4f2bc35a40e42bd1d4a8
```

Notice first that regardless of the size of the input data file, the resulting hash consists of 32 hexadecimal characters. Notice also that even small differences in the input file (as in the last three entries) lead to large and seemingly uncorrelated changes in the hash. As we will see later, this latter property has both advantages and disadvantages when it comes to identifying a piece of content based on a hash.

Designed by the United States National Security Agency, SHA-1 (secure hash algorithm) takes as input an arbitrary data file and generates a 160-bit hash, typically represented as a sequence of 40 hexadecimal characters (Eastlake and Jones 2001). A simple text file containing the text "An Overview of Perceptual Hashing," for example, yields the SHA-1 hash `b1bf46c0f558cce2ae0a23f2aa7c509e7f91301e`. Like the MD5 hash, the SHA-1 hash is of fixed size and exhibits the avalanche effect. Also like MD5, SHA-1 was initially designed as a cryptographic hash. Although SHA-1 was found not to be cryptographically secure (Wang, Yin, and Yu 2005), it continues to be used for basic data integrity checks and data matching.

Both MD5 and SHA-1, among others, have been used by various platforms to identify a specific digital media file. Shown in Figure 1 is the basic structure for this type of media matching. In an offline stage (upper portion of figure), a hash is extracted from each offending image (or video, audio, or any other data file) and stored for future comparison; then, in an online stage (lower portion of figure), a hash is extracted from each piece of uploaded content and compared against all database hashes, after which any matched content is blocked, reviewed, reported, or subjected to whatever policy a service provider initiates.

Because even the smallest change in the data causes a large change in the resulting hash, the only meaningful comparison of two hashes is an exact match for equality between their hexadecimal characters. The advantage of this property is that, because the extracted hashes for two distinct pieces of content are themselves distinct, it is exceedingly unlikely that one piece of uploaded content will be mistakenly identified as another piece of content. The disadvantage of this property is that only exact content matches can be discovered.

Returning to the video of the Christchurch attack, MD5 or SHA-1 hard hashing would have been effective at blocking the exact same video file from re-upload, but not its numerous variants that underwent refilming, editing, or splicing. Even something as simple as removing metadata, modifying a single pixel, or adding or dropping a single frame will render hard hashing ineffective at content matching. This, again, has both advantages and disadvantages: hard hashing is exceedingly unlikely to incorrectly block content, but it is also easy to circumvent, either intentionally or unintentionally.

While any service provider should use hard hashing to limit redistribution of banned or illegal content, this technology itself is easy to circumvent. Perceptual hashing provides an additional and important layer of protection because it is, by design, able to detect

many of the variants that hard hashing misses.

3 Perceptual Image Hashing

A digital image is made up of an array of pixels (picture elements). Each pixel is itself composed of three values corresponding to the primary red, green, and blue colors (RGB). In a typical digital image, each 8-bit pixel takes on a value from 0 to 255. A bright, pure red pixel, for example, is represented by the triple values (255,0,0), a dim, pure blue pixel by (0,0,64), a black pixel by (0,0,0), a mid-level gray pixel by (128,128,128), and a pure white pixel by (255,255,255). A modest 12-megapixel digital sensor produces a digital image of size 3000×4000 pixels. With a total of 256 possible values per pixel for each of three color channels, this sensor can produce a mind-boggling number of different images, exceeding the some 10^{24} estimated stars in the universe.

The goal of perceptual hashing is to mimic the human visual system's assessment of comparing two images based on the underlying scene content, as compared to a purely numeric comparison based on the pixel values. This is accomplished by extracting from the massive pixel-space representation a concise, distinct, perceptually meaningful signature that is resilient to modifications of the image, including compression, color shifts, cropping, rotation, the addition of a logo or overlain text, or any other modification that does not fundamentally change the underlying content but does alter the underlying pixel values. The extraction and comparison of a perceptual hash must also be efficient so that it can operate on billions of daily uploads.

Version 1. Consider the following strawman perceptual image hash consisting of the average pixel value, rounded to the nearest integer, in each of the three RGB color channels. The hash for the image of the welder in Figure 1, for example, is (92, 88, 73). As with MD5 and SHA-1, this hash can alternatively be represented as a fixed-size sequence of hexadecimal characters, computed by converting each decimal number to hexadecimal (`5c`, `58`, `49`) and concatenating to yield `5c5849`.

A pair of hashes extracted from two images can be compared for equality by comparing each of the three color channel averages (or their hexadecimal representation). The welder image, for example, with hash (92, 88, 73), is clearly distinct from the sunflower image in Figure 1, with hash (155,153,17), because all three values are significantly different.

The comparison of the triple of values in this simple hash can be combined into a single metric as follows. Define the hash for two images to be $\vec{h}_1 = (r_1, g_1, b_1)$ and $\vec{h}_2 = (r_2, g_2, b_2)$ (the $\vec{\cdot}$ notation reminds us we are dealing with a vector-valued quantity consisting of two or more values). The standard Euclidean distance—the shortest distance between the pair of 3-D points represented by the hashes $\vec{h}_1$ and $\vec{h}_2$—is given by

$$d \quad = \quad \sqrt{(r_1 - r_2)^2 + (g_1 - g_2)^2 + (b_1 - b_2)^2}, \tag{1}$$

yielding a single value d that quantifies the difference between the two hashes $\vec{h}_1$ and $\vec{h}_2$. Note that if the hashes are identical ($r_1 = r_2$, $g_1 = g_2$, and $b_1 = b_2$), then the difference is 0. As the individual parts of the hash diverge, the difference d increases.

In addition to collapsing the comparison of the triple of values to a comparison of a single value, this Euclidean metric allows for the comparison not just for equality but also for near equality. If, for example, our reference hash is (92, 88, 73) as compared to a query hash of (91, 88, 73), then the difference between these hashes is only 1 intensity step (on a scale of 0 to 255). Such a small difference might result from an innocuous manipulation

such as image resizing or recompression. In order to catch these variants, two hashes can be said to match if their distance is less than a small threshold, $d < \tau$, where, if the threshold $\tau = 0$, then a strict equality is respected. A larger τ is desirable as it makes the hash more resilient, but at the cost of reducing distinctiveness. There is, therefore, an inherent tension between distinctiveness and resilience that must be balanced. The proper balance depends on a service's tolerance for the accuracy of correctly matching all image variants (true positive) and incorrectly matching two distinct images (false positive), each of which can be balanced with the choice of τ: a small value of τ decreases the true positive rate, but also decreases undesirable false positives; a larger value of τ increases the true positive rate, but also increases false positives.

Given a database of N reference hashes, a query hash can be compared, one at a time, against each reference hash, stopping when the computed distance is less than a specified threshold τ. The computational efficiency of this database comparison will grow linearly with the size of the database N: double the size of the database to $2N$, and the computational cost will also double. Because each individual comparison will be relatively fast (typically on the order of milliseconds), this cost may not be prohibitive. As the database grows into the many millions, however, the computational cost of billions of daily full database comparisons may become more significant.

This cost can be alleviated by storing the reference hashes in a kd-tree (Jon Louis 1975) (or similar type of data structure), affording an efficient search of all N entries with only $\log(N)$ comparisons: double the size of the database from N to $2N$, and the computational cost will only increase by one additional comparison (not N additional comparisons as with a linear search). With such a slow growth in computational cost, the hash database can grow rapidly with little impact on the overall computational cost to the entire hashing pipeline.

Let's now consider this hashing technique in terms of the five desirable properties of a hash, enumerated in the Introduction:

1. **Distinct.** The extracted hashes for two distinct pieces of content should themselves be distinct. This property does not hold: composed of three values, each of which can take on one of 256 values, there are only a total of $256^3 = 16,777,216$ unique possible hashes, a woefully small number as compared to the universe of possible images, meaning that it is guaranteed this hash will not be unique.

2. **Resilient.** The extracted hash for a piece of content should be resilient to simple content modifications that do not fundamentally alter the underlying content. This property does not hold: this hash is resilient to, for example, a 90-degree rotation of the image, but is not resilient to a simple hue shift of the image, which changes the average pixel value in each color channel. Similarly, this hash is not resilient to the addition of a red border, which increases the average pixel value in the red channel.

3. **Deterministic.** The extracted hash for a piece of content is always the same. This property holds: the computation of the average of pixel values is itself deterministic.

4. **Efficient.** The computational cost of extracting a hash and comparing hashes should be low. This property holds: it is computationally trivial to compute the average pixel value in each color channel and compute the Euclidean distance between two hashes, as in Equation (1).

5. **Non-reversible.** It should be practically impossible to reconstruct the original content from only the hash. This property holds: because any combination of the same pixels in an image will yield the same average value, there is no way of inferring the precise pixel pattern from only an average value.

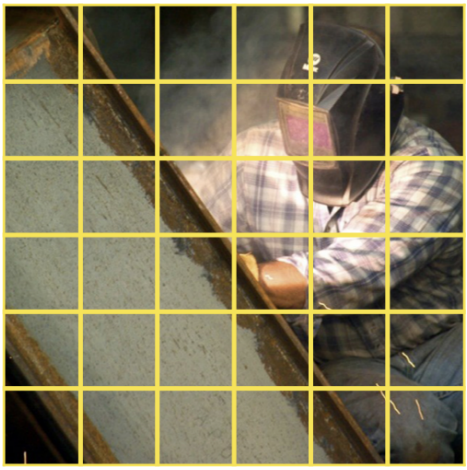

Figure 2: The image of the welder is resized to 300 $\times$ 300 pixels and partitioned into 36 non-overlapping, 50 $\times$ 50 pixel blocks. A partial hash is extracted from each of these blocks and combined to yield the image hash.

Because of the lack of distinctiveness and resilience, this hash is not suitable, but it has provided us with a starting point to consider improvements.

One goal of hashing is to reduce the representation of an image from its (typically) millions of pixels to a more concise representation. In our first iteration, each color channel was reduced to a single value. With only some 16 million possible hashes, this collapsing to a single value per color channel was perhaps too aggressive.

Version 2. In this second iteration of a hash, consider the partitioning of an image into 6 $\times$ 6, non-overlapping blocks, as in Figure 2 (an image can be initially resized to a square aspect ratio of size divisible by six [e.g., 300 $\times$ 300 pixels] to ensure an even partitioning). From each of these 36 blocks, the same set of three average pixel values—again, converted to integers—from each RGB color channel can be extracted. The full hash is constructed by concatenating all 36 triple values into a single hash consisting of 36 $\times$ 3 = 108 values. Composed now of 108 values, each of which can take on one of 256 values, there are a staggering $256^{108} = 1.23 \times 10^{260}$ unique possible hashes. With so many possible hashes, we at least have a chance of obtaining distinctiveness.

The inspiration for this type of local, block-based partitioning as compared to, for example, rows, columns, or concentric circles is that many stages of the human visual system operate in a similar fashion (Wandell 1995). In particular, from receptive fields in the retina to the first processing stages in the visual cortex, the early stages of the human visual system are defined by this type of spatially localized information processing.

Even though this block-based hash is longer, the same Euclidean distance metric can be applied to two hashes $\vec{h}_1$ and $\vec{h}_2$ (each now a 108-D vector, as compared to the previous 3-D vector):

$$d = \sqrt{d_1 + d_2 + \cdots + d_{36}}, \tag{2}$$

where the i^{th} block-based squared distance d_i, $i \in [1, 36]$, is

$$d_i = (r_{1,i} - r_{2,i})^2 + (g_{1,i} - g_{2,i})^2 + (b_{1,i} - b_{2,i})^2, \tag{3}$$

and where $(r_{1,i}, g_{1,i}, b_{1,i})$ is the block-based hash for the i^{th} block of the reference image, and $(r_{2,i}, g_{2,i}, b_{2,i})$ is the block-based hash for the i^{th} block of the query image. As before,

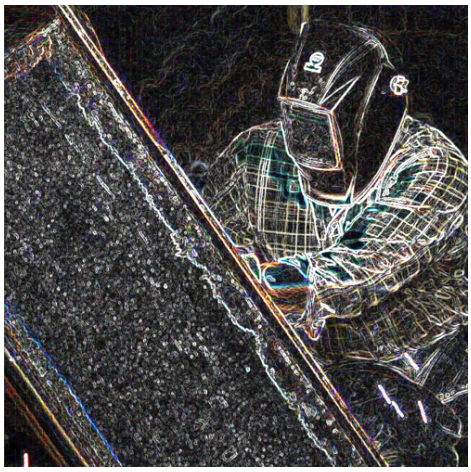

Figure 3: The gradient of an image emphasizes perceptually meaningful parts of an image corresponding to large transitions in color or brightness.

the distance between two hashes can be compared for equality $d = 0$, or near equality $d < \tau$.

Let's now consider the five desirable properties of this new hash:

1. **Distinct.** This property holds, somewhat: with a significantly larger universe of possible hashes exceeding 1.23×10^{260}, this hash promises more distinctiveness than the first iteration with a mere 1.67×10^6 unique hashes. At the same time, however, it is likely that, for example, two different landscape images, with a blue sky in the upper half of the image and green rolling hills in the lower half, will have similar hashes and will be incorrectly matched within a non-zero tolerance τ.
2. **Resilient.** This property does not hold: this hash is not resilient to a simple hue shift of the image, which changes the average pixel value in each color channel. Similarly, because the individual hash components are extracted from a fixed grid partitioning of the image, this hash will not be resilient to a cropping or shifting of the image that results in a misalignment of the original grid.
3. **Deterministic.** This property holds: the computation of the average block-based pixel values is itself deterministic.
4. **Efficient.** This property holds: it is computationally efficient to resize the image, compute the block-based average pixel value in each color channel, and then compare the resulting 108 values, as seen in Equations (2)-(3).
5. **Non-reversible.** This property mostly holds: because the hash is composed of the average color value from each of 36 blocks, the hash is itself a low-resolution, 6 $\times$ 6 pixel version of the image. At this resolution, however, there is little identifiable content to be extracted from the hash.

Overall, this second iteration improved on the first with better distinctiveness, but still suffers from resilience and has given up a little in terms of non-reversibility. Because pixel values can be so easily manipulated while retaining perceptual similarity, relying only on simple measurements (e.g., pixel average) extracted from the underlying RGB pixel values is unlikely to yield a resilient hash.

Version 3. Shown in Figure 3 is a filtered version of the welder image. This image is computed by differencing neighboring pixels in the horizontal direction (image rows) and

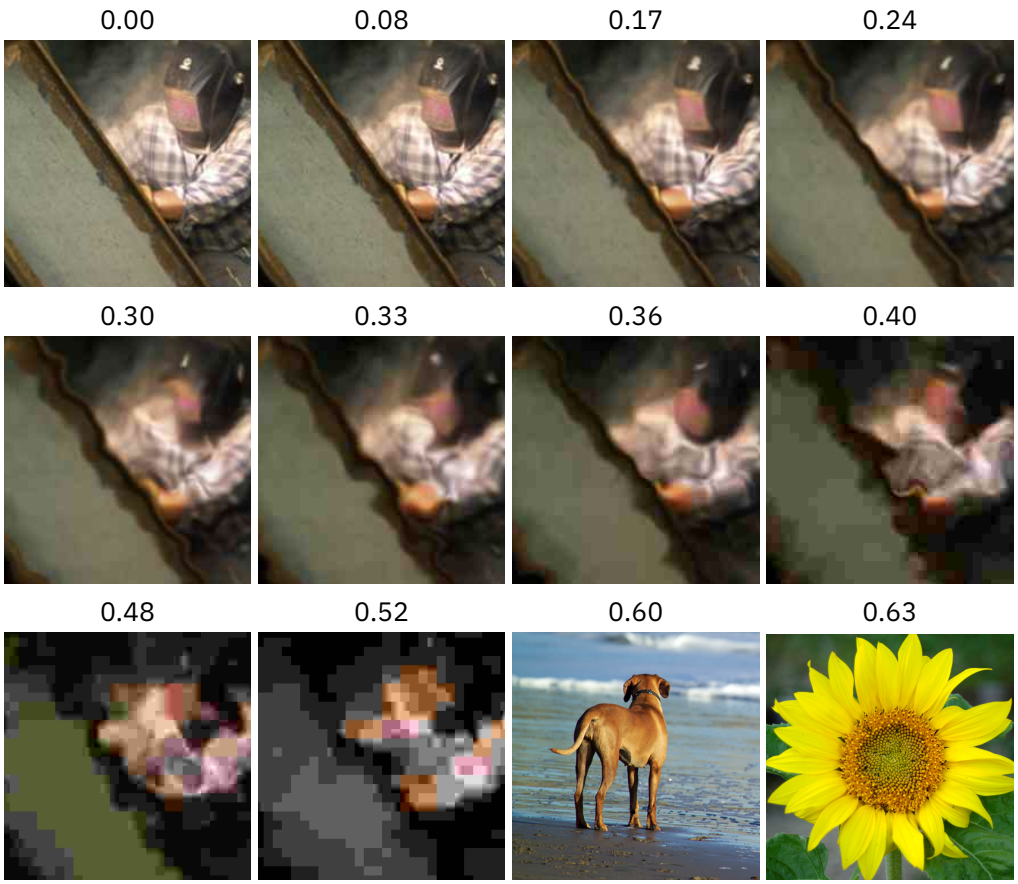

Figure 4: Ten versions of the original welder image (top left) with increasing levels of photometric and geometric distortions. Shown above each image is the numeric hash distance relative to the original (the larger the hash distance, the more dissimilar the images). By way of comparison, the distance of two unrelated images (lower right) is larger than the even most distorted versions of the original.

in the vertical direction (image columns), then squaring each of these difference images and summing them together. In this filtered image, the transitions (edges) between bright and dark regions are emphasized (brighter pixels) while uniform color patches are de-emphasized (darker pixels).

A perceptual hash can be extracted from this filtered image by computing the average gradient value in each color channel for each 6 $\times$ 6 non-overlapping block. As with the previous hash, this yields a hash consisting of 36 $\times$ 3 = 108 values. Alternatively, the RGB image can initially be converted to a grayscale image (gray = 0.299R + 0.587G + 0.114B). The final hash can then be extracted from this now single-channel image, yielding a hash with only 36 values. Because the gradient information across color channels is typically highly correlated, operating on only a grayscale image improves computational efficiency without a significant impact on distinctiveness. By again treating two such hashes as 108-D (color) or 36-D (grayscale) vectors, the Euclidean distance between two hashes provides a measure of similarity between the underlying images.

Shown in Figure 4 are ten versions of the welder image with increasing levels of photometric and geometric distortions. Shown above each image is the Euclidean distance

between a gradient-based hash extracted from each image and the hash from the original undistorted image in the top left panel. Shown in the last two panels are two unrelated images. Even the two most distorted welder images in the last row are more similar to the original than these distinct images, showing the resilience and distinctiveness of this type of perceptual hashing.

With respect to the five desirable properties, this new hash is more **distinct** than the color-based hashes because the image gradient captures more distinct scene structure; it is also **resilient** to simple color changes, but not other manipulations like cropping or rotation. Because the image filtering is itself deterministic and efficient, this new hash remains **deterministic** and **efficient**. Because of the low resolution of the blocks, the initial image filtering, and extraction of only summary statistics, the **non-reversible** property mostly holds.

Most perceptual hashes, consisting of three basic parts, follow the same pattern as those described above: (1) preprocessing (e.g., image resizing, image gradient); (2) hash extraction (e.g., average color or gradient); and (3) hash comparison (e.g., Euclidean distance). The many choices for each of these stages result in varying compromises between the five desired properties of a hash (see Smeulders et al. 2000; Liu et al. 2007; Du, Ho, and Cong 2020 for extensive surveys of different techniques).

The most typical compromises are between distinctiveness and resilience, and the specific manipulations to which the hash is resilient. A hard hash, for example, is highly distinct but exhibits no resilience to even a small content change. At the other extreme, a global hash based on the average per-channel pixel color with a threshold of τ=255 is highly resilient but utterly non-distinct. Similarly, some hashes are resilient to color and brightness changes but not necessarily to geometric changes like rotation and cropping. In practice, the choice of a hash is based on a number of factors, including:

1. **Scale.** When operating at the scale of a major social media platform, for example, with billions of daily uploads, any hash must be highly efficient and distinct. At this scale, even a 1/100 or even 1/10,000 false positive rate (incorrectly matching two images) is untenable.
2. **Tolerance.** When trying to limit the upload of, for example, legal adult pornography, resilience may be less important than, for example, trying to limit child sexual abuse imagery (CSAM).
3. **Security.** When trying to limit the upload of copyright-infringing material, for example, non-reversibility may be less critical than in other more sensitive domains.

We will discuss, in the Implementations section, several real-world deployments of perceptual hashing along with their relative advantages and disadvantages.

4 Perceptual Video Hashing

A digital video typically consists of between 24 and 30 images (frames) per second. Each frame is recorded in the same format as described in the previous section. The simplest extension of perceptual hashing from images to videos is to hash each reference video frame using any perceptual image hash of choice. On upload, each query video frame is then hashed and compared against all reference video frames. If a specified number of frames are matched, then the video can be considered a match. The number of required matching frames can be configured to control the balance between distinctiveness and resilience: a large number of required frames increases distinctiveness, whereas a smaller threshold increases resilience.

The advantage of this approach is that it requires no new infrastructure or implementation beyond a basic perceptual image hashing implementation. There are, however, two disadvantages of this approach. First, at 30 frames per second (fps), even a short three-minute video consists of 3 min $\times$ 60 sec/min $\times$ 30 frames/sec = 5,400 individual frames, each of which would have to be hashed and compared against the full database. Even with an efficient image hashing algorithm and the improved computational efficiency of a kd-tree for hash comparison, a frame-based video hash database will rapidly grow in size, with each video adding thousands to potentially hundreds of thousands of individual hashes. Second, by considering only a single frame at a time, we are not taking advantage of the distinctive nature embedded in the temporal ordering of frames; doing so would almost certainly improve the distinctiveness of any perceptual hash.

Because successive frames in a video are typically highly similar, the computational inefficiency of hashing every video frame can be alleviated by sampling the video in time. If, for example, only every 30th frame was selected and hashed (for an effective frame rate of 1 fps), then a three-minute video would yield only 3 min $\times$ 60 sec/min $\times$ 1 frame/sec = 180 individual frames, down from 5,400 at the full 30 fps. The advantage of this approach is, again, that it requires no new infrastructure, and the added computational complexity at this reduced frame rate is more manageable, though still somewhat costly as the number of reference videos increases. The disadvantage remains that the distinctive temporal order of frames is not being leveraged.

Successive frames of most videos are highly similar because of the relatively high temporal sampling rate of a typically stationary camera imaging an often static or slowly changing scene. We can, therefore, further reduce the complexity of analyzing a video by reducing this frame-to-frame redundancy beyond sampling every 30th (or more) frames.

Perceptual image hashing, conveniently, provides a mechanism for measuring the similarity between two successive frames. Redundancy in a video can be eliminated by using perceptual image hashing to eliminate similar successive frames. Starting with the first frame, each successive frame is dropped from consideration if the perceptual image hash is less than a generous threshold τ; otherwise, the frame is retained. A perceptual image hash is then extracted from each of the remaining key-frames and concatenated to yield a final video hash. Unlike the image-based hashing that yields a fixed-length hash, this video hash can be of arbitrary length. This presents both a challenge and an opportunity for comparing two hashes.

A Euclidean distance cannot be used to compare two hashes of different length. Instead, the longest common substring (LCS) (Gusfield 1997) can be used to compare two variable-length hashes. By way of intuition, the LCS of the two strings `ABABACABBC` and `ABACABACBBCA` is six because the longest common string shared by these strings is `ABACAB`. Note that these strings also have the substring `BBC` in common, but this is shorter than the substring of length six. The advantage of using LCS to compare two hashes is that it allows us to find not just matching videos but also shorter video segments or video compilations where video segments are embedded within a larger video. Because each component of the final video hash is itself an image hash, it can be compared using the same Euclidean distance metric, allowing for both an exact or near frame-by-frame match. A second threshold can be applied to the length of the longest common substring before triggering a match.

Running on a standard Linux machine, our Java-based implementation of this perceptual video hashing requires approximately 10 milliseconds (ms) to process a single video frame and approximately 2.5 ms to compare two hashes (Farid 2018a). To improve efficiency, a multicore version of this algorithm allows for a video to be partitioned into

an arbitrary number of short segments, each of which can be analyzed on a separate computer core. The individual results from each segment can then be combined to create a single hash. With this approach, the rate-limiting step to analyze any video is simply the number of available computing cores.

When implementing any frame-based video hash, care must be taken not to add to the reference hash database parts of the video that are innocuous or nondescript. Many videos, for example, append generic opening and closing credits and may create fade ins/outs, where a small number of frames will be uniformly black or white. These parts of any reference video must be removed prior to extracting the perceptual video hash.

As with perceptual image hashes, most perceptual video hashes consist of the same three basic steps: (1) preprocessing; (2) hash extraction; and (3) hash comparison. The many choices for each of these stages result in varying compromises between the five desired properties of a hash.

Although we won't discuss perceptual audio hashing here, audio files are hashed following the same basic structure as images and video, and are an important addition to any perceptual hashing system (Özer et al. 2005). Perceptual audio hashing would have been, for example, a powerful tool to contend with a myriad of variants of the Christchurch terror video in which the underlying audio was similar, even if the visual appearance was not.

5 Implementations

The academic literature is home to a rich source of different perceptual hashing techniques. At the same time, a number of these techniques have been deployed on a range of online services, for a range of different applications. In this section, we will review some of these techniques. Because some of these techniques are proprietary and/or releasing algorithmic details may threaten their efficacy, we will not delve into the specific algorithmic details but instead simply review the various techniques and how and where they are deployed.

In 2007, YouTube deployed Content ID (originally called Video Identification) (YouTube) after facing threats of litigation and massive monetary fines for not doing enough to prevent—and for profiting from—the sharing of copyrighted materials. This perceptual audio/video hashing extracts a distinct signature from owner-supplied content. When flagged at upload, content owners can choose to block, track views, or monetize the content.

By 2008, facing an explosion in the spread of child sexual abuse material (U.S. Senate 2007), the leading technology companies had yet to develop similar technology to contend with this horrific content. Led by Microsoft, a perceptual image hash—PhotoDNA (Microsoft)—was developed, tested, and deployed in 2009 on Microsoft's Bing search engine and on what was formerly called SkyDrive cloud service. In 2010, Facebook deployed PhotoDNA on their services. In 2011, Twitter followed suit, while Google waited until 2016 to deploy. In addition to these titans of tech, PhotoDNA is now in worldwide deployment across a range of platforms. As a measure of PhotoDNA's efficacy, shown in Figure 5 are the number of yearly reports submitted to the National Center for Missing and Exploited Children's (NCMEC) CyberTipline (a single report may consist of tens to hundreds of individual images or videos). According to NCMEC, a major contributor to the more than 100× increase in reports over the past decade has been the adoption by service providers of perceptual hashing tools like PhotoDNA (Bursztein et al. 2019).

PhotoDNA was designed to perceptually hash images because, at the time, digital video

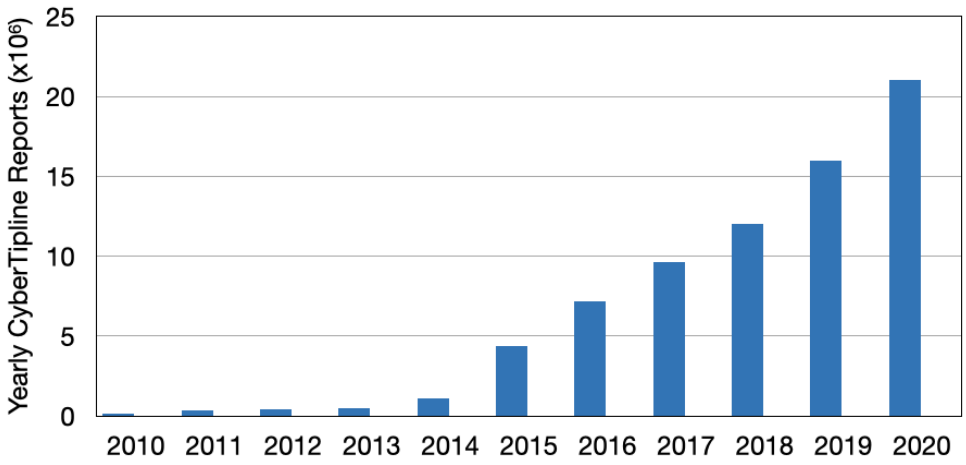

Figure 5: Yearly CSAM reports to NCMEC's CyberTipline. From 2010 to 2020, the number of yearly reports jumped from slightly more than 100,000 to over 20,000,000.

had not yet become ubiquitous. In the intervening decade, videos have come to dominate many online platforms. In 2016, largely in response to the growing threat of international and domestic extremism, the Counter Extremism Project (CEP) developed eGlyph (CEP 2016), an audio, image, and video perceptual hash. In an alternate application to content moderation, the CEP used eGlyph to determine the prevalence on YouTube of ISIS-generated, terror-related videos (Project 2018; Farid 2018b). Over a twelve-week period starting in early 2018, CEP used YouTube's API to analyze videos associated with a simple keyword search (e.g., Caliphate, Jihad, Flames of War, etc.), and compared the resulting videos against a video-hash database of ISIS videos previously identified as violating YouTube's terms of service. With a hash database of only 229 videos, CEP found a total of 1,348 ISIS videos uploaded to YouTube, garnering a total of 163,391 views. These videos were uploaded by 278 different YouTube accounts, one of which uploaded as many as 50 videos before the channel was either removed by YouTube administrators or deleted by the owner. More than 90% of the identified videos were uploaded and remained online long enough to be found, revealing a failure on the part of YouTube to proactively scan for previously identified terror-related content. As this application showed, perceptual hashing can also be used to verify the effectiveness of a service provider's content-moderation tools.

In response to a rise in online extremism and terrorism, the Global Internet Forum to Counter Terrorism (GIFCT, https://gifct.org/) was founded in 2016 by Facebook, Microsoft, Twitter, and YouTube to "foster technical collaboration among member companies, advance relevant research, and share knowledge with smaller platforms." As part of this mission, the GIFCT has created an image and video hashing database of identified terrorism materials that it shares with industry partners.

In 2019, Facebook announced an open-source image and video hashing, PDQ (with the non-obvious root "a Perceptual algorithm utilising a Discrete Cosine Transform and outputting [amongst others] a Quality metric") and TMK+PDQF (Temporal Match Kernel + PDQ with Floating-point operations) (Davis 2019; Facebook; Dalins, Wilson, and Boudry 2019). Inspired by the perceptual hash pHash (Zauner 2010), PDQ follows the same pattern of other perceptual image hashes described in the Perceptual Image Hashing section:

1. **Preprocessing.** Transform the image (resize to a smaller resolution, convert from RGB to luminance, and compute the image gradient).
2. **Hash extraction.** The hash is extracted from measurements made from a block-based, discrete cosine transform (DCT) to yield a 256-bit (64 hexadecimal characters).
3. **Hash comparison.** Two hashes are compared using a Hamming distance, measuring the minimum number of substitutions needed to change one string of characters into another. The video hash TMK+PDQF begins by first temporally resampling the video to a reduced 15 frames per second and extracting a per-frame hash using PDQF, a modified version of PDQ. The final hash is computed from different combinations of these frame-based hashes.

In August of 2021, Apple announced the development of NeuralHash for detecting known CSAM in images stored in Apple's iCloud (Apple 2021a). This technology was scheduled to roll out in iOS 15 and iPadOS 15, but was delayed due to pressure from privacy groups. Unlike PhotoDNA that maps perceptually similar images to similar, but not necessarily identical, hashes, NeuralHash converts each image to a unique numeric value. This unique image-to-hash mapping, along with private set intersection (Apple 2021b)—a secure technique for database comparisons—allows for a more privacy-preserving hash database comparison. As compared to PhotoDNA, which performs the image hashing and comparison upon upload to a server, Apple's NeuralHash performs the image hashing and comparison directly on the device. This client-side comparison allows for a more privacy-respecting hashing, as it does not reveal the image contents to the server unless a match is found upon upload to iCloud. Apple reports a 1 in 1 trillion likelihood of incorrectly matching a user's account; manual review further mitigates the chance of a false report.

Unlike PhotoDNA, eGlyph, and NeuralHash, PDQ and TMK+PDQF are open source. There are advantages and disadvantages to both approaches. Open-source software allows a larger research community to more easily contribute to advancing a technology. On the other hand, if the inner workings of the hashing technology are revealed, an adversary can more easily circumvent detection.

Lastly, TinEye (tineye.com) and Google's reverse image search (images.google.com) offer powerful internet-scale tools for searching for perceptually similar images. These tools can be useful in larger investigations of the source of an image.

6 End-to-End Encryption

Following a trend in other messaging apps, Facebook recently announced plans to move all their messaging services to an end-to-end encrypted system (E2EE) (Facebook 2021). This move would prevent anyone, including Facebook, from seeing the content of any personal communication.

While E2EE provides users with some added privacy, the associated risks are not insignificant. In announcing his plans, Mark Zuckerberg conceded it came at a cost: "At the same time, there are real safety concerns to address before we can implement end-to-end encryption across all of our messaging services. Encryption is a powerful tool for privacy, but that includes the privacy of people doing bad things. When billions of people use a service to connect, some of them are going to misuse it for truly terrible things like child exploitation, terrorism, and extortion" (Zuckerberg 2019).

In 2020, for example, the NCMEC's CyberTipline received over 21 million reports of

apparent CSAM (see Figure 5). Over 20 million of these reports were generated by Facebook's various services (National Center for Missing and Exploited Children 2020), the vast majority of which were generated by automatic perceptual hashing technologies. If fully implemented, an E2EE system would render these technologies ineffective. All perceptual image and video hashing technologies described to this point require the receiving service provider to have direct access to the underlying media. Within a fully E2EE system, content passing through the provider remains encrypted and inaccessible to PhotoDNA and the like.

In this section we consider how perceptual hashing can be deployed within an E2EE system. Any such system will require a compromise between privacy and security. No system will be perfect, and we must choose solutions that find the appropriate balance.

Traditional perceptual hashing operates on media as it passes through a service provider's system (so-called server-side hashing). In an E2EE system, the media is only accessible by the sender and receiver, with no intermediary access. One relatively obvious solution, therefore, would be to extract the hash on the sender's device (so-called client-side hashing), and transmit this hash alongside the fully encrypted message. The service provider can then analyze the hash against a database of offending content. If the hash is non-reversible, this approach somewhat preserves privacy (for added security, the hash can also be encrypted—but not end-to-end encrypted—so only the service provider with the proper credentials can decrypt). Apple's NeuralHash is an example of this form of client-side hashing, where both the hashing and database comparison are performed entirely on a user's device. The drawback of this approach is that software for extracting the hash must be deployed on billions of devices, creating a potential vulnerability in which the hashing algorithm is reverse-engineered and circumvented. This threat can be partially mitigated by converting the original media into an intermediate representation that is non-reversible but not the complete hash, and then completing, server-side, the hash computation and comparison. In this approach, neither the full hashing algorithm nor the full media content is revealed. Another drawback of this client-side hashing is that, beyond having indirect knowledge of a specific image/video being transmitted, the contents of the message remain encrypted and inaccessible. This could pose challenges for legal investigations or prosecutions.

A second approach uses an encryption scheme that allows for limited computations on the encrypted data, without the need or even ability to decrypt the data. We will refer to decrypted data as "plaintext" and the resulting encrypted data as "ciphertext." Homomorphic encryption schemes (Naehrig, Lauter, and Vaikuntanathan 2011) (specifically, partially homomorphic encryption schemes) have two intriguing mathematical properties: (1) the decrypted product of two ciphertexts is equal to the sum of their corresponding plaintexts; and (2) the decrypted exponentiation of a ciphertext by a scalar value is equal to the product of the scalar and corresponding plaintext. If a perceptual hash, in plaintext, can be extracted and compared by only summing pixel values together and multiplying a pixel value by a scalar multiple, then this perceptual hash can be implemented in the encrypted ciphertext (see, for example, Singh and Farid 2019). The advantage of this approach is that, unlike the client-side hashing, this hashing can be performed server-side. A disadvantage is that the cost of ciphertext computations is currently significantly higher than their plaintext counterparts.

Beyond contending with CSAM and other illegal content, homomorphic perceptual hashing can be used in other domains. In 2017, for example, Facebook announced a trial program to combat the growing problem of nonconsensual intimate imagery (often referred by the misnomer "revenge porn") (Facebook 2017). Those wishing to prevent re-uploads of their intimate imagery were asked to send the offending images to Face-

book, who would in turn extract a perceptual hash from this content and block future uploads. Although well intentioned, this program was met with some skepticism. At a time when social media was (and remains) under scrutiny for their failure to protect user data and privacy, it seemed unreasonable to ask users to trust Facebook with some of their most personal and intimate images. If homomorphic encryption and hashing is used, then instead of directly sending intimate images, the victim need only extract a homomorphic perceptual hash directly on their device. This hash can then be securely shared with a service provider. A drawback of this approach is that trolls could easily upload any image they wish to be blocked, so this approach would have to be paired with a single manual review upon the first hash match to verify the image in question is in fact nonconsensual of the reporting complainant.

Client-side hashing or homomorphic encryption can allow for perceptual hashing to continue to operate within an E2EE system. Neither of these solutions, however, contend with the larger issue of lawful access. Even with a lawful warrant, for example, access to past or future communications is not possible within an E2EE messaging services. Some argue this is necessary to protect user privacy, while others argue that any messaging service must allow for lawful access for investigations and prosecutions.

In a third alternative, a secure enclave allows for software to be executed on a remote, otherwise untrusted computer, while providing some integrity and confidentiality guarantees (Costan and Devadas 2016; Ferraiuolo et al. 2017). With specialized, trusted hardware, a remote computer creates a secure container, after which a remote user uploads the desired computation and data into this container. By design, the trusted hardware protects the computational integrity and data confidentiality. Although not a true E2EE system—because the enclave itself has access to the decrypted data—this type of hardware has many attractive properties that allow for a reasonable balance between privacy and security.

7 Discussion

With billions of daily uploads, content moderation on the largest social media platforms has proven exceedingly challenging. A combination of user-generated content ranging from the illegal to the dangerous and hateful, alongside algorithms amplifying this content because it drives user engagement, has resulted in news feeds and watch lists awash with the internet's flotsam and jetsam. As a result, we are plunged into increasingly angry, isolating, and dangerous echo chambers. While the past decade may have been good for a relatively small number of technology company shareholders, it has been decidedly not good for many individuals, societies, and democracies.

There are many challenges in mitigating harm stemming from our online communities, ranging from the scale and speed with which content is uploaded and spread, to the tensions between the desire for an open and free internet balanced against individual and public safety. Perceptual hashing should be a central tool of any content moderation system. These techniques, however, must be paired with other technologies and human moderation. While perceptual hashing can mitigate the harm of redistribution of everything from CSAM to terror-related content, new illegal and harmful content is, sadly, created every day. Contending with these ever-evolving threats requires additional technologies to analyze text, audio, images, and video for potentially illegal or harmful content. Instead of demanding perfection from these technologies, we should view them as triage technologies that filter the billions of uploaded content to a more manageable volume for human review. More care, however, must also be taken to consider the impact on human moderators, many of whom report PTSD-like symptoms along with a

host of other mental health issues after spending their days viewing the dredges of the internet (Arsht and Etcovitch 2018).

Although well established, perceptual hashing technologies are not without their detractors. It has been argued that this type of systematic content scanning can lead to abuse by totalitarian governments or overreach by service providers. For example, some argue that even if justified in the case of CSAM, such technologies should not be deployed in other arenas—for example, terrorism and hate speech—in which illegal or inappropriate content may be more difficult to define. While there is merit to the argument that some content can be difficult to categorize, there is an abundance of content that is clearly illegal, dangerous, or obviously violative of a provider's terms of service. A benefit of perceptual hashing, unlike more opaque and uninterruptible AI classifiers, is that the contents of any hash database can be audited for appropriateness. A second benefit is the highly targeted nature of hashing, as compared to AI classifiers with unpredictable generalizability. To this end, deployment of perceptual hashing should be paired with a clear specification of the criteria under which content is added to a hash database, along with periodic audits of the database. At the Canadian Centre for Child Protection (https://protectchildren.ca), for example, each piece of content is triple verified by independent expert examiners before its addition to the Centre's hash database.

As with any inherently adversarial relationship, perceptual hashing is vulnerable to counterattack by those seeking to avoid detection. Some attacks can be countered by simply augmenting the hash database. If, for example, a hash is not resilient to simple geometric transformations of an image (e.g., 90-degree rotation, or XY-flipping), then these transformed images can simply be hashed and added to the hash database. Other attacks, however, may be more challenging to counter. A combination of multiple hashes, each with their own distinct vulnerabilities, can be deployed to make counterattack more difficult.

Returning to the New Zealand Christchurch terror video, several points of failure contributed to the Christchurch, US Capitol, and other terror attacks over the past decade, including online radicalization, distribution, and redistribution.

Radicalization: Every day, four petabytes (more than four million gigabytes) of data are uploaded to Facebook. But not all of this content is equal in the eyes of Facebook. Starting in 2009, Facebook eliminated the ability to chronologically sort the News Feed, turning over editorial control to algorithmic curation. Similarly, every minute of every day, more than 500 hours of video are uploaded to YouTube. The likelihood that any video is widely seen, however, depends largely on recommendation algorithms. A full 70% of watched YouTube footage is recommended by the company's algorithms (Faddoul, Chaslot, and Farid 2020). By 2016, Twitter and Instagram joined Facebook and YouTube in unleashing attention-grabbing recommendation algorithms to control what we read, see, hear, and, ultimately, believe. In 2018, Facebook's internal research found "Our algorithms exploit the human brain's attraction to divisiveness." The research went on to conclude that, if left unchecked, Facebook's recommendation algorithms will promote "more and more divisive content in an effort to gain user attention and increase time on the platform" (Horwitz and Seetharaman 2020). A separate internal Facebook study found that 64% of people who joined an extremist group on Facebook did so because of the company's recommendation. Put simply, the recommendation algorithms have learned that users respond to outrage and sensational material, and have therefore learned to promote this material to maximize user engagement and, in turn, ad revenue.

Distribution: Although 200 viewers watched the livestream of the Christchurch attack, nobody reported it. Facebook had various automatic AI-based content moderation tools, but they failed to detect this particular video because it did not match the pattern

previously seen by Facebook's AI system: "AI systems are based on 'training data', which means you need many thousands of examples of content in order to train a system that can detect certain types of text, imagery or video. This approach has worked very well for areas such as nudity, terrorist propaganda and also graphic violence where there is a large number of examples we can use to train our systems. However, this particular video did not trigger our automatic detection systems. To achieve that we will need to provide our systems with large volumes of data of this specific kind of content, something which is difficult as these events are thankfully rare." (Rosen 2019). As this incident showed, even modern AI tools and tens of thousands of human moderators are not necessarily sufficient to disrupt even the most heinous acts from being shared on social media.

Redistribution: Facebook's systems also failed to fully prevent the redistribution of variants of the livestreamed attack. In order to prevent redistribution, a digital signature (hash) was extracted from the original video and its various screen-recorded variants. Future uploads could then be compared to these hashes in order to automatically prevent re-upload. This technology, however, failed to contend with the large number of variants: "We've been asked why our image and video matching technology, which has been so effective at preventing the spread of propaganda from terrorist organizations, did not catch those additional copies. What challenged our approach was the proliferation of many different variants of the video, driven by the broad and diverse ways in which people shared it: First, we saw a core community of bad actors working together to continually re-upload edited versions of this video in ways designed to defeat our detection. Second, a broader set of people distributed the video and unintentionally made it harder to match copies" (Rosen 2019). Once again, this incident highlighted the challenges with reducing both sharing and redistribution of content.

While much has and should be written about algorithmic amplification leading to online radicalization (Neumann 2013) and content-moderation tools for automatically detecting problematic and illegal content (Gillespie 2020), we have focused here only on technologies to limit redistribution of previously identified content. Even though these technologies have proven to be reliable and effective, they have not been developed, refined, and deployed with sufficient energy or sense of urgency proportional to the depth and breadth of online harms that have emerged over the past decade. More can and should be done to develop and deploy increasingly more distinct, resilient, and secure perceptual hashes. More can and should be done to create industry-shared databases of illegal, dangerous, and hateful content. And, more can and should be done to insist that all online service providers are held to a higher standard when it comes to the offline harms that result from their often laissez-faire attitude towards online content moderation.

References

Apple. 2021a. *CSAM Detection Technical Summary.* https://www.apple.com/child-safety/pdf/CSAM_Detection_Technical_Summary.pdf.

———. 2021b. *The Apple PSI System.* https://www.apple.com/child-safety/pdf/Apple_PSI_System_Security_Protocol_and_Analysis.pdf.

Ardern, Jacinda. 2019. *NZ and France seek to end use of social media for acts of terrorism.* https://www.beehive.govt.nz/release/nz-and-france-seek-end-use-social-media-acts-terrorism.

Arsht, Andrew, and Daniel Etcovitch. 2018. "The human cost of online content moderation." *Harvard Law Review Online.*

Bursztein, Elie, Einat Clarke, Michelle DeLaune, David M Elifff, Nick Hsu, Lindsey Olson, John Shehan, Madhukar Thakur, Kurt Thomas, and Travis Bright. 2019. "Rethinking the detection of child sexual abuse imagery on the Internet." In *The World Wide Web Conference,* 2601–2607.

Costan, Victor, and Srinivas Devadas. 2016. "Intel SGX explained." *IACR Cryptol. ePrint Arch.* 2016 (86): 1–118.

Counter Extremism Project CEP. 2016. *How CEP's eGLYPH Technology Works.* https://www.counterextremism.com/video/how-ceps-eglyph-technology-works.

Dalins, Janis, Campbell Wilson, and Douglas Boudry. 2019. *PDQ & TMK+ PDQF–A Test Drive of Facebook's Perceptual Hashing Algorithms.* arXiv: 1912.07745.

Davis, Antigone. 2019. *Open-Sourcing Photo- and Video-Matching Technology to Make the Internet Safer.* https://about.fb.com/news/2019/08/open-source-photo-video-matching.

Du, Ling, Anthony TS Ho, and Runmin Cong. 2020. "Perceptual hashing for image authentication: A survey." *Signal Processing: Image Communication* 81:115713.

Eastlake, Donald, and Paul Jones. 2001. *US secure hash algorithm 1 (SHA1).*

Facebook. 2017. *The Facts: Non-Consensual Intimate Image Pilot.* https://about.fb.com/news/h/non-consensual-intimate-image-pilot-the-facts.

———. 2021. *Messenger Policy Workshop: Future of Private Messaging.* https://about.fb.com/news/2021/04/messenger-policy-workshop-future-of-private-messaging.

———. *The TMK+PDQF Video-Hashing Algorithm and the PDQ Image-Hashing Algorithm.* https://github.com/facebook/ThreatExchange/blob/master/hashing/hashing.pdf.

Faddoul, Marc, Guillaume Chaslot, and Hany Farid. 2020. *A Longitudinal Analysis of YouTube's Promotion of Conspiracy Videos.* arXiv: 2003.03318.

Farid, Hany. 2018a. "Reining in online abuses." *Technology & Innovation* 19 (3): 593–599.

———. 2018b. *Verifying #BigTech promises.* https://www.eureporter.co/frontpage/2018/05/11/verifying-bigtech-promises/.

Ferraiuolo, Andrew, Andrew Baumann, Chris Hawblitzel, and Bryan Parno. 2017. "Komodo: Using verification to disentangle secure-enclave hardware from software." In *26th Symposium on Operating Systems Principles,* 287–305.

Gillespie, Tarleton. 2020. "Content moderation, AI, and the question of scale." *Big Data & Society* 7 (2): 2053951720943234.

Gusfield, Dan. 1997. "Algorithms on stings, trees, and sequences: Computer science and computational biology." *ACM SIGACT News* 28 (4): 41–60.

Hellerman, Herbert. 1967. *Digital computer system principles.* McGraw-Hill Companies.

Horwitz, Jeff, and Deepa Seetharaman. 2020. *Facebook Executives Shut Down Efforts to Make the Site Less Divisive.* https://www.wsj.com/articles/facebook-knows-it-encourages-division-top-executives-nixed-solutions-11590507499.

Jon Louis, Bentley. 1975. "Multidimensional binary search trees used for associative searching." *Communications of the ACM* 18 (9): 509–517.

Liu, Ying, Dengsheng Zhang, Guojun Lu, and Wei-Ying Ma. 2007. "A survey of content-based image retrieval with high-level semantics." *Pattern Recognition* 40 (1): 262–282.

Luhn, Hans Peter. 1953. "A new method of recording and searching information." *American Documentation* 4 (1): 14–16.

Microsoft. *photoDNA.* https://www.microsoft.com/en-us/photodna.

Naehrig, Michael, Kristin Lauter, and Vinod Vaikuntanathan. 2011. "Can homomorphic encryption be practical?" In *3rd ACM workshop on Cloud Computing Security Workshop,* 113–124.

National Center for Missing and Exploited Children. 2020. *2020 Reports by Electronic Service Providers.* https://www.missingkids.org/gethelpnow/cybertipline.

Neumann, Peter R. 2013. "Options and strategies for countering online radicalization in the United States." *Studies in Conflict & Terrorism* 36 (6): 431–459.

Özer, Hamza, Bülent Sankur, Nasir Memon, and Emin Anarım. 2005. "Perceptual Audio Hashing Functions." *EURASIP Journal on Advances in Signal Processing* 2005 (12): 1–14.

Project, Counter Extremism. 2018. *The eGlyph Web Crawler: ISIS Content on YouTube.* https://www.counterextremism.com/sites/default/files/eGLYPH_web_crawler_white_paper_July_2018.pdf.

Rivest, Ronald. 1992. *The MD5 message-digest algorithm.*

Rosen, Guy. 2019. *A Further Update on New Zealand Terrorist Attack.* https://about.fb.com/news/2019/03/technical-update-on-new-zealand.

Singh, Priyanka, and Hany Farid. 2019. "Robust Homomorphic Image Hashing." In *Workshop on Media Forensics at CVPR,* 11–18.

Smeulders, Arnold WM, Marcel Worring, Simone Santini, Amarnath Gupta, and Ramesh Jain. 2000. "Content-based image retrieval at the end of the early years." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22 (12): 1349–1380.

Solon, Olivia. 2019. *Six months after Christchurch shootings, videos of attack are still on Facebook.* https://www.nbcnews.com/tech/tech-news/six-months-after-christchurch-shootings-videos-attack-are-still-facebook-n1056691.

Sonderby, Chris. 2019. *Update on New Zealand.* https://about.fb.com/news/2019/03/update-on-new-zealand.

U.S. Senate. 2007. *H.R.3845 - Providing Resources, Officers, and Technology to Eradicate Cyber Threats to Our Children Act of 2007.* https://www.congress.gov/bill/110th-congress/house-bill/3845/text.

Wandell, Brian. 1995. *Foundations of Vision.* Oxford University Press.

Wang, Xiaoyun, Yiqun Lisa Yin, and Hongbo Yu. 2005. "Finding collisions in the full SHA-1." In *Annual international cryptology conference,* 17–36. Springer.

Wang, Xiaoyun, and Hongbo Yu. 2005. "How to break MD5 and other hash functions." In *Annual international conference on the theory and applications of cryptographic techniques,* 19–35. Springer.

YouTube. *Using Content ID.* https://support.google.com/youtube/answer/3244015.

YouTubeInsider. 2019. https://twitter.com/YouTubeInsider/status/1107645354361741312.

Zauner, Christoph. 2010. *Implementation and benchmarking of perceptual image hash functions.*

Zuckerberg, Mark. 2019. *A Privacy Focused Vision for Social Networking.* https://www.facebook.com/notes/mark-zuckerberg/a-privacy-focused-vision-for-social-networking/10156700570096634/.

Authors

Hany Farid is a Professor in Electrical Engineering & Computer Sciences and the School of Information at the University of California, Berkeley.

Data Availability Statement

Not applicable

Funding Statement

Not applicable

Ethical Standards

Not applicable

Keywords

content moderation; perceptual hashing; robust hashing; fuzzy hashing; PhotoDNA