Universal Adversarial Head: Practical Protection against Video Data Leakage

Jiawang Bai 1  Bin Chen 1  Dongxian Wu 1  Chaoning Zhang 2  Shu-Tao Xia 1

Abstract

While online video sharing becomes more popular, it also causes unconscious leakage of personal information in the video retrieval systems like deep hashing. An adversary can collect users’ private information from the video database by querying similar videos. This paper focuses on bypassing the deep video hashing based retrieval to prevent information from being maliciously collected. We propose universal adversarial head (UAH), which crafts adversarial query videos by prepending the original videos with a sequence of adversarial frames to perturb the normal hash codes in the Hamming space. This adversarial head can be obtained just using a few videos, and mislead the retrieval system to return irrelevant videos on most natural query videos. Furthermore, to obey the principle of information protection, we expand the proposed method to a data-free paradigm to generate the UAH, without access to users’ original videos. Extensive experiments demonstrate the protection effectiveness of our method under various settings.

1. Introduction

As an era of big data arrives, massive videos are being uploaded to the Internet, e.g., YouTube, TikTok, and Twitter. For instance, the users upload more than 500 hours of video to YouTube every minute (Robertson) and more than 1 billion videos get viewed each day on TikTok (G). However, this might cause unconscious leakage of personal information by some content-based retrieval systems. For example, utilizing the advanced face video retrieval techniques (Qiao et al., 2020; Wang et al., 2020), a snoop holding a user’s face video as the query can maliciously collect more personal information from online shared videos. The potential threat of the video data leakage flow is shown in Figure 1. Traditional methods for video privacy protection hide the privacy information by masking (Wickramasinghe et al., 2004) or scrambling (Dufaux & Ebrahimi, 2008) the protected objects, which degrade the video quality and can not be applied to the online shared videos. Therefore, it is imperative to design a practical solution to prevent video data leakage in the retrieval system, without affecting users’ experience.

Among video retrieval systems, the hashing method is promising due to its low storage cost and high search efficiency (Liong et al., 2016). It maps semantically similar videos to similar compact binary codes in the Hamming space by the hashing function. Along with the development of deep learning, deep hashing has become a mainstream video retrieval algorithm that achieves state-of-the-art retrieval performance (Li et al., 2019; Yuan et al., 2020). However, if it were exploited maliciously, the information leakage concern for the video data would be intensified. Therefore, we explore information protection on deep video hashing in this paper.

Figure 1. Threat of the video data leakage and the expected protection effectiveness. Relevant videos in the database can be maliciously collected by querying with the original video, while the collection utilizing the modified video fails. Best viewed in color.
1. Specifically, the modification method should achieve two
objectives: which meets usability requirement mentioned in
2. the proposed universal adversarial head.

To address the above concern, this paper studies how to by-
pass the deep video hashing with slight changes to the query
video. In other words, the modification to the video has
little impact on normal usages (e.g., sharing online), but can
mislead the retrieval system to return the irrelevant videos,
so as to avoid malicious information collection. We demon-
strate the expected effectiveness of the protection in Figure
1. Specifically, the modification method should achieve two
goals: usability and stealthiness. The usability requires
that the modification can be applied to almost all videos in
the same manner without extra computation. It is especially
important for practicality since benign users can adopt our
technique directly on their videos. It alleviates the need for
them to share their videos or generate such modifications
specifically for their videos. The stealthiness requires that
the modification should not affect the normal usages of the
modified videos. For example, the quality of the video can
not be largely decreased, so that it can be normally shared
between benign users.

We accomplish the protection through a benign usage of the
adversarial attack (Szegedy et al., 2014; Kurakin et al., 2017;
Bai et al., 2020). The adversarial attack perturbs normal
samples by intentional small perturbations and fools Deep
Neural Networks (DNNs) to make incorrect predictions
confidently, which has been well-studied in the machine
learning security (Akhtar & Mian, 2018; Bai et al., 2021).
It is usually regarded as a threat to deep learning, especially
the universal attack. The universal attack calculates a uni-
versal perturbation based on only a few inputs and causes
almost all inputs to be misclassified with high probability
(Moosavi-Dezfooli et al., 2017; Benz et al., 2020; 2021;
Zhang et al., 2020a; 2021). Inspired by this, we utilize the
universal attack against deep video hashing as a modification
method for information protection, which ensures usable-
ness. However, prior works of the universal attack could
not meet the stealthiness requirement, since they mainly
considered the perturbations on original samples, which
partly decrease the quality of the modified videos. There-
fore, we propose to prepend a universal adversarial head
(UAH), as shown in Figure 2, that consists of a sequence
of clean frames (dubbed clean head) with an adversarial
perturbation. The content of the clean head can be speci-
fied flexibly. Although such an adversarial head is visible,
the UAH does not look suspicious because many videos
uploaded to the online video platforms may contain the clip
of welcome, introduction, or copyright information at the
beginning. Since there is no perturbation on the original
frames, the UAH avoids affecting the quality of the videos.
In addition, the superior performance of the UAH as demon-
strated in our experiments also motivates us to apply the
adversarial perturbation on the prepended head.

Once the clean head is specified, the remaining problem
is to generate an imperceptible adversarial perturbation of
the UAH. Utilizing some original videos, we optimize the
objective, which is to enlarge the distance between the hash
codes of the original videos and those of the videos with the
UAH. However, in the context of information protection for
the video data, we may not get access to the original videos
from the users. Therefore, we further explore to generate the
UAH in a more challenging and practical data-free scenario
(Mopuri et al., 2017; 2018). Extensive experiments verify
the effectiveness of our method under various settings.

2. The Proposed Method

2.1. Preliminaries

Let \( x \in \mathbb{R}^{M \times H \times W \times C} \) denote a given query video sampled
from a distribution \( p \), where \( M \) is the number of frames,
and \( H, W \) and \( C \) are the height, width and the number of
channels for each frame, respectively. \( f_\theta(\cdot) \) is a non-linear
function parameterized by \( \theta \), which transforms the input
video \( x \) into a \( K \)-dimensional representation. A sign activa-
tion function is then used for binarizing the \( K \)-dimensional
representation into a \( K \)-bit hash code \( h \in \{-1, +1\}^K \). We
formulate the previous video hashing model \( F(\cdot) \) as follows:

\[
    h = F(x) = \text{sign}(f_\theta(x)).
\]

The most common choice for \( f_\theta(\cdot) \) is the powerful feature
extractor-pair with CNN+RNN (Liong et al., 2016) fram-
work, i.e., features extracted by a CNN are fed into an RNN.
The hashing model \( F(\cdot) \) generates a binary hash code for
each video in the database and the query video. To retrieve
the semantically similar videos, the similarity is measured
by Hamming distance between the hash code of each video
in the database and that of the query video. Then the re-
trieval system returns a sorted list of videos according to
these computed Hamming distances.

The main focus of this paper is to seek a modification \( T(\cdot) \)
on the original query video \( x \) to generate an adversarial
video \( T(x) \), which can bypass the deep video hashing model
\( F(\cdot) \). For the universal adversarial attack, \( T(\cdot) \) is applied
to all videos sampled from a distribution \( p \) in the same
manner, which meets usableness requirement mentioned in
Universal Adversarial Head: Practical Protection against Video Data Leakage

the introduction. Moreover, $T(x)$ can induce the hashing retrieval system to generate a hash code which is close to semantically irrelevant videos. Formally, we formulate it to an optimization problem as follows:

$$\max_{T(\cdot)} \mathbb{E}_{x \sim p} d_H (F(x), F(T(x)))$$

where $d_H (\cdot, \cdot)$ denotes the Hamming distance. As shown in most works on universal attack (Moosavi-Dezfooli et al., 2017; Khrulkov & Oseledets, 2018; Zhang et al., 2020b), $T(\cdot)$ is specified as directly adding the universal adversarial perturbation (UAP) to the original sample, i.e., $(x + \delta)$, which partly decreases the quality of the video and further affects the normal usages. In addition, it is difficult to generate a universal adversarial perturbation on the original video for attacking deep video hashing as demonstrated in our experiments.

2.2. Universal Adversarial Head

To overcome the above challenges, we specify $T(\cdot)$ as prepending adversarial frames in front of the original query video. Thus the generation of adversarial video $\hat{x}$ can be represented as:

$$\hat{x} = T(x) = [t + \delta, x]$$

where $t \in \mathbb{R}^{M \times H \times W \times C}$ denotes the clean frames and $\delta \in \mathbb{R}^{M \times H \times W \times C}$ is the adversarial perturbation on $t$. We name $t$ and $(t + \delta)$ as clean head and adversarial head in this paper, respectively. $[\cdot]$ denotes the aggregation operator that combines the adversarial head and the original video into an adversarial video. The stealthiness requirement can be realized by selecting the clean head which is adaptive to the video content. The adversarial head $(t + \delta)$ is universal, which means that it can be prepended on almost all videos from the distribution $p$ to bypass the deep video hashing.

Next we show how to generate $\delta$ based on the adversarial head. We assume that we can get access to a video set $X = \{x_i\}_{i=1}^{N}$, where $x_i \in \mathbb{R}^{M \times H \times W \times C}$ is sampled from a distribution $p$. To mislead the output of the deep hashing models on $\hat{x}$, the perturbation $\delta$ can be obtained by enlarging the distance between the hash code of each video $x_i \in X$ and that of the corresponding adversarial video $\hat{x}_i$. Moreover, to further ensure the stealthiness, we introduce the $\ell_{\infty}$ restriction on $\delta$. Then the objective function can be formulated as follows:

$$\min_{\delta} \frac{1}{N} \sum_{i=1}^{N} d_H (F(x_i), F([t + \delta, x_i]))$$

s.t. $||\delta||_{\infty} \leq \epsilon$,

where $\epsilon$ denotes the maximum perturbation strength.

For a pair of binary codes $h_i$ and $h_j$, since $d_H (h_i, h_j) = \frac{1}{2} (K - h_i^\top h_j)$, we can equivalently replace Hamming distance with inner product in the objective function. It’s worth noting that the activation function $\text{sign}(\cdot)$ in Eq. (1) is not differentiable in backpropagation. Thus, we alternatively employ $F'(x) = \tanh(f_p(x))$ to approximate $\text{sign}(\cdot)$ in the generation process. In summary, the overall optimization objective is as follows:

$$\min_{\delta} \frac{1}{N \cdot K} \sum_{i=1}^{N} (F(x_i))^\top F'(\|t + \delta, x_i\|)$$

s.t. $||\delta||_{\infty} \leq \epsilon$.

2.3. Data-free Adversarial Generation

In the optimization problem (5), generating UAH requires access to the videos from users, which itself may be contrary to the intention of information protection. Thus, we explore the data-free scenario (Mopuri et al., 2017), in which we can provide information protection technique without access to original videos. Specifically, we are still allowed to access the target model but not the data samples from the distribution $p$, which results in a data-free variant of universal adversarial head (DF-UAH). The proposed DF-UAH consists of two major steps: 1) generate proxy query videos; 2) optimize the universal adversarial head.

The videos are necessary carriers for solving the optimization problem in Eq. (5) to obtain the UAH. Consequently, we propose to generate some proxy query videos to realize the attack. These generated videos are supposed to behave similarly like the videos sampled from the data distribution $p$, i.e., they can be mapped through the target hashing model into the Hamming space. The key idea in our method is to sample some hash codes in the Hamming space and reverse engineer the input videos that leads to the output of these hash codes. We randomly sample a hash code $h' \in [-1, 1]^K$ to direct the proxy query video generation. We initialize the proxy query video $x'$ as the random noise and encourage the model output to be close to $h'$. The objective function is as follows:

$$\min_{x'} \|F'(x') - h'\|_2^2$$

When the loss value is optimized to approach 0, the proxy query video can be mapped into the Hamming space by the target model as the normal video. By minimizing the optimization problem in Eq. (6) with different $h'$, we can obtain a set of proxy query videos $X' = \{x'_i\}_{i=1}^{N}$. With $X'$, we can generate the UAH as the method in Section 2.2 by replacing $X$ with $X'$ in Eq. (5).

3. Experiments

We conduct experiments on a benchmark video dataset: UCF-101 (Soomro et al., 2012). We set the maximal length of the original video to 40. We follow the state-of-the-art video hashing methods (Gu et al., 2016; Liong et al., 2016;
Table 1. White-box attack results (MAP) against different architectures with various code lengths. “None” denotes the results of clean query videos. The best results are indicated in bold.

<table>
<thead>
<tr>
<th>Model</th>
<th>Method</th>
<th>16bits</th>
<th>32bits</th>
<th>48bits</th>
<th>64bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>52.60</td>
<td>51.80</td>
<td>52.76</td>
<td>49.32</td>
</tr>
<tr>
<td>UAP</td>
<td>51.91</td>
<td>51.65</td>
<td>52.99</td>
<td>49.52</td>
<td></td>
</tr>
<tr>
<td>UAH</td>
<td>5.99</td>
<td>4.21</td>
<td>5.00</td>
<td>3.85</td>
<td></td>
</tr>
<tr>
<td>DF-UAH</td>
<td>6.75</td>
<td>5.08</td>
<td>6.17</td>
<td>5.65</td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Transfer-based black-box attack results (MAP) against different architectures. “None” denotes the results of clean query videos. The best results are indicated in bold.

<table>
<thead>
<tr>
<th>Source Model</th>
<th>Method</th>
<th>A-R</th>
<th>A-L</th>
<th>V11-R</th>
<th>V11-L</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>51.8</td>
<td>53.76</td>
<td>57.85</td>
<td>56.95</td>
</tr>
<tr>
<td>UAP</td>
<td>51.65</td>
<td>53.38</td>
<td>57.41</td>
<td>56.42</td>
<td></td>
</tr>
<tr>
<td>A-FM</td>
<td>11.77</td>
<td>28.50</td>
<td>35.95</td>
<td>48.09</td>
<td></td>
</tr>
<tr>
<td>UAH</td>
<td>4.21</td>
<td>20.90</td>
<td>34.29</td>
<td>42.17</td>
<td></td>
</tr>
<tr>
<td>DF-UAH</td>
<td>5.08</td>
<td>26.26</td>
<td>35.28</td>
<td>40.00</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. White-box attack results (MAP) against different architectures with various code lengths. “None” denotes the results of clean query videos. The best results are indicated in bold.

Table 2. Transfer-based black-box attack results (MAP) against different architectures. “None” denotes the results of clean query videos. The best results are indicated in bold.

<table>
<thead>
<tr>
<th>Source Model</th>
<th>Method</th>
<th>A-R</th>
<th>A-L</th>
<th>V11-R</th>
<th>V11-L</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
<td>51.8</td>
<td>53.76</td>
<td>57.85</td>
<td>56.95</td>
</tr>
<tr>
<td>UAP</td>
<td>51.65</td>
<td>53.38</td>
<td>57.41</td>
<td>56.42</td>
<td></td>
</tr>
<tr>
<td>A-FM</td>
<td>11.77</td>
<td>28.50</td>
<td>35.95</td>
<td>48.09</td>
<td></td>
</tr>
<tr>
<td>UAH</td>
<td>4.21</td>
<td>20.90</td>
<td>34.29</td>
<td>42.17</td>
<td></td>
</tr>
<tr>
<td>DF-UAH</td>
<td>5.08</td>
<td>26.26</td>
<td>35.28</td>
<td>40.00</td>
<td></td>
</tr>
</tbody>
</table>

We take the UAP attack (Moosavi-Dezfooli et al., 2017) as a baseline, in which the adversarial perturbation is added on the original frames. We also compare with the Appending Adversarial Frames Method (A²FM) (Chen et al., 2021), which appends a few dummy frames to a video clip and then adds adversarial perturbations only on these new frames. We adopt the standard evaluation metric mean average precision (MAP) in the video retrieval task (Gu et al., 2016) to measure the attack performance. The lower MAP value indicates a better attack performance.

For the UAH and A²FM attack, the length of the adversarial frames is 5. For all attacks, the perturbation size \( \epsilon \) is 0.031 and the number of iterations is 2,000. The perturbations of UAP, A²FM, and UAH attack are computed on the set X including 100 videos randomly selected from the query set and evaluated on other 100 videos. For the DF-UAH, we generate 10 proxy videos and retain the other settings as above. We choose images with the texts of “Welcome to our channel!” as the clean frame for the UAH and A²FM attack in the following experiments (see Figure 2).

3.1. White-box Results

The results of the white-box attack are shown in Table 1. Under the white-box setting, all attacks are generated on the target model and used to attack this model. The UAP attack can only reduce the MAP results slightly in all cases. The MAP gaps between the query videos without and with universal perturbations are only 0.20%. It indicates that adding the perturbation on the original frames to perform the universal attack is infeasible. In contrast, the A²FM and UAH can effectively reduce the MAP results in attacking all models with different code lengths on the two datasets. Furthermore, the results demonstrate the significant superiority of the proposed UAH compared with the A²FM. Specifically, the query videos with the UAH dramatically reduce the MAP results by 48.87% on average. Moreover, the DF-UAH attack can achieve satisfactory attack performance, even with the absence of the data.

3.2. Black-box Results

Table 2 shows results of the transfer-based black-box attack, i.e., attacking the target model using the adversarial videos which are generated on a single source model. It shows that the UAP attack fails to generate transferable adversarial perturbations. Compared to the A²FM, our method can significantly degrade the retrieval performance of the black-box target model in all cases. We also observe that the DF-UAH is also transferable across different architectures. Moreover, an interesting phenomenon is that the transferability heavily depends on the architectures of the source and target model. More specifically, the source model whose architecture is closer to the target model achieves lower MAP results.

4. Conclusion

In this paper, we introduce the UAH and extend it to the data-free scenarios to protect the privacy information for the video data. We would like to emphasize that privacy protection for online shared videos deserves attention greatly. This work opens a door for applying the adversarial attack methods to the various video retrieval systems from the view of privacy protection. The more effective and stealthy methods are worth further study in the future.
References


Robertson, M. 500 hours of video uploaded to youtube every minute. URL https://tubularlabs.com/blog/hours-minute-uploaded-youtube/.


