Y. Zhan et al., "Cube-Evo: A Query-Efficient Black-Box Attack on Video Classification System," in IEEE Transactions on Reliability, doi: 10.1109/TR.2023.3261986.

https://doi.org/10.1109/TR.2023.3261986

Cube-Evo: A Query-Efficient Black-Box Attack on Video Classification System

Yu Zhan, Ying Fu, Liang Huang, Jianmin Guo, Heyuan Shi, Houbing Song, and Chao Hu

***Abstract*—The current progressive research in the domain of black-box adversarial attack enhances the reliability of deep neural network (DNN)-based video systems. Recent works mainly carry out black-box adversarial attacks on video systems by query-based parameter dimension reduction. However, the additional temporal dimension of video data leads to massive query consumption and low attack success rate. In this article, we embark on our efforts to design an effective adversarial attack on popular video classification systems. We deeply root the observations that the DNN-based systems are sensitive to adversarial perturbations with high frequency and reconstructed shape. Specifically, we propose a systematic attack pipeline Cube-Evo, aiming to reduce the search space dimension and obtain the effective adversarial perturbation via the optimal parameter group updating. We evaluate the proposed attack pipeline on two popular datasets: UCF101 and JESTER. Our attack pipeline reduces query consumption and achieves a high success rate on various DNN-based video classification systems. Compared with the state-of-the-art method Geo-Trap-Att, our pipeline averagely reduces 1.6× query consumption in untargeted attacks and 2.9× in targeted attacks. Besides, Cube-Evo improves 13% attack success rate on average, achieving new state-of-the-art results over diverse video classification systems.**

***Index Terms*—Adversarial examples, black-box attack, deep learning, system testing, video classification.**

I. Introduction

In the past few decades, videos have become indispensable media data with the development of applications, covering a wide range of fields, such as marketing, entertainment, and social networks. Some reliable statistics are that Facebook's video content page receives over 8 billion daily views, and the YouTube video platform will have 2.3 billion users worldwide by 2023. Therefore, how to manage massive video data and improve service quality has become a popular research topic.

The deep neural network (DNN)-based video systems are widely deployed due to the excellent performance. However, recent work [35] has exposed their security issues with malicious adversarial examples, which can make DNN-based video systems output wrong results with high confidence (referred to as adversarial attack). This introduces a series of threats to some popular applications, such as face recognition [47], action classification [12], and video surveillance [31]. In the adversarial scenario, the potential adversary aims to upload a carefully tampered and indistinguishable adversarial video to the victim video system. Such videos will make the victim system output arbitrarily wrong predictions (referred to as untargeted attack) or predefined labels of the adversary (referred to as targeted attack). In this article, we perform the real-world attack in a black-box way, where the adversary cannot access the structure and parameters of the victim video system. On one hand, real-world adversarial examples will reveal urgent problems for video classification systems. On the other hand, they can help researchers improve the robustness [14], [15], [16], [27] of these DNN-based systems.

The existing black-box video adversarial attacks have two main types: transfer attack [8], [23], [41], [42] and query attack [20], [24], [40], [44], [49]. Among them, query-based attacks are widely adopted and performed by the gradient estimation methods, which rely on multiple queries to find the expected gradient with maximal similarity to the true gradient of the system. These methods can effectively synthesize adversarial videos with high success rates, such as basic iterative method [22]. Unfortunately, as observed in [6], the number of queries highly depends on the dimension of input data and the number of perturbed pixels. This makes adversarial attacks for videos much harder than images because of the temporal information, e.g., a sequence of static images. In other words, the adversary needs to cost more queries when attacking video systems, and this behavior may be detected by the security mechanism. Although the existing works tried to improve the query-based gradient estimation methods by the data dimension reduction mechanism, attacking video systems still face the challenge of huge query consumption with a low attack success rate.

Researchers have also put forward valuable observations to improve attack efficiency. For example, the tiling strategy [19] demonstrates that image pixels with near coordinates have a similar adversarial gradient. Based on that, splitting the image

Manuscript received 27 June 2022; revised 12 November 2022; accepted 15 March 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62177046, Grant 61977062, and Grant 62202500, in part by the Hunan Natural Science Foundation of China under Grant 2021JJ30866, and in part by the National Key R&D Program of China under Grant 2022YFB3104003. Associate Editor: Henrique Madeira. *(Corresponding authors: Heyuan Shi; Ying Fu.)*

Yu Zhan, Liang Huang, Heyuan Shi, and Chao Hu are with the Central South University, Changsha 410017, China (e-mail: zhanyu@csu.edu.cn; hliang@csu.edu.cn; hey.shi@foxmail.com; huchao@csu.edu.cn).

Ying Fu is with the National University of Defense Technology, Changsha 410073, China (e-mail: fuying@nudt.edu.cn).

Jianmin Guo is with the Tsinghua University, Beijing 100190, China (e-mail: guojm17@mails.tsinghua.edu.cn).

Houbing Song is with the University of Maryland, Baltimore County, Baltimore, MD 21250 USA (e-mail: h.song@ieee.org).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TR.2023.3261986.

Digital Object Identifier 10.1109/TR.2023.3261986

as multiple squares and estimating the gradient of each square instead of each pixel will achieve higher efficiency. Al-Dujaili and O'Reilly[1] and Moon et al.[30] utilize a similar strategy that divides the image into coarse grids to search for adversarial perturbations locally. In summary, this implies that DNN-based (e.g., convolution neural network) image systems are sensitive to square-shaped adversarial perturbations because they conform to the 2-D convolution filters. Furthermore, Yin et al. [48] report that high-frequency perturbations easily affect DNN-based systems. This inspires various attacks [1], [30] to construct corners of intersecting spheres, such as inserting bound values of data domain (e.g., $[0, 255]^d$ for images) or randomly selecting the perturbation value from the componentwise value sets (e.g., $\{-10, 10\}$). In this article, we propose a black-box adversarial attack pipeline Cube-Evo for DNN-based video classification systems to address the shortcoming of the huge number of queries. Cube-Evo is devoted to searching for adversarial video, using as few queries as possible while ensuring the perturbation imperceptibility. Specifically, we first exploit the cube-based partition strategy to partition the parameter group of the original video. Then, we characterize the parameter group into two factors: position and magnitude. The position parameter group indicates that the group needs to perturb, and the magnitude parameter group decides the specific perturbation value. We continuously perform the random-search-based evolution scheme to search the optimal position parameter group and update the magnitude group by uniformly sampling the distribution from the component set. For detail, the random-search-based evolution scheme aims to improve the effectiveness of the population set. We first initialize the population set, which contains a collection of position and magnitude parameter groups, and exploit a hybrid and sequential approach of recombination, mutation, and selection operations to improve the population quality. During searching, we update the perturbation values by randomly sampling the componentwise sets and unifying them as a whole pixel channel. If the binary position equals 1, we will update the corresponding perturbation value in this way, otherwise not. We perform the attack pipeline until the query budget is exceeded or the adversarial examples are obtained in the population set, which can successfully attack the victim video classification system.

We conduct our attack pipeline on two popular datasets: UCF101 [34] and Jester [28] datasets. Extensive experiments demonstrate that our scheme outperforms the existing state-of-the-art (SOTA) methods Geo-Trap-Att [24] by reducing the fewer average number of queries (ANQ) by 37% in untargeted attack and 66% in targeted attack and improves the average attack success rate (ASR) by 13%. Compared to the transfer-based attack, Cube-Evo achieves the comparative attack result and low local resources cost. Moreover, experiments show that our attack scheme is more robust to different video systems, and the generated perturbations are more dispersed, which is not easily observed by human eyes. Our main contributions can be summarized as follows.

1) Our proposed scheme roots in the observations that video classification systems are sensitive to the reconstructed-shape adversarial perturbations. Continuously, we design a cube-based tiling strategy with a sliding mask to simultaneously partition video pixels at the same location in consecutive frames.
2) We design an effective attack pipeline for video classification systems. We first partition the original perturbation search space into cube-based parameter groups. Then, we exploit the random-search-based evolution algorithm for searching effective groups and updating their values with componentwise sets.
3) We extensively evaluate our attack pipeline on two popular datasets, i.e., UCF101 [34] and JESTER [28], based on four video classification systems, i.e., C3D [36], SlowFast [13], TPN [45], and I3D [4], with different metrics. Compared with the SOTA methods [24], [40], [49], experimental results show that our method can greatly improve the attack efficiency and synthesize adversarial perturbations with better imperceptibility.

The rest of this article is organized as follows. We introduce the related work in Section II and formulate our problem in Section III. Section IV describes our proposed scheme. We evaluate the performance of our scheme in Section V. Finally, Section VI concludes this article.

II. Related Work

A. Black-Box Adversarial Attack on Image Systems

There are three main types of image black-box adversarial attacks. The first is to generate adversarial examples in the pretrained local proxy system and then transfer to the victim system [7], [26], [50] (referred to as transfer attack), but the attack performance is unsatisfactory. The second is to estimate the gradient by querying the victim system [6], [19] (referred to as query attack). The third is the combination of the above two schemes. The adversary aims to search for the cross subspace of the local proxy system and the victim system to achieve the attack performance [9], [17], [46] (referred to as a hybrid attack), but it may also cost a lot of local computation resources. In this article, we focus on the query attack because of its low computation cost while achieving a high attack success rate.

As for query attack works, Ilyas et al. [19] observe that pixels with similar coordinates have similar gradients. Thus, they design a square-based tiling strategy to divide the image into a collection of squares to reduce the gradient estimation dimension. Bhagoji et al. [3] use principal component analysis and Tu et al. [37] exploit the autoencoder to search the potential and lower dimensional parameter subspace for efficient gradient estimation. An alternative type of query attack is the random-search-based scheme with more promising results. Croce and Hein [10] propose randomly selecting the modification pixels in the range of edges that vary widely and are not aligned with the axis. Andriushchenko et al. [2] propose randomly selecting local square locations and updating their values by componentwise set. Croce et al. [11] propose a general framework based on the random search, leveraging perturbation parameters of positions and values to achieve sparse attack (i.e., perturb partial image pixels). Meunier et al. [29] combine the tiling strategy with the random search method while adding perturbations to all the pixels, which may lead to the low imperceptibility of perturbations. Although random-search-based methods perform better than gradient estimation methods, their direct application in the video domain is hindered by the high-dimensional features of video data. In this article, we first attempt to design a cube-based attack tiling strategy in adversarial video attacks to exploit the temporal and spatial pixels and reduce the parameter space dimension. Inspired by Croce et al. [11], we characterize the parameter space of perturbations into two types (magnitudes and positions). We explore the evolution strategy with the high-frequency perturbation to achieve comparative attack performance.

B. Black-Box Adversarial Attack on Video Systems

Although recent works have explored adversarial video attacks in the white-box setting [8], [39], they are not in line with real-world application scenarios because of its loose attack restrictions, i.e., the adversary allows access to the full parameters of the victim video systems. In the black-box setting, the existing video adversarial attack works mainly adopt query-based gradient estimation methods, which can be categorized into two types. The first type leverages extra knowledge, such as combining query-based methods with transfer-based methods or training a local attack network for hybrid adversarial attacks. For example, Jiang et al. [20] first obtain transferable perturbations on local models and then correct adversarial gradients of each frame with the query-based method. Yan and Wei [44] train the agent model with a reinforcement learning framework and estimate adversarial gradients only on keyframes. However, these works consume huge local computation resources, making the attack process inefficient. On the contrary, the second type exploits the parameter dimension reduction mechanism to achieve competitive attack performance with low computation cost, i.e., without any local model or well-trained selector. Wei et al. [40] perform the gradient search on the keyframes and salient regions selected by the inheritance mechanism. Zhang et al. [49] exploit the optical flow to reveal the relative motion of regions between video frames for constructing an efficient motion-excited sampler. Li et al. [24] employ standard geometric transformation operations to reduce the search space and obtain the low-dimensional structured parameter search space. In this article, we first explore the random-search-based attack in the video field and design an efficient attack pipeline for implementing black-box query attacks without massive computational costs.

III. Problem Formulation

Let $\mathcal{F}_\theta : \mathbb{V} \to \mathbb{Y}^L$ be the victim video classification system. $\mathcal{F}$ represents the mapping relationship with the training video dataset $\mathbb{V}$ and the label space $\mathbb{Y}$ with L categories. θ is the system parameter. We denote the query video clip as $\boldsymbol{v} \in \mathcal{R}^{T\times H\times W\times C}$, where T is the number of frames, and W, H, and C denote the frame width, height, and color channel, respectively.

According to the adversary's knowledge acquired from the victim system, adversarial attacks can be divided into decision-based attacks (the adversary can only obtain the top-1 label from the system) and score-based attacks (the adversary can obtain all the categories of prediction scores). We discuss the score-based attack in this article. Given a query video clip $\boldsymbol{v}$, the victim system will output prediction scores for the label y of video $\boldsymbol{v}$, which can be assigned by the top-one output score, i.e., $\arg\max_{l=1,\dots,L} \mathcal{F}_\theta^l(\boldsymbol{v}) = y$.

In this article, we consider a black-box setting scenario that the adversary cannot access the system parameter $\boldsymbol{\theta}$. Given a query video $\boldsymbol{v}$, the potential adversary that aims to generate malicious perturbation δ and synthesize visually indistinguishable adversarial video $\boldsymbol{v}_{\text{adv}}$, i.e., $\boldsymbol{v}_{\text{adv}} = \boldsymbol{v} + \boldsymbol{\delta}$, finally makes the system output wrong. We mathematically formulate this problem as follows:

$$\arg\min_{\boldsymbol{\delta}} \mathcal{L}\left(\mathcal{F}_\theta(\boldsymbol{v}_{\text{adv}}), y_{\text{adv}}\right) \quad \text{s.t. } ||\boldsymbol{v}_{\text{adv}} - \boldsymbol{v}||_\infty \le \tau \tag{1}$$

where $\mathcal{L}(\cdot)$ is the objective function to measure the difference between the system's output label score. y_{adv} is the adversarial label, which varies with attack targets. In particular, the untargeted attack tries to change the ground truth label y predicted by the video classification system to an arbitrary label, i.e., $\arg\max_{l=1,\dots,L} \mathcal{F}_\theta^l(\boldsymbol{v}) \neq y$. The targeted attack is to change the predicted label to a specific label y_t, i.e., $\arg\max_{l=1,\dots,L} \mathcal{F}_\theta^l(\boldsymbol{v}) = y_t$. Note that the higher attack success rate and fewer queries indicate better attack performance. In order to achieve the imperceptibility of adversarial perturbations, we utilize $||\cdot||_\infty$ norm with the perturbation budget τ_{max}. Here, the adversarial perturbation will be limited into $[-\tau_{\text{max}}, \tau_{\text{max}}]$ by the CLIP$(\cdot)$ function.

Based on the above settings, we summarize the goal of our attack as follows:

1) *Attack goal*: The synthetic adversarial video $\boldsymbol{v}_{\text{adv}}$ can fool the video classification system $\mathcal{F}_\theta$ to output the wrong label, i.e., $\arg\max_{l=1,\dots,L} \mathcal{F}_\theta^l(\boldsymbol{v}) \neq y$.
2) *Query budget*: The adversary should use as few queries as possible to achieve the attack goal, i.e., query numbers $\le Q$, where Q is the query budget.
3) *Distortion imperceptibility*: The synthetic adversarial video $\boldsymbol{v}_{\text{adv}}$ is visually indistinguishable to the human eyes, i.e., $||\boldsymbol{v}_{\text{adv}} - \boldsymbol{v}||_\infty \le \tau$, where τ is the perturbation budget.

IV. Proposed Framework

In this section, we first introduce the overview of our black-box attack framework, which is shown in Fig. 1. Then, we describe the parameter dimension reduction mechanism based on the cube-based partition and exploit the random-search-based evolution algorithm to generate effective adversarial videos. Important symbols and corresponding definitions are summarized in Table I.

A. Framework Overview

In this section, we describe our proposed framework for black-box video adversarial attacks, called Cube-Evo. We first partition the parameterized video pixels through the cube-based partition strategy. Then, we initialize the perturbation magnitude parameter set Δ and the binary perturbation position parameter set Ω,

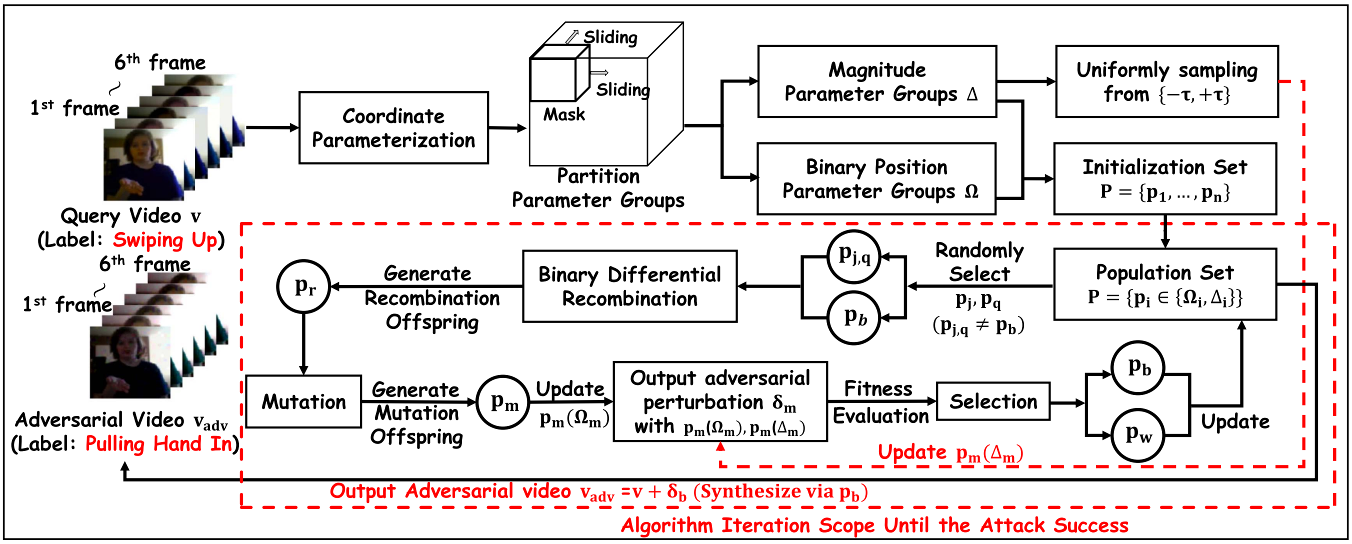
6th frame
1st frame
Query Video v
(Label: Swiping Up)
6th frame
1st frame
Adversarial Video v_adv
(Label: Pulling Hand In)
Coordinate Parameterization
Sliding
Sliding
Mask
Partition Parameter Groups
Magnitude Parameter Groups Δ
Binary Position Parameter Groups Ω
Uniformly sampling from {−τ, +τ}
Initialization Set P = {p1, …, pn}
Population Set P = {pi ∈ {Ωi, Δi}}
Randomly Select
pj, pq
(pj,q ≠ pb)
pj,q
pb
Binary Differential Recombination
Generate Recombination Offspring
pr
Mutation
Generate Mutation Offspring
pm
Update pm(Ωm)
Output adversarial perturbation δm with pm(Ωm), pm(Δm)
Fitness Evaluation
Selection
pb
pw
Update
Update pm(Δm)
Output Adversarial video v_adv = v + δb (Synthesize via pb)
Algorithm Iteration Scope Until the Attack Success

Fig. 1. Illustration overview of our proposed attack pipeline. The content in the red line and red box will be executed multiple times until the attack success.

TABLE I
NOTATION AND DEFINITION

Symbol	Definition
$\boldsymbol{v}$	Query video
$\boldsymbol{v}_{adv}$	Synthetic adversarial video
$\boldsymbol{\delta}$	Synthetic adversarial perturbation
$\mathcal{L}(\cdot,\cdot)$	Objective function
$\mathcal{M}$	Sliding Mask
$\mathcal{F}_\theta$	Video classification system
ϕ	Original parameter search space
$\phi_\mathcal{G}$	Partition parameter search space
$\Delta_\mathcal{G}$	Perturbation value of partition parameter
$\Omega_\mathcal{G}$	Binary selection partition parameter
$\mathcal{P}$	Evolution population set
$g(\cdot)$	Fitness value evaluation function

which is used to create the population set and $\mathcal{P}$. Next, we start running the iterative attack algorithm. We deploy the "random-search-based evolutionary algorithm" for updating Ω and "uniformly sampling from $\{-\tau,+\tau\}$" for updating Δ. For detail, the "random-search-based evolutionary algorithm" first initializes the population set $\mathcal{P} = \{p_1,\ldots,p_n | p_i \in \{\Omega_m, \Delta_m\}\}$. Then, it randomly selects two individuals p_j, p_q and p_b from the population set $\mathcal{P}$, where $p_j, p_q \neq p_b$, to exploit the binary differential recombination and generate the offspring p_r. The newly generated offspring p_r continues to mutate for generating p_m and then obtain $p_m(\Omega_m)$. The magnitude parameter group $p_m(\Delta_m)$ will be updated according to the binary parameter group $p_m(\Omega_m)$ via "uniformly sampling from $\{-\tau,+\tau\}$" and inherited from $p_b(\Delta_b)$. Continuously, we can synthesize the adversarial perturbation δ_m via p_m and obtain the current iteration adversarial video $\boldsymbol{v}^m_{\text{adv}} = \boldsymbol{v}+\delta_m$. Finally, we can obtain the fitness value of p_m via querying the victim model with the current adversarial video $\boldsymbol{v}^m_{\text{adv}}$. We select the best population individual p_b and the worst population individual p_w via its fitness value for updating the population set $\mathcal{P}$. The algorithm will continuously run until the attack succeeds, and we summarize the attack pipeline in Algorithm 1.

B. Reducing Search Space Dimension for Query-Based Attack

As mentioned above, we start our attack by developing the cube-based partition strategy to transform the original parameter search space into a lower dimension space. Given a query video $\boldsymbol{v} \in \mathcal{R}^{T\times H\times W\times C}$, we formally define the video pixel coordinates as (x,y,t,c), where (x,y,t) represents the coordinates of each pixel in the tth video frame and c is the color channel index. Therefore, we can parameterize the video pixel coordinates as the original parameter search space of adversarial perturbations, such as $\phi \in \mathcal{R}^{T\times H\times W\times C}$.

To achieve the search space reduction, we construct the sliding mask $\mathcal{M}$ with predefined stride S of size $r\times r\times r\times C$ to extract the temporal–spatial structure of the video pixels. Here, the sliding mask $\mathcal{M}$ divides the original parameter search space into multiple sets of partition groups. We denote the partition groups as $\{\phi_{\mathcal{G}(x,y,t,c)}\}^K$, for $x \in \{1,2,\ldots,P\}, y \in \{1,2,\ldots,Q\}$, and $t \in \{1,2,\ldots,L\}$, where $P = \frac{W-r}{S}+1$, $Q = \frac{H-r}{S}+1$, $L = \frac{T-r}{S}+1$, c is the color channel index, and K is the number of partition groups ($K = 3136$ in our experiment). To simplify our algorithm, we follow the work [11] to characterize the parameter partition groups $\phi_\mathcal{G}$ into two groups representing perturbation positions $\Omega_\mathcal{G} \in \{0,1\}^K$ and magnitudes $\Delta_\mathcal{G} \in \mathcal{R}^{K\times\gamma}$, where γ denotes the number of elements in one partition group, i.e., $\gamma = r\times r\times r\times C$. Intuitively, we decompose the process of synthesizing adversarial perturbations into pairs of binary position parameter group (i.e., $\Omega_\mathcal{G}$) and magnitude parameter group (i.e., $\Delta_\mathcal{G}$). The binary position parameter decides whether there is need to add the perturbation, and the magnitude parameter decides the value of added perturbation.

According to the observation in [48] that the DNN-based system is sensitive to the high-frequency adversarial perturbation, we follow the simple and effective way [2], [30] to update

Algorithm 1. Cube-Evo.

```
Input: Query video $\boldsymbol{v}$, sliding mask $\mathcal{M}$ with stride $s$
    and size $r$, query video label $y$, adversarial
    video label $y_{\text{adv}}$, query budget $\mathcal{Q}$, victim video
    classification system $\mathcal{F}_\theta$, population size $p$,
    mutation factor $u$;
Output: Adversarial video $\boldsymbol{v}_{\text{adv}}$;
Initialize the partition parameter search space $\phi$ with
    the sliding mask $\mathcal{M}$;
Characterize $\phi$ into two variable $\Omega$ and $\Delta$;
Initialize the population set $\mathcal{P} = \{p_1,\ p_2,\ldots,p_n\}$;
$q \leftarrow n$, $p_w \leftarrow \arg\max_p(\mathcal{P})$, $p_b \leftarrow \arg\min_p(\mathcal{P})$;
for $q \leq \mathcal{Q}$ do
    Yield $p_r$ via the randomly select $p_j, p_q \in \mathcal{P}$ where
      $p_{j,q} \neq p_b$, and combine $p_b$ to perform binary
      differential recombination.;
    Yield $p_m$ by altering the group of $p_r$ with the
      fraction $u$ from 0 to 1;
    Calculate the corresponding fitness value $g(p_m)$;
    if $g(p_m) < g(p_w)$ then
        $\mathcal{P}, p_w \leftarrow p_m$;
    end
    $p_w \leftarrow \arg\max_p(\mathcal{P})$, $p_b \leftarrow \arg\min_p(\mathcal{P})$;
    Construct adversarial video $\boldsymbol{v}_{\text{adv}} = \boldsymbol{v}+\delta_b$ where $\delta_b$
      synthesized by $p_b(\Omega_b)$ and $p_b(\Delta_b)$;
    $q$++;
    if $\boldsymbol{v}^b_{\text{adv}}$ achieve attack goal then
        break;
    end
end
return the adversarial video $\boldsymbol{v}^b_{\text{adv}}$;
```

the value of adversarial perturbation by randomly selecting the value $+\tau$ or $-\tau$, where τ is the perturbation budget. For detail, if the kth group value $\Omega_{\mathcal{G}_k}$ is equal to 1, the corresponding kth parameter group $\Delta_{\mathcal{G}_k}$ will be uniformly updated with pixel color channel index. Specifically, the value of the color channel is randomly sampling from the distribution $\{-\tau, \tau\}$, such as $\Delta_{\mathcal{G}_k(c=1)} = \tau$, $\Delta_{\mathcal{G}_k(c=2)} = \tau$, and $\Delta_{\mathcal{G}_k(c=3)} = -\tau$, where $\Delta_{\mathcal{G}_k(c=1)} = \tau$ denotes the first color channel of all the pixels in this group and is set as τ. Hence, we can formulate the update process of the kth parameter group $\Delta_{\mathcal{G}_k}$ as follows:

$$\Delta_{\mathcal{G}_k(c)} = \begin{cases} \texttt{Uniform}(\{-\tau, \tau\}), & \texttt{if} \quad \Omega_{\mathcal{G}_k} = 1 \\ 0, & \texttt{otherwise} \end{cases} \tag{2}$$

where c is the channel index and $\mathrm{c} = 1, 2, 3$. Hence, given a query video $\boldsymbol{v}$, we can update the parameter $\Delta_{\mathcal{G}}$ by (2) and strategically set partial parameters of $\Omega_{\mathcal{G}}$ as 1 and finally easily synthesize the adversarial video $\boldsymbol{v}_{\text{adv}}$. Notice that our parameter search space is reduced from $\mathcal{R}^{T\times H\times W\times C}$ to K. In the next section, we will introduce the random-search-based evolution algorithm to quickly find the optimal solution of $\Omega_{\mathcal{G}}$.

C. Synthesizing the Adversarial Video via Parameter Groups

This section describes the parameter $\Delta_{\mathcal{G}}$ update process and adversarial video $\boldsymbol{v}$ synthesis after the dimension reduction of parameter search space. We present the update process in Fig. 2 and introduce the synthesis in detail as follows.

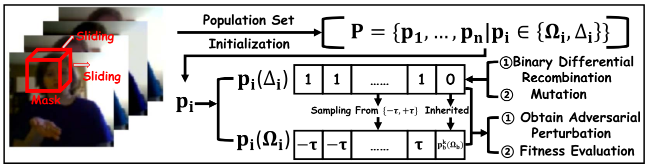

Fig. 2. Illustration of the update process for two parameters Ω and Δ, and the synthesis of adversarial perturbations.

1) Population Initialization: Given the partition parameters Ω and Δ, our goal is to find the most effective parameter group for synthesizing adversarial perturbation δ from the population set by continuously improving the quality of the population set. Specifically, each individual of population set $\mathcal{P}$ contains two parameters Ω and Δ, such as $\mathcal{P} = \{p_1(\Omega_1, \Delta_1), \ldots, p_n(\Omega_n, \Delta_n)\}$, where n is the size of population set. All the parameters of the individual population are initialized by zero matrices, i.e., none of the pixels is perturbed. In the population initialization process, we update each population candidate by randomly changing d bits of the parameter Ω from 0 to 1 and updating the value of corresponding parameter Δ by (2). For instance, if the ith group of Ω^i is initialized as 1, then update the ith group of Δ^i via (2), otherwise equal to 0. Finally, we synthesize the corresponding adversarial perturbation and video of each population candidate and make the fitness value evaluation by querying the victim video classification system. We denote the fitness value evaluation function as $g(p) = \mathcal{L}(\mathcal{F}_\theta(\boldsymbol{v}_{\text{adv}}), y_{\text{adv}})$, where $\mathcal{L}$ is the measure similarity function, e.g., cross-entropy function. Among the population set $\mathcal{P}$, we denote the individual population with the best fitness value as p_b and the worst individual population as p_w.

2) Binary Parameter Differential Recombination: The population set diversity is the most critical element of the algorithm's ability for effective parameters. We consider two mainstream evolution strategies to optimize the parameter group Ω: 1) genetic algorithm (GA) and 2) differential evolution (DE) algorithm. The main difference between these two strategies is that the GA achieves the offspring diversity with the binary encoding via the uniform crossover operation, e.g., uniformly selecting bits from two binary inputs $p_j(\Omega_j)$ and $p_q(\Omega_q)$ to synthesize a new candidate $p_r(\Omega_r)$. Hence, we first randomly select two individuals p_j and p_q from the population set $\mathcal{P}$, where $p_{j,q} \neq p_b$. Subsequently, we run the uniform recombination operation on Ω_j and Ω_q by randomly selecting each bit to generate the offspring $p_o(\Omega_o)$. We perform an additional step to simultaneously retain the optimal element of the best population $p_b(\Omega_b)$ and the new offspring $p_o(\Omega_o)$ generated by $p_j(\Omega_j)$ and $p_q(\Omega_q)$. Hence, we can generate the final recombination offspring p_r, and we formally define the kth group of the parameter $p_r(\Omega^k_r)$ as

$$p_r\left(\Omega^k_r\right) = p_b\left(\Omega^k_b\right) \oplus p_o\left(\Omega^k_o\right) \tag{3}$$

where $\oplus$ denotes the XOR operator. Moreover, the corresponding magnitude parameter group $p_r(\Delta_r)$ is updated via (2) if $p_r(\Omega^k_r)$ is altered from 0 to 1, otherwise inherited from $p_b(\Delta_b)$. We define the kth group of the parameter $p_r(\Delta_r^k)$ as

$$p_r(\Delta_r^k) = \begin{cases} \texttt{Uniform}(\{-\tau,\tau\}), & \texttt{if} \quad p_r(\Omega_r^k) = 1 \text{ and } p_b(\Omega_b^k) = 0 \\ p_b(\Delta_b^k), & \texttt{Otherwise} \end{cases}. \tag{4}$$

3) Mutation and Selection: Notice that the offspring generated only by uniform crossover is in low diversity. The mutation operation in DE encodes the real number space and realizes the diversity of offspring through the mutation operation, e.g., randomly changing the partial bits of $p_r(\Omega_r)$ from 0 to 1 while we perform in the binary space. Since mutation operation is the critical component of the evolution strategy for population diversity, one recent work [38] proposed to combine the uniform crossover and mutation to achieve richer offspring diversity. Inspired by the DE algorithm and the GA, we obtain the optimal binary parameter Ω through the uniform crossover and mutation in a hybrid and sequential manner. We randomly select fraction value u of $p_r(\Omega_r)$ (i.e., $d = 5$ groups in our experiment) and change it from 0 to 1, where the kth group $p_r(\Omega_r^k)$ is not equal to 1. Finally, we update the worst p_w or best p_b individual population of the population set by the newly synthesized offspring p_m. The fundamental idea of the evolution algorithm is to make the optimal individual with better fitness value survive in the population set. We run the selection operation to update the worst fitness population p_w with p_m if $g(p_m) < g(p_w)$ and improve the population quality. We perform the algorithm until the output p_b, which can synthesize effective adversarial videos (i.e., successfully attacking the video classification system).

V. Evaluation

A. Datasets and Metrics

To evaluate the performance of our Cube-Evo black-box adversarial attack, we introduce two popular datasets i.e., UCF101 [34] with 13 320 videos and 101 human action categories (e.g., Breast Stroke, Brushing Teeth, and Boxing Punching Bag) and JESTER [28] with 148 092 videos and 27 gesture action categories (e.g., Rolling Hand Forward, Shaking Hand, and Swiping Left). We follow the work [24] to randomly select one correct prediction query video from each category in UCF101 [34] and randomly select four correct prediction query videos from each category in JESTER [28] because it has less number of categories. In other words, we select 101 and 108 query videos from the UCF101 and JESTER datasets for attacking, respectively. For targeted attack, we randomly choose a target class for each video. We adopt three widely used metrics to measure the performance of our black-box adversarial example attack. The first metric is the ASR, i.e., the average success rate of attack within the query and perturbation budget. The second metric is the ANQ, i.e., the average queries number of attacks on all the query videos. The higher the average success rate and lower the queries, the better the attack performance. The third metric is the perceptibility (PER) to measure the imperceptibility of the synthesized adversarial perturbation, i.e, PER $= \sum_{i=1} |\boldsymbol{v}_{\text{adv}} - \boldsymbol{v}|_i$. The lower the PER, the better the imperceptibility of the synthesized adversarial video.

TABLE II
Performance of Four Video Classification Systems

Datasets	Black-box Video Classification System			
	C3D	SlowFast	TPN	I3D
UCF101	79.74%	84.97%	74.54%	70.54%
JESTER	89.54%	88.78%	91.57%	90.12%

B. Implementation Detail and Baseline

We carried out the implementation on PyTorch platform with Intel(R) Core(TM) i7-9700 CPU@3.00 GHz and two NVIDIA Geforce RTX 2080Ti. We take four typical video classification systems C3D [36], SlowFast [13], TPN [45], and I3D [4] as the victim video classification systems, and we report the classification performance of four video systems in Table II.

Regarding hyperparameters, we set the maximum adversarial perturbation budget $\tau_{\max}$ as 10 and start employing the flicker loss [32] as our objective function. r and s in Section IV-B are used to ensure the size of the mask and sliding stride, and we set $s = 4$ and $r = 4$ according to our empirical result. We set the number query budget as 60 000 for untargeted attacks and 100 000 for targeted attacks for computation efficiency. We set population size p as 10, the number of altering groups $d = 5$, and the mutation fraction value $u = 0.005$.

We compare our proposed attack pipeline to three SOTA methods for performance comparison: 1) Heuristic-Att [40] performs the inheritance mechanism to ensure the keyframes, and we exploit [18] to detect the saliency region; 2) Motion-Sampler-Att [49] performs the motion-vector-accumulation-based sampler to estimate the gradient, and we exploit the TVL1 flow [5] method to obtain the motion information; and 3) Geo-Trap-Att [24] performs the standard geometric transformation operation for gradient estimation, and we exploit the translation-dilation operation. As mentioned above, our attack pipeline consists of a cube-based partition strategy and an evolution algorithm to reduce the parameter search dimension. We provide empirical evidence that our practice can achieve better attack performance by analyzing the following two baselines: Cube-Bandit and Cube-Rand. Cube-Bandit is based on the bandit gradient estimation method [19] widely used in black-box adversarial video attacks [24], [49]. we exploit the upsampling strategy to add the same noise in each cube-based-group pixels instead of every video pixels. Cube-Rand randomly selects one cube-based partition group to add componentwise adversarial perturbation in each iteration. We summarize the following research questions to construct the experiments.

1) *RQ1.* How is the attack performance of Cube-Evo compared to other baseline methods?
2) *RQ2.* How is the attack stability of Cube-Evo affected by the various impact factor?
3) *RQ3.* How is the quality of the adversarial videos generated by Cube-Evo?

TABLE III
UNTARGETED ATTACK

Dataset	Method	C3D			SlowFast			TPN			I3D		
		ASR(↑)	ANQ(↓)	PER(↓)	ASR(↑)	ANQ(↓)	PER(↓)	ASR(↑)	ANQ(↓)	PER(↓)	ASR(↑)	ANQ(↓)	PER(↓)
UCF101	Heuristic-Att [40]	66.34%	30224	1.31	96.00%	10163	1.89	50.00%	36450	1.18	80.00%	22157	2.26
	Motion-Sampler-Att [49]	78.22%	16412	5.63	94.00%	6443	6.52	70.00%	34667	2.56	94.42%	9674	6.80
	Geo-Trap-Att [24]	80.20%	13275	6.97	99.00%	1686	8.62	76.19%	15518	5.51	97.00%	5833	8.74
	Cube-Bandit	81.19%	12325	7.42	98.00%	3539	8.89	74.00%	18052	5.85	98.00%	2848	8.71
	Cube-Rand	81.19%	12306	**1.01**	99.00%	2139	**1.39**	66.67%	21904	**0.62**	98.00%	2560	**1.44**
	Cube-Evo(Ours)	**86.14%**	**8983**	5.31	**100.00%**	**529**	5.56	**85.71%**	**12602**	3.78	**99.00%**	**1180**	6.06
JESTER	Heuristic-Att [40]	73.15%	29380	1.84	83.67%	18640	1.95	66.67%	30290	1.74	80.56%	23762	1.99
	Motion-Sampler-Att [49]	84.26%	7118	5.99	98.15%	6114	6.31	88.95%	7608	2.52	96.30%	5830	6.31
	Geo-Trap-Att [24]	95.37%	4114	6.98	**100.00%**	566	8.62	92.59%	5602	5.51	**100.00%**	2033	9.16
	Cube-Bandit	94.44%	5633	8.91	**100.00%**	1336	8.99	93.52%	6339	8.62	**100.00%**	2571	9.23
	Cube-Rand	**96.30%**	4091	**1.01**	**100.00%**	1046	**1.37**	**95.37%**	4344	**0.62**	**100.00%**	1107	**1.44**
	Cube-Evo(Ours)	**96.30%**	**2884**	5.31	**100.00%**	**340**	5.56	**95.37%**	**3294**	3.78	**100.00%**	**516**	5.89

We take the best results with bold for the mark. Cube-Evo performs the successful attack rate [high ASR] with fewer ANQ. Our method Cube-Rand achieves impressive results with the lowest PER.

C. *Attack Performance of Cube-Evo*

To fully evaluate the performance of our proposed attack pipeline, we experiment with two attack types (i.e., untargeted attack and targeted attack) to demonstrate the method's performance. We conduct it on four victim video classification systems with two datasets and report the evaluation of the results under three metrics (i.e., ASR, ANQ, and PER).

1) Attack Performance on Untargeted Attack: Table III reports the untargeted attack performance under various attack methods. Heuristic-Att and Motion-Sampler-Att obtain competitive PER results which means that it can generate more imperceptible adversarial perturbation. However, Geo-Trap-Att achieves the best attack success rate with the fewest queries number without breaking imperceptibility among the SOTA methods. Hence, we focus on it, and we have the following conclusion.

1) Compared with the Geo-Trap-Att, our proposed method achieves a significant performance improvement with a higher attack success rate and fewer queries. Specifically, Cube-Evo reduces the ANQ by 2287 (i.e., reduced by 37%), while the average attack success rate increased by an average of 3%. Our attack pipeline decreases the average value of PER by 0.89, which leading less adversarial perturbation.
2) When attacking SlowFast on the UCF101 dataset, we noticed that Cube-Evo decreases the number of queries by 1157 (i.e., improves the query efficiency by 68%). Similarly, Cube-Evo decreases the number of queries by 4653 when attacking I3D (i.e., improves the query efficiency by 79%), while C3D and TPN only decrease the number of queries by 4295 and 2916 (i.e., improve the query efficiency by 32% and 19%, respectively). We conclude that the SlowFast and I3D systems are more susceptible to reconstructed-shape adversarial perturbation than other systems because they extract both the temporal and spatial features through the 3-D convolution filters. A similar conclusion can be found in attacking the JESTER dataset. In other words, we consider that the C3D and TPN systems have higher robustness, making these two systems insensitive to adversarial perturbation.
3) Our methods Cube-Bandit and Cube-Rand achieve competitive performance. For example, when attacking the I3D system on the UCF101 dataset, Cube-Rand reduces 3273 queries, and Cube-Bandit reduces 2985 queries with both achieving 98% attack success rate. In particular, Cube-Rand achieves competitive results, which prove the advancement of the cube-based strategy and the random search algorithm.
4) We observe that Cube-Rand obtains the best imperceptibility adversarial attack among all the methods, i.e., the value of PER is the lowest. Cube-Rand only selects one group of added perturbations at each iteration. Hence, the generated adversarial perturbations are sparse with accompanied by higher perturbation consumption.

2) Attack Performance on Targeted Attack: Table IV shows the targeted attack performance of our proposed attack pipeline. Although the targeted attack is generally considered more complicated than the untargeted attack, the performance of our proposed attack pipeline shows more robustness performance. For example, when comparing the degree of decrease from untargeted to targeted attack, Geo-Trap-Att increases the number of queries by 29 669. In contrast, Cube-Evo only increases the number of queries by 7984. Moreover, in attacking the TPN system in the UCF101 dataset, Geo-Trap-Att achieved an attack success rate of 36.84% with 77 958 queries. In comparison, Cube-Evo achieved the attack success rate of 68.42% with only 34 819 queries, significantly improving the attack performance. This demonstrates that Cube-Evo requires fewer ANQs and achieves better attack performance. In addition, our method Cube-Rand also shows the prospect attack performance, which further proves the effectiveness of our method. In summary, the experimental results show the superior performance of Cube-Evo in the attack on video classification systems.

TABLE IV
TARGETED ATTACK

Dataset	Method	C3D			SlowFast			TPN			I3D		
		ASR(↑)	ANQ(↓)	PER(↓)	ASR(↑)	ANQ(↓)	PER(↓)	ASR(↑)	ANQ(↓)	PER(↓)	ASR(↑)	ANQ(↓)	PER(↓)
UCF101	Heuristic-Att [40]	34.81%	33578	6.99	57.52%	23205	6.42	33.79%	98661	7.14	40.69%	97145	4.33
	Motion-Sampler-Att [49]	73.33%	92280	7.10	71.43%	51515	7.57	30.00%	96069	2.37	48.57%	83786	6.45
	Geo-Trap-Att [24]	83.47%	66479	9.99	92.00%	18108	9.45	36.84%	77958	6.48	69.00%	49732	8.72
	Cube-Bandit	87.13%	38698	9.97	90.00%	29892	9.56	41.11%	72561	6.67	86.00%	34662	9.65
	Cube-Rand	**100.00%**	12527	**5.38**	93.00%	13504	**4.37**	51.00%	52859	**2.11**	77.00%	29063	**3.70**
	Cube-Evo(Ours)	**100.00%**	**10009**	9.45	**95.00%**	**11672**	7.44	**68.42%**	**34819**	6.25	**88.00%**	**15709**	9.02
JESTER	Heuristic-Att [40]	48.94%	51761	7.88	57.36%	57457	7.02	41.28%	94839	3.99	67.21%	73697	6.69
	Motion-Sampler-Att [49]	86.67%	54605	6.37	78.95%	39676	6.58	65.56%	69622	2.41	75.51%	52219	6.53
	Geo-Trap-Att [24]	**100%**	9898	9.95	**100%**	6127	9.79	79.25%	32864	9.08	96.30%	17717	9.79
	Cube-Bandit	99.07%	12360	9.93	98.15%	19289	9.79	77.78%	38867	9.09	92.59%	24810	9.79
	Cube-Rand	**100%**	4768	**3.53**	**100%**	5174	**3.47**	87.04%	18753	**3.24**	99.07%	5484	**2.84**
	Cube-Evo(Ours)	**100%**	**2608**	9.49	**100%**	**3704**	9.47	**88.68%**	**13713**	8.65	**100.00%**	**1967**	9.55

We take the best results with bold for the mark. Cube-Evo performs the successful attack rate [high ASR] with fewer ANQ. Our method Cube-Rand achieves impressive results with the lowest PER.

TABLE V
ATTACK PERFORMANCE WITH DIFFERENT PERTURBATION BUDGET τ_{MAX}

Attack Type		Untarget						Target					
$\tau_{\max}$	Dataset	UCF101			JESTER			UCF101			JESTER		
	Methods	ASR(↑)	ANQ(↓)	PER(↓)	ASR(↑)	ANQ(↓)	PER(↓)	ASR(↑)	ANQ(↓)	PER(↓)	ASR(↑)	ANQ(↓)	PER(↓)
8	Geo-Trap-Att [24]	96.00%	6908	7.11	99.07%	3013	7.64	65.00%	56456	6.93	95.37%	21054	7.86
	Cube-Evo (Ours)	**99.00%**	**1603**	**5.37**	**100%**	**611**	**5.31**	**88.00%**	**17983**	**7.26**	**98.15%**	**4400**	**7.74**
10	Geo-Trap-Att [24]	97.00%	5833	8.74	**100%**	2033	8.74	69.00%	49732	8.72	96.30%	17717	9.79
	Cube-Evo (Ours)	**99.00%**	**1180**	**6.06**	**100%**	**516**	**5.89**	**88.00%**	**15709**	**9.02**	**100%**	**1967**	**9.55**
16	Geo-Trap-Att [24]	97.00%	3331	13.08	**100%**	722	13.49	80.00%	34753	13.91	99.07%	8024	15.39
	Cube-Evo (Ours)	**100.00%**	**297**	**7.59**	**100%**	**271**	**6.47**	**90.00%**	**12396**	**14.45**	**100%**	**1113**	**13.69**

We take the best results with bold for mark (attack type: untargeted and targeted attack, victim video classification system: I3D, datasets: UCF101 and JESTER).

3) Answer to RQ1: Compared with other approaches, Cube-Evo attack pipeline is efficient in generating adversarial videos in both the attack types, capable of drastically reducing the performance of video systems with high success rates. Moreover, two baseline strategies Cube-Bandit and Cube-Evo have also shown promising performance.

D. Ablation Study

1) Effect of Different Perturbation Budget $\tau_{\max}$: The authors in [22] and [35] demonstrated that the magnitude of adversarial perturbation can affect the attack performance. In this subsection, we report the impact of different adversarial perturbation budgets $\tau_{\max}$ as follows. Among the SOTA methods, Geo-Trap-Att achieves the best attack performance, and we select it as our comparison method.

Table V reports the result of execution on various perturbation budget $\tau_{\max}$ values on Geo-Trap-Att and Cube-Evo method, and we have the following conclusion.

1) Geo-Trap-Att and Cube-Evo conform to the regularity that when the value of $\tau_{\max}$ increases, the performance of corresponding three indicators (i.e., ASR, ANQ, and PER) will improve.
2) We notice that when $\tau_{\max} = 10$, the ANQs for Geo-Trap-Att and Cube-Evo are 19 372 and 77 348, respectively. Moreover, when $\tau_{\max} = 8$, the value of the average query number in Cube-Evo and Geo-Trap-Att increased by 5225 and 10 083, respectively. Cube-Evo should be more susceptible to the value of $\tau_{\max}$, since the value of adversarial perturbation is sampled from $\{-\tau_{\max}, \tau_{\max}\}$. It turns out that Cube-Evo is more robust than Geo-Trap-Att. A similar conclusion can be obtained in the line of $\tau_{\max} = 16$.
3) We observe the implementation of the untargeted attack on the PER metric and demonstrate the prospect of Cube-Evo. For example, when the value of $\tau_{\max} = 10$, the average values of PER of Geo-Trap-Att and Cube-Evo are 8.95 and 5.98, respectively. When $\tau_{\max} = 16$, the PER value of Geo-Trap-Att increased by 4.34, while Cube-Evo only increases by 1.05. These observations conclude that fewer parameter sets will be selected for perturbation when our proposed attack pipeline obtains a larger perturbation budget.

2) Hyperparameter Setting (r, s): Before setting the parameters to $s = 4$ and $r = 4$, we perform a grid search on the parameters r and s. Here, we randomly select 20 query videos on the JESTER dataset with the I3D model to test the performance of the targeted attack. Note that the sliding mask $\mathcal{M}$ that we set in

TABLE VI
GRID SEARCH ON HYPERPARAMETERS (r, s)

Parameter		ASR(↑)	ANQ(↓)	PER(↓)	Parameter	ASR(↑)	ANQ(↓)	PER(↓)	Parameter	ASR(↑)	ANQ(↓)	PER(↓)
Non-Overlap	r=1,s=1 (TI=200704)	60.00%	78826	0.68	r=1,s=2 (TI=25088)	95.00%	15414	1.25	r=1,s=4 (TI=3136)	95.00%	8347	2.47
	r=1,s=8 (TI=392)	100.00%	2171	5.51	r=1,s=12 (TI=200)	100.00%	2017	7.45	r=1,s=14 (TI=128)	100.00%	3467	8.30
	r=2,s=2 (TI=25088)	95.00%	14269	1.16	r=2,s=4 (TI=3136)	95.00%	8001	2.36	r=2,s=8 (TI=392)	100.00%	2087	5.43
	r=2,s=12 (TI=200)	100.00%	2160	7.70	r=2,s=14 (TI=128)	100.00%	2694	8.21	r=4,s=4 (TI=3136)	100.00%	2546	2.85
	r=4,s=8 (TI=392)	100.00%	3585	6.38	r=8,s=8 (TI=392)	95.00%	7308	6.83	**Cube-Evo** (r=4,s=4)	100.00%	1058	8.53
Parameter		ASR(↑)	ANQ(↓)	PER(↓)	Parameter	ASR(↑)	ANQ(↓)	PER(↓)	Parameter	ASR(↑)	ANQ(↓)	PER(↓)
Overlap	s=1,r=2 (TI=181415)	70.00%	70752	0.51	s=1,r=4 (TI=154453)	5.00%	96453	0.10	s=1,r=12 (TI=51005)	0.00%	100000	0.01
	s=2,r=4 (TI=21175)	95.00%	21780	0.83	s=2,r=8 (TI=14045)	5.00%	95175	0.13	s=2,r=10 (TI=10816)	5.00%	95072	0.10
	s=4,r=8 (TI=2187)	60.00%	44888	1.01	s=4,r=10 (TI=1352)	40.00%	61601	0.66	s=4,r=12 (TI=1352)	10.00%	90115	0.56

We implement the Cube-Rand (our baseline 2) in this experiment on the JESTER dataset, I3D model with the targeted attack. We randomly select 20 query videos for efficiency evaluation. We compute the tilling (TI) number in different parameters (i.e., the number of partition groups). When $r = 4$ and $s = 4$, Cube-Rand achieves the comparative ASR and PER results. Our proposed Cube-Evo achieves the best ASR and ANQ results.

our article finally divides the video pixels into partition groups. We can obtain different partition group policies by adjusting the step size s and the size r of the sliding mask $\mathcal{M}$, and similar schemes are also discussed in [43]. When $s < r$, the obtained groups will contain the overlapped pixels, and the nonoverlapped group will be obtained when $s \geq r$. We present the grid search results with the above two partition schemes through Table VI and summarize the following conclusions.

1) Intuitively, the attack effect of the nonoverlapping parameter partition group is generally better than that of the overlapping parameter partition group. This conclusion differs from the work on image black-box attack [2], which exploits the random selection scheme of square-shaped pixels in the image black-box adversarial attack (i.e., overlapping parameter group attack).
2) In the nonoverlapping parameter group setting, r and s are smaller, and the number of partition parameters is larger, e.g., tilling = 200 704 is the largest when $r = 1$ and $s = 1$. Conversely, the larger r and s, the smaller the number of parameter groups to be partitioned, e.g., tilling = 392 is the smallest when $r = 8$ and $s = 8$. We find that when $r = 4, s = 4$ can simultaneously achieve competitive ASR and ANQ results and achieve the best PER. We consider that when tilling = 3136, Cube-Rand obtains sufficient parameter search space. When tilling = 392, the parameter search space is insufficient, leading to a larger PER value.
3) In the overlapping partition strategy, the optimal ASR and ANQ results are achieved when $s = 2$ and $r = 4$. We believe that the optimal attack efficiency can be achieved when $r = 4$, which is consistent with the results in the nonoverlapping partition strategy. We conclude that the video classification system mainly extracts four consecutive frames of the video and 16 pixels ($r = 4$) in the same position of the corresponding frame to achieve the feature extraction of the video.

TABLE VII
ATTACK WITH DIFFERENT SEED

Method	Key-Frame-Att [44]			Geo-Trap-Att [24]			Cube-Evo (Ours)		
	ASR (↑)	ANQ (↓)	PER (↓)	ASR (↑)	ANQ (↓)	PER (↓)	ASR (↑)	ANQ (↓)	PER (↓)
AVG	93.98%	10078	3.80	94.35%	19234	9.77	99.35%	2756	9.36
SD	0.46%	195	0.01	1.87%	941	0.03	0.59%	602	0.19
SE	0.15%	65	0.62	0.62%	313	0.01	0.19%	201	0.06

We implement the experiment on the JESTER dataset with I3D under the targeted attack. We provide the AVG, SD, and SE with multiple running results. Cube-Evo performs the statistical result of successful attack rate [high ASR] with fewer ANQs.

3) Random Seed for Attack Influence: We consider the randomness of the code program, which will cause fluctuation in the attack performance. Hence, we report the statistical results on the different random seeds of the program after running the attack ten times. We collect the average result (AVG), standard deviation (SD), and standard error (SE) and report the result in Table VII. Cube-Evo receives statistically fewer queries and higher attack success rates than those of the baseline methods.

4) Local Resource Consumption: We implement the transfer-based method in our experiments on the JESTER dataset and I3D with the targeted attack. We show the comparison attack results in Table VIII and present the related local resource consumption (e.g., the number of the local model parameters size, GPU cost, and consumption time in every iteration). Patch-Att and Keyframe-Att require more local resources and consumption time since they need to compute the backward gradient or train a reinforcement learning agent in each iteration to enhance the attack performance. Compared to the transfer-based attack, Geo-Trap and Cube-Evo exploit

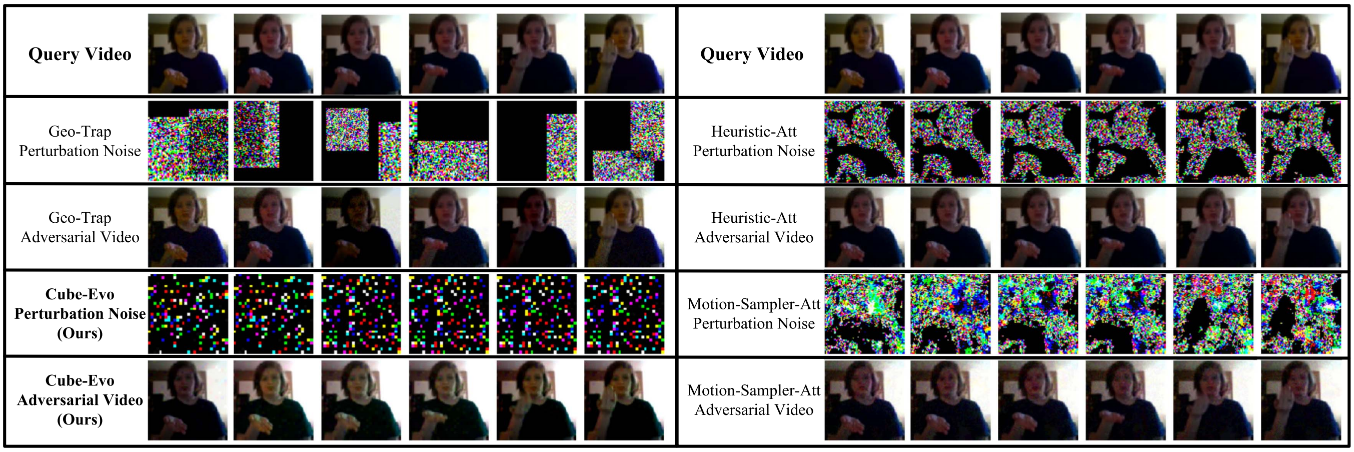
Query Video
Geo-Trap Perturbation Noise
Geo-Trap Adversarial Video
Cube-Evo Perturbation Noise (Ours)
Cube-Evo Adversarial Video (Ours)
Query Video
Heuristic-Att Perturbation Noise
Heuristic-Att Adversarial Video
Motion-Sampler-Att Perturbation Noise
Motion-Sampler-Att Adversarial Video

Fig. 3. Adversarial video (six frames) synthesized by various adversarial attack methods. (Attack type: untargeted attack, victim video classification system: I3D and dataset: JESTER).

TABLE VIII
LOCAL RESOURCE CONSUMPTION COMPARED TO TRANSFERABLE-BASED METHODS

Dataset	Method	Model Size	GPU Cost	Time Consumption
JESTER	Patch-Att [20]	44.70M	525MB	0.48s
	KeyFrame-Att [44]	47.33M	882MB	0.85s
	Geo-Trap-Att [24]	**0.00**	**0.00**	**0.10s**
	Cube-Evo (Ours)	**0.00**	**0.00**	0.12s

We implement the experiment on the JESTER dataset with I3D under the targeted attack. We take the best results with bold for the mark. We record three metrics: the number of the local model parameters size (model size), GPU Cost, and the time consumption in every iteration (time consumption).

the geometric transformation operation to reduce the search dimension, which is low-cost and easy to deploy.

5) Adversarial Attack Against Defense Mechanism: Defense mechanisms against neural networks are very important in evaluating attack performance. Overall, all the approaches to adversarial attacks have broad usages, such as discovering potential security holes in DNNs and evaluating the robustness testing of DNN-based software. Developers deploying DNN-based software in real scenarios will consider corresponding defense mechanisms to improve software robustness. We deploy the defense mechanism and detection mechanism for the I3D network on the JESTER dataset, i.e., Adversarial training-based [33] and Feature Squeeze [25]. We provide the attack performance under the three scenarios (i.e., without defense mechanism, Adversarial training-based defense mechanism, and Feature Squeeze detection mechanism) in Table IX. The query-based attack method can be defended by using other detection methods such as PRADA [21]. It detects the distribution of continuous query data from a single user. However, this defense can be bypassed by creating Sybil users or changing the attack network proxies. We summarize the following conclusion.

1) After adversarial training, the performance of all the attack methods degrades to some extent. For example, the ASR of Cube-Evo without the defense mechanism is 100% and requires 1967 queries. The ASR drops by 1.41%, and the number of attacks increases by 5350. Therefore, adversarial training can enhance a certain defense effect but cannot eliminate potential attack risks.
2) The scheme of Feature Squeeze shows effective adversarial video detection performance. The detection effect of Patch-Att is relatively poor, which adds perturbation to all the pixels.

Answer to RQ2: Cube-Evo's attack process is robust and low cost, not disturbed by the impact factor (e.g., perturbation budget, random program seed, and defense mechanism).

E. Quality of Generated Adversarial Video by Cube-Evo

This subsection briefly shows the adversarial videos generated by the methods mentioned above. Fig. 3 shows an adversarial visualization example. We observe that adversarial videos generated by the existing methods are indistinguishable from query videos. Although the illustrated example obtains sparse perturbation, the adversarial perturbations generated by our method are more scattered and more imperceptible than the concentrated perturbation.

Answer to RQ3: Cube-Evo can generate indistinguishable adversarial videos and significantly degrade the performance of video classification systems.

F. Novelty Statement and Future Work

This article mainly explores query-based adversarial attacks on video classification systems in a black-box setting. We mainly solve two main challenges: the high dimension of video data leads to a large number of queries and a low attack success rate. Previous work introduced relevant prior knowledge, such as transferable-patch-based perturbation, select keyframes and key regions, optical flow estimation, and geometric transformations. Table X compares the difference between the baseline method of video black-box attacks. The main contributions of this article are divided into two points: the first is to develop a Cube-based partition strategy to reduce the search space, and the second is to define a reasonable parameter group framework to realize the evolution-strategy-based search scheme. The partition strategy and the parameter group framework explored in this

TABLE IX
ATTACK AGAINST DEFENSE AND DETECTION MECHANISM

Method	Without Defense			Adversarial Training-Based [33]			Feature Squeeze [25]
	ASR(↑)	ANQ(↓)	PER(↓)	ASR(↑)	ANQ(↓)	PER(↓)	ADR(↓)
Patch-Att [20]	86.11%	31432	**3.43**	81.82%	39978	**3.79**	74.08%
KeyFrame-Att [44]	94.44%	9985	3.79	90.88%	13939	4.58	55.85%
Geo-Trap-Att [24]	96.30%	17717	9.79	91.76%	24959	9.35	**52.78%**
Cube-Evo(Ours)	**100.00%**	**1967**	9.55	**98.59%**	**7317**	9.67	53.89%

We implement the experiment on the JESTER dataset on I3D with the targeted attack. We implement the metrics: ASR, ANQ, PER, and attack detection rate (ADR). Moreover, with the lower ADR, the detection mechanism is more difficult to detect malicious adversarial video. We take the best results with bold for the mark.

TABLE X
COMPARED TO THE OTHER BASELINE METHOD

Method	**No Pretraining model**	**Spatial and Temporal**	**High-Frequency Perturbation**
Patch-Att [20]	No	No	No
Heuristic-Att [40]	Yes	Yes	No
Motion-Att [49]	Yes	No	No
KeyFrame-Att [44]	No	No	No
Geo-Trap-Att [24]	Yes	No	No
Cube-Evo(Ours)	Yes	Yes	Yes

Cube-Evo, compared to current black-box attack methods for videos, without utilizing a pretrained model (no pretraining model, yes is better), considers both spatial and temporal dimension (spatial and temporal, yes is better), and considers high-frequency adversarial perturbation (high-frequency perturbation, yes is better).

article bring inspiration for future work, i.e., to explore more efficient partition strategies or parameter group optimization schemes.

Although Cube-Evo achieves comparable attack performance and queries, the perceptual results still need to be improved. We believe that the recombination and mutation in the random-based search algorithm's search strategy rapidly increase the magnitude of adversarial perturbations (i.e., reduced imperceptibility of adversarial videos). Our future work will consider a more balanced scheme that achieves both the target goal and imperceptibility. Recent work in query-based attacks in the image domain fully incorporates the "hot start" capability of locally pretrained models. However, it is not directly portable in black-box attacks in the video domain. The black-box attack on the video classification system still requires a large number of queries. Continuously, we will utilize the local pretrained video classification model to achieve a hot start and a more efficient black-box attack.

VI. CONCLUSION

In this article, we investigated the black-box adversarial attack on the video classification system, improving the query efficiency and attack success rate of the synthesized adversarial video. Our attack intended to deploy the cube-based tiling strategy and random-search-based evolution algorithm over the attack pipeline. We extensively evaluated over two popular datasets (UCF101 and JESTER) and four victim video classification systems (C3D, SlowFast, TPN, and I3D) under our sequential combination setting. Compared with SOTA methods, experimental results showed that our proposed attack pipeline exhibits favorable attack success rates while reducing query consumption.

REFERENCES

[1] A. Al-Dujaili and U.-M. O'Reilly, "Sign bits are all you need for black-box attacks," in *Proc. Int. Conf. Learn. Represent.*, 2020. [Online]. Available: https://openreview.net/forum?id=SygW0TEFwH

[2] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: A query-efficient black-box adversarial attack via random search," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 484–501.

[3] A. N. Bhagoji, W. He, B. Li, and D. Song, "Practical black-box attacks on deep neural networks using efficient query mechanisms," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 158–174.

[4] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4724–4733.

[5] A. Chambolle, "An algorithm for total variation minimization and applications," *J. Math. Imag. Vis.*, vol. 20, nos. 1/2, pp. 89–97, 2004.

[6] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, "ZOO: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proc. 10th ACM Workshop Artif. Intell. Secur.*, 2017, pp. 15–26.

[7] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh, "EAD: Elastic-net attacks to deep neural networks via adversarial examples," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 10–17.

[8] Z. Chen, L. Xie, S. Pang, Y. He, and Q. Tian, "Appending adversarial frames for universal video attack," in *Proc. Winter Conf. Appl. Comput. Vis.*, 2021, pp. 3198–3207.

[9] S. Cheng, Y. Dong, T. Pang, H. Su, and J. Zhu, "Improving black-box adversarial attacks with a transfer-based prior," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 10932–10942.

[10] F. Croce and M. Hein, "Sparse and imperceivable adversarial attacks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 4723–4731.

[11] F. Croce, M. Andriushchenko, N.D. Singh, N. Flammarion, and M. Hein, "Sparse-RS: A versatile framework for query-efficient sparse black-box adversarial attacks," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 6437–6445.

[12] A. Diba et al., "Spatio-temporal channel correlation networks for action classification," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 299–315.

[13] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6201–6210.

[14] J. Guo, Y. Jiang, Y. Zhao, Q. Chen, and J. Sun, "DLFUZZ: Differential fuzzing testing of deep learning systems," in *Proc. 26th ACM Joint Meeting Eur. Softw. Eng. Conf. Symp. Found. Softw. Eng.*, 2018, pp. 739–743.

[15] J. Guo, Y. Zhao, H. Song, and Y. Jiang, "Coverage guided differential adversarial testing of deep learning systems," *IEEE Trans. Netw. Sci. Eng.*, vol. 8, no. 2, pp. 933–942, Apr./Jun. 2021.

[16] J. Guo, Q. Zhang, Y. Zhao, H. Shi, Y. Jiang, and J.-G. Sun, "RNN-test: Towards adversarial testing for recurrent neural network systems," *IEEE Trans. Softw. Eng.*, vol. 48, no. 10, pp. 4167–4180, Oct. 2022.

[17] Y. Guo, Z. Yan, and C. Zhang, "Subspace attack: Exploiting promising subspaces for query-efficient black-box attacks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 3820–3829.

[18] X. Hou and L. Zhang, "Saliency detection: A spectral residual approach," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2007, pp. 1–8.

[19] A. Ilyas, L. Engstrom, and A. Madry, "Prior convictions: Black-box adversarial attacks with bandits and priors," in *Proc. Int. Conf. Learn. Represent.*, 2019. [Online]. Available: https://openreview.net/forum?id=BkMiWhR5K7

[20] L. Jiang, X. Ma, S. Chen, J. Bailey, and Y.-G. Jiang, "Black-box adversarial attacks on video recognition models," in *Proc. 27th ACM Int. Conf. Multimedia*, 2019, pp. 864–872.

[21] M. Juuti, S. Szyller, S. Marchal, and N. Asokan, "PRADA: Protecting against DNN model stealing attacks," in *Proc. IEEE Eur. Symp. Secur. Privacy*, 2019, pp. 512–527.

[22] A. Kurakin, I.J. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," in *Proc. Int. Conf. Learn. Represent.*, 2017. [Online]. Available: https://openreview.net/forum?id=HJGU3Rodl

[23] S. Li et al., "Stealthy adversarial perturbations against real-time video classification systems," in *Proc. Netw. Distribut. Syst. Secur. Symp.*, 2019. [Online]. Available: https://www.ndss-symposium.org/ndss2019/accepted-papers/

[24] S. Li et al., "Adversarial attacks on black box video classifiers: Leveraging the power of geometric transformations," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 2085–2096.

[25] G. Liu, I. Khalil, and A. Khreishah, "ZK-GanDef: A GAN based zero knowledge adversarial training defense for neural networks," in *Proc. IEEE/IFIP 49th Ann. Int. Conf. Dependable Syst. Netw.*, 2019, pp. 64–75.

[26] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into transferable adversarial examples and black-box attacks," in *Proc. Int. Conf. Learn. Represent.*, 2017. [Online]. Available: https://openreview.net/forum?id=Sys6GJqxl

[27] D. Ma, J. Guo, Y. Jiang, and X. Jiao, "HDTest: Differential fuzz testing of brain-inspired hyperdimensional computing," in *Proc. IEEE/ACM 58th Des. Autom. Conf.*, 2021, pp. 391–396.

[28] J. Materzynska, G. Berger, I. Bax, and R. Memisevic, "The jester dataset: A large-scale video dataset of human gestures," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops*, 2019, pp. 2874–2882.

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

[30] S. Moon, G. An, and H. O. Song, "Parsimonious black-box adversarial attacks via efficient combinatorial optimization," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4636–4645.

[31] R. Nawaratne, D. Alahakoon, D. D. Silva, and X. Yu, "Spatiotemporal anomaly detection using deep learning for real-time video surveillance," *IEEE Trans. Ind. Informat.*, vol. 16, no. 1, pp. 393–402, Jan. 2020.

[32] R. Pony, I. Naeh, and S. Mannor, "Over-the-air adversarial flickering attacks against video recognition networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 515–524.

[33] A. Shafahi et al., "Adversarial training for free!," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 3353–3364.

[34] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," in *Proc. Int. Conf. Comput. Vis.*, 2012. [Online]. Available: https://www.crcv.ucf.edu/papers/UCF101_CRCV-TR-12-01.pdf

[35] C. Szegedy et al., "Intriguing properties of neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2014. [Online]. Available: https://openreview.net/forum?id=kklr_MTHMRQjG

[36] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2015, pp. 4489–4497.

[37] C.-C. Tu et al., "AutoZOOM: Autoencoder-based zeroth order optimization method for attacking black-box neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 742–749.

[38] V. O. Vo, E. Abbasnejad, and D. C. Ranasinghe, "Query efficient decision based sparse attacks against black-box deep learning models," in *Proc. Int. Conf. Learn. Represent.*, 2022. [Online]. Available: https://openreview.net/pdf?id=73MEhZ0anV

[39] X. Wei, J. Zhu, S. Yuan, and H. Su, "Sparse adversarial perturbations for videos," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 8973–8980.

[40] Z. Wei et al., "Heuristic black-box adversarial attacks on video recognition models," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 12338–12345.

[41] Z. Wei, J. Chen, M. Goldblum, Z. Wu, T. Goldstein, and Y.-G. Jiang, "Towards transferable adversarial attacks on vision transformers," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 2668–2676.

[42] Z. Wei, J. Chen, Z. Wu, and Y.-G. Jiang, "Boosting the transferability of video adversarial examples via temporal translation," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 2659–2667.

[43] K. Xu et al., "Structured adversarial attack: Towards general implementation and better interpretability," in *Proc. Int. Conf. Learn. Represent.*, 2019.

[44] H. Yan and X. Wei, "Efficient sparse attacks on videos using reinforcement learning," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 2326–2334.

[45] C. Yang, Y. Xu, J. Shi, B. Dai, and B. Zhou, "Temporal pyramid network for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 588–597.

[46] J. Yang, Y. Jiang, X. Huang, B. Ni, and C. Zhao, "Learning black-box attackers with transferable priors and query feedback," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 12288–12299.

[47] J. Yang et al., "Neural aggregation network for video face recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5216–5225.

[48] D. Yin, R. G. Lopes, J. Shlens, E. D. Cubuk, and J. Gilmer, "A Fourier perspective on model robustness in computer vision," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 13255–13265.

[49] H. Zhang, L. Zhu, Y. Zhu, and Y. Yang, "Motion-excited sampler: Video adversarial attack with sparked prior," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 240–256.

[50] Q. Zhang, Y. Ding, Y. Tian, J. Guo, M. Yuan, and Y. Jiang, "AdvDoor: Adversarial backdoor attack of deep learning system," in *Proc. 30th ACM SIGSOFT Int. Symp. Softw. Testing Anal.*, 2021, pp. 127–138.