Sparse Adversarial Perturbations for Videos

Xingxing Wei, Jun Zhu,* Sha Yuan, Hang Su
Dept. of Comp. Sci. & Tech., Institute for Artificial Intelligence,
State Key Lab for Intell. Tech. & Sys., THBI Lab, Tsinghua University, Beijing, China
{xwei11, dcszj, yuansha, suhangss}@mail.tsinghua.edu.cn

Abstract

Although adversarial samples of deep neural networks (DNNs) have been intensively studied on static images, their extensions in videos are never explored. Compared with images, attacking a video needs to consider not only spatial cues but also temporal cues. Moreover, to improve the imperceptibility as well as reduce the computation cost, perturbations should be added on as few frames as possible, i.e., adversarial perturbations are temporally $sparse$. This further motivates the $propagation$ of perturbations, which denotes that perturbations added on the current frame can transfer to the next frames via their temporal interactions. Thus, no (or few) extra perturbations are needed for these frames to misclassify them. To this end, we propose the first white-box video attack method, which utilizes an $l_{2,1}$-norm based optimization algorithm to compute the sparse adversarial perturbations for videos. We choose the action recognition as the targeted task, and networks with a CNN+RNN architecture as threat models to verify our method. Thanks to the propagation, we can compute perturbations on a shortened version video, and then adapt them to the long version video to fool DNNs. Experimental results on the UCF101 dataset demonstrate that even only one frame in a video is perturbed, the fooling rate can still reach 59.7%.

Introduction

In the past decade, Deep Neural Networks (DNNs) have shown great superiority in computer vision tasks, like image recognition (He et al. 2016), image restoration (Dong et al. 2014) and visual tracking (Wang and Yeung 2013). Although DNNs obtain the state-of-the-art performance in these tasks, they are known to be vulnerable to adversarial samples (Szegedy et al. 2013), i.e., the images with visually imperceptible perturbations that can mislead the network to produce wrong predictions. The adversarial samples are usually calculated by the Fast Gradient Sign Method (FGSM) (Goodfellow, Shlens, and Szegedy 2014) and optimization-based methods (Moosavi-Dezfooli et al. 2016). One reason for adversarial samples is that they are fell on some areas in the high-dimensional feature space which are not explored during training. Thus, investigating adversarial samples not only helps understand the working mechanism of deep networks, but also provides opportunities to improve the networks' robustness (Xie et al. 2017; Dong et al. 2018; Ma, Xu, and Cao 2019).

Up to now, many studies about adversarial samples have been investigated, such as adversarial perturbations for a single image (Moosavi-Dezfooli, Fawzi, and Frossard 2016), universal adversarial perturbations (Moosavi-Dezfooli et al. 2016) and adversarial samples for object detection and segmentation (Xie et al. 2017). However, these studies are all based on images, while leaving videos unexplored. Investigating adversarial samples on videos is of both theoretical and practical values, as deep neural networks have been widely applied in video analysis tasks (Donahue et al. 2017; Nguyen, Yosinski, and Clune 2015; Wang et al. 2016).

Technically, the main difference between videos and images lies in the temporal structure contained in videos. Therefore, a properly designed attacking method should explore the temporal information to achieve efficiency and effectiveness. We expect that the perturbations added on one frame can propagate to other frames via temporal interactions, which will be called the $propagation$ of perturbations. Besides, a video have many frames, computing perturbations for each frame is time-consuming, and actually not necessary. Whether it is possible that perturbations are added on only few frames, and then are propagated to other frames to misclassify the whole video. In this way, the generated adversarial videos also have high imperceptibility and are hard to be detected. Because perturbations are added on sparse frames rather than the whole video, we call it the $sparsity$ of perturbations. Actually, the $propagation$ and $sparsity$ interact with each other, $propagation$ helps boost the $sparsity$, meanwhile the $sparsity$ constraint will lead to better $propagation$.

For these reasons, in this paper, we aim to attack the video action recognition task (Poppe 2010), where the temporal cue is a key component for the predicted label. This is naturally suitable to explore the temporal adversarial perturbations. For the threat model, we choose the networks with a CNN+RNN architecture, which is widely used in action recognition, such as Long-term Recurrent Convolu-

*Corresponding Author

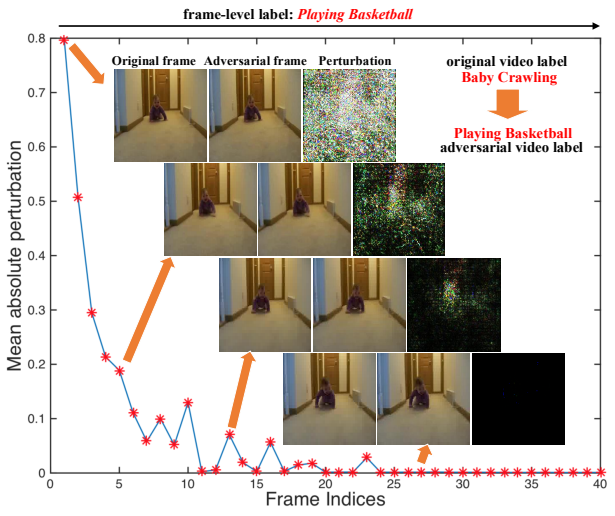

Figure 1: An illustration of our output for a video from UCF101 dataset with label of BabyCrawling. The computed perturbations successfully fool DNNs to output label of PlayingBasketball. The Mean Absolute Perturbation (MAP) of each frame is plotted. From the figure, we see that MAP values are significantly reduced along with the varying frames. In the final 20 frames, they fall into almost zero, showing the $sparsity$ of perturbations. Note that, though these frames have no perturbations, they still make DNNs predict wrong label for each frame (see the top arrow line). That's to say, the perturbations from previous frames $propagate$ here and show their power. Four triples indicate the results from the 1, 5, 13, 27th frames, respectively. In each triple, we see the adversarial frame is the same to the original frame in the appearance. We better visualize the perturbations with increasing their amplitudes $\times$ 255.

tional Network (LRCN) (Donahue et al. 2017) or network in (Yue-Hei Ng et al. 2015). To achieve $sparsity$. we apply an $l_{2,1}$-norm regularization on perturbations during the optimization. The $l_{2,1}$-norm uses the l_1 norm across frames, and thus, enforces to select few key frames to add perturbations. As for $propagation$, we find perturbations show good propagation under the $l_{2,1}$ constraint within the recurrent neural network (such as Vanilla RNN, LSTM and GRU) because of the interaction with $sparsity$. Another advantage of the propagation is that we can compute perturbations on a shortened version video, and then adapt them to the long version video to fool DNNs, which provides a more efficient method to attack videos.

It is noteworthy that, we combine the $l_{2,1}$-norm with RNN to **jointly** design the video attack method, rather than the single $l_{2,1}$-norm. The $l_{2,1}$-norm doesn't directly encode the temporal structure. Instead, it is the intrinsic motivation for perturbation propagation within the RNN network (while propagation under l_2-norm is limited, see Fig.4). The illustrations of our output and method are given in Fig. 1 and Fig. 2, respectively.

In summary, this paper has the following contributions:

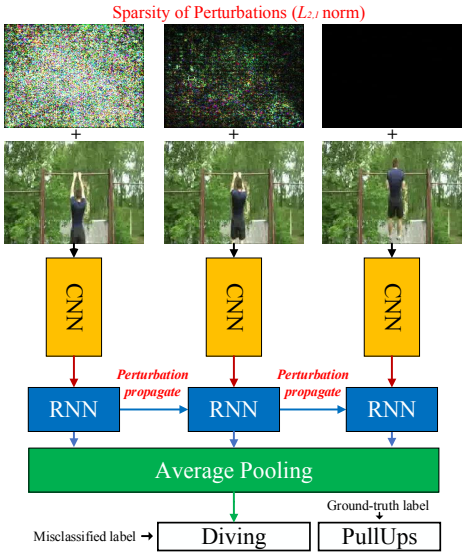

Figure 2: The illustration of our method. An $l_{2,1}$ norm is used during the optimization, which enforces the sparsity of computed perturbations. Within the CNN+RNN architecture (action recognition network), perturbations are encoded after the CNN, and then propagate to the next frames via RNN, finally resulting in the misclassified label for the whole video. Thanks to the propagation of perturbations, perturbations added on the final frames fall into zeros.

- To our knowledge, we are the first to explore white-box adversarial samples in videos, and further, propose the video attack method. Considering the specific $sparsity$ and $propagation$ of video adversarial perturbations, we propose an $l_{2,1}$-norm regularization based optimization algorithm. We verify our method and evaluate its transferability on the UCF101 dataset.
- We give a comprehensive evaluation of the $sparsity$ and $propagation$ of perturbations, and furthermore, propose the propagation-based method for adversarial videos, i.e., computing perturbations on a shortened version video, and then adapt them to the long version video. We also find that LSTM and GRU are easier to be attacked than Vanilla RNN, because LSTM and GRU can represent long memory, which is favor to the perturbation propagation (see experiments).

The rest of this paper is organized as follows. In Section 2, we briefly review the related work. We present our algorithm in Section 3. Section 4 reports all experimental results. Finally, we summarize the conclusions in Section 5.

Related Work

The related work comes from two aspects: action recognition with deep learning and adversarial attack.

Action Recognition with Deep Learning

Action recognition is a core task in computer vision, where its goal is to predict a video-level label when given a video

clip (Poppe 2010). With CNNs achieving state-of-the-art performance on image recognition, many works have looked into designing effective deep CNNs for action recognition. For example, some approaches aim at fusing CNN features extracted on spatial frames with the temporal interactions (Karpathy et al. 2014). To integrate temporal information, CNN+RNN based models, which use a CNN to extract frame features and a RNN to integrate features over time, are presented to recognize activities (Donahue et al. 2017; Nguyen, Yosinski, and Clune 2015). Optical flow is also useful to encode the temporal cue. For that, two stream CNNs with one stream for static images and the other stream for optical flows are proposed to combine the information of object appearance and temporal motions (Simonyan and Zisserman 2014). Temporal Segment Networks (TSN) choose frames and optical flow on different time segments to extract information (Wang et al. 2016). In our paper, to better explore how the perturbations change along with the time, we choose the networks with a CNN+RNN architecture as the threat model.

Adversarial Attack

Generating adversarial examples for image classification has been extensively studied recently. Szegedy et al. [2013] first find that adversarial examples can make CNNs predict a wrong label with high confidence while the adding perturbations to the original images are visually imperceptible, Goodfellow, Shlens, and Szegedy [2014] propose a simple Fast Gradient Sign Method (FGSM) to generate adversarial examples based on the linear nature of CNNs. Moosavi-Dezfooli et al. [2016a] first show the existence of universal adversarial perturbations. Moosavi-Dezfooli, Fawzi, and Frossard [2016b] present a simple algorithm to compute the minimal adversarial perturbation by assuming that the loss function can be linearized around the current data point. Liu et al. [2016] study the transferability of both non-targeted and targeted adversarial examples, and propose an ensemble-based approaches to generate adversarial examples with stronger transferability. Baluja and Fischer [2017] train a network to generate adversarial examples for a particular threat model. Kurakin, Goodfellow, and Bengio [2016] show that the adversarial examples also exist in the physical-world machine learning system. The above papers are all based on images, while we focus on video adversarial samples, which bring new challenges.

Hosseini, Xiao, and Poovendran [2017] exploit a simple inserted mechanism to fool the Google's Cloud Video Intelligence API, which is an experimental manner to perform attack, and cannot give a detailed explanation about how it works. Our paper gives a detailed study of the CNN+RNN architecture widely used in action recognition, and then use an $l_{2,1}$-norm based optimization method to accomplish the sparsity and propagation of video perturbations, which is more interpretable. Moreover, our paper gives more experiments about the transferability across the network and videos, which is more comprehensive.

Methodology

In this section, we introduce the proposed $l_{2,1}$-norm based algorithm for video adversarial samples. Our method is an optimization-based approach.

Let $\mathbf{X} \in \mathbb{R}^{T\times W\times H\times C}$ denote a clean video, and $\hat{\mathbf{X}}$ denote its adversarial video, where T is the number of frames, W, H, C are the width, height, and channel for a specific frame, respectively. $\mathbf{E} = \hat{\mathbf{X}} - \mathbf{X}$ is the adversarial perturbations. To generate non-targeted adversarial examples, we approximate the solution to the following objective function:

$$\arg\min_{\mathbf{E}} \lambda||\mathbf{E}||_{2,1} - \ell(\mathbf{1}_y, J_\theta(\hat{\mathbf{X}})), \tag{1}$$

where $\ell(\cdot,\cdot)$ is the loss function to measure the difference between the prediction and the ground truth label. In this paper, we choose the widely used cross-entropy function $\ell(u, v) = \log(1-u\cdot v)$, which is shown to be effective (Carlini and Wagner 2017). $J_\theta(\cdot)$ is the threat model with parameters θ. $\mathbf{1}_y$ is the one-hot encoding of the ground truth label y. $||\mathbf{E}||_{2,1}$ is the $\ell_{2,1}$ norm of $\mathbf{E}$, which is a metric to quantify the magnitude of the perturbation. λ is a constant to balance the two terms in the objective.

To obtain a universal adversarial perturbation across videos, we solve the following problem:

$$\arg\min_{\mathbf{E}} \lambda||\mathbf{E}||_{2,1} - \frac{1}{N}\sum_{i=1}^{N} \ell(\mathbf{1}_{y_i}, J_\theta(\hat{\mathbf{X}}_i)), \tag{2}$$

where N is total number of training videos, and $\hat{\mathbf{X}}_i$ is the i-th adversarial video.

To better control the sparsity and study the perturbation propagation across frames, we add a temporal mask on the video to enforce some frames having no perturbations. The problem is modified as follows:

$$\arg\min_{\mathbf{E}} \lambda||\mathbf{M}\cdot\mathbf{E}||_{2,1} - \frac{1}{N}\sum_{i=1}^{N} \ell(\mathbf{1}_{y_i}, J_\theta(\mathbf{X}_i + \mathbf{M}\cdot\mathbf{E})), \tag{3}$$

where $\mathbf{M} \in \{\mathbf{0}, \mathbf{1}\}^{T\times W\times H\times C}$ is the temporal mask. We let $\Theta = \{1, 2, ..., T\}$ be the set of frame indices, Φ is a subset with K elements within Θ, and $\Psi = \Theta - \Phi$. If $t \in \Phi$, we set $M_t = 0$, and if $t \in \Psi$, $M_t = 1$, where $M_t \in \{\mathbf{0}, \mathbf{1}\}^{W\times H\times C}$ is the t-th frame in $\mathbf{M}$. In this way, we enforce the computed perturbations to be added only on the selected video frames. We here regard $S = \frac{K}{T}$ as the sparsity.

If the goal is to generate targeted adversarial examples (i.e., the misclassified label is set to the pre-fixed label, which is called target label), the problem can be modified as follows:

$$\arg\min_{\mathbf{E}} \lambda||\mathbf{M}\cdot\mathbf{E}||_{2,1} + \frac{1}{N}\sum_{i=1}^{N} \ell(\mathbf{1}_{y_i^*}, J_\theta(\mathbf{X}_i + \mathbf{M}\cdot\mathbf{E})), \tag{4}$$

where y_i^* is the targeted label. Eq.(4) outputs the perturbations to make $J_\theta(\cdot)$ predict y_i^* with a high probability.

Problem (3,4) are the final objective function proposed by us to solve the video attack task. Some detailed information of Eq.(3,4) is described as follows:

Perturbation Regularization The $l_{2,1}$-norm in problem (3,4) is a metric to quantify the magnitude of the perturbation. As mentioned before, we hope that the perturbations are added on as fewer frames as possible. Therefore, we choose $l_{2,1}$ norm to meet this goal, where is widely used in sparse coding methods (Wright et al. 2009; Yang et al. 2010). $||\mathbf{E}||_{2,1} = \sum_t^T ||E_t||_2$, where $E_t \in \mathbb{R}^{W\times H\times C}$ is the *t*-th frame in $\mathbf{E}$. $l_{2,1}$ norm apply the l_1 norm across the frames, and thus, can ensure the sparsity of generated perturbations. In the experiment, we also show the results using l_2 norm, as the comparison with the $l_{2,1}$ norm.

Threat Model In action recognition, the current state-of-the-art approach is the two-stream model (Donahue et al. 2017), i.e., one stream is to capture the RGB frames, and another stream is to capture the optical flow images (motion information) between two adjacent RGB frames. The outputs from these two streams are fused to predict the final label with various kinds of fusion methods. These two streams usually have the same network architecture, where one choice is CNN+Pooling, and another is CNN+RNN architecture. Compared with CNN+Pooling, CNN+RNN can encode the temporal information. In our paper, we regard the networks with CNN+RNN architecture as the threat model $J_\theta(\cdot)$. The results of attacking CNN+Pooling also are reported for comparisons. We give the illustration of the CNN+RNN model in Fig. 2, where video frames are firstly input to CNN, and then, fed to RNN. The final label is predicted via an average pooling. Note that, the CNN and RNN in the figure are the general terms for the spatial and temporal networks, respectively. CNN can be specified as ResNet, Inception V3, etc, and RNN as LSTM, GRU, etc.

Training Problems (3,4) are easy to solve. Any Stochastic Gradient Descent (SGD) algorithm can solve them. Here, we use the Adam (Kingma and Ba 2014) algorithm to get the results. Because $l_{2,1}$ norm is used, initializing the perturbations with zeros will lead to NaN values. We instead initialize them using a small value. In the experiments, we use 0.0001. After some iterations, the perturbations will converge to a sparse result. λ in problem (3,4) is set to a constant, which is tuned in the training set.

Temporal Mask Although $l_{2,1}$ norm ensures the sparsity of perturbations, the specific number of polluted frames is totally decided by the optimization algorithm, and cannot be designated in advance by users. However, this is usually useful in practice. By adding the temporal mask in optimization process, we can simply sample some preferring frames to align perturbations according to different settings (For example, temporal mask $\mathbf{M}$ can be predefined according to the needed sparsity), and observe their results still under $l_{2,1}$ norm constraint. Actually, Eq.2 is a special case of Eq.3, where $\mathbf{M}$ equals to all ones. The temporal mask makes our framework more flexible. We investigate some candidate choices of $\mathbf{M}$, and give the corresponding discussions about its impact to the proposed method (see experiments).

Experiments

In this section, we give the experiments from three aspects.

Datasets and Metrics

Datasets: We choose the widely used dataset in action recognition: $\mathbf{UCF101}$ (Soomro, Zamir, and Shah 2012). It contains 13,320 videos with 101 action classes covering a broad set of activities such as sports, musical instruments, body-motion, human-human interaction, human-object interaction. The dataset splits more than 8000 videos in the training set, and more than 3000 videos in the testing set. Because there are no other existing methods for video adversarial samples, we can only compare with the methods based on images, i.e., computing perturbations for each frame (Moosavi-Dezfooli, Fawzi, and Frossard 2016) in a video. This setting is coincident with the outputs using Eq.(1) with l_2 norm, which are reported as the comparisons.

Metrics: We use three metrics to evaluate various aspects.

$\mathbf{Fooling\ ratio}$ $(\mathbf{F})$: is defined as the percentage of adversarial videos that are successfully misclassified (Moosavi-Dezfooli et al. 2016).

$\mathbf{Perceptibility}$ $(\mathbf{P})$: denotes the perceptibility score of the adversarial perturbation $\mathbf{r}$. We here use the Mean Absolute Perturbation (MAP): $P = \frac{1}{N}\sum_i |\mathbf{r}_i|$, where N is the number of pixels, and $\mathbf{r}_i$ is the intensity vector (3-dimensional in the RGB color space).

$\mathbf{Sparsity}$ $(\mathbf{S})$: denotes the proportion of frames with no perturbations (clean frames) versus all the frames in a specific video to fool DNNs. $S = \frac{K}{T}$, where K is the number of clean frames, and T is the total number of frames in a video.

Perturbation Propagation

In this section, we give the experimental results about the perturbation propagation.

Visualization for Perturbations We firstly give the visualization of perturbations computed using Eq.(2) with $l_{2,1}$ norm, which are universal perturbations across videos. In Fig. 3, we see that the adversarial videos are not distorted by the perturbations, and are imperceptible to human eyes. Furthermore, the perturbations show the sparse property (black means no perturbations), i.e., they are reduced across frames along with the time, which is owing to the used $l_{2,1}$ norm. In the next section, we will discuss the propagation of perturbations, inspired by these sparse results.

Perturbation Propagation To show the perturbation propagation, we give four examples outputted by Eq.(1) with $l_{2,1}$ norm in Fig. 4 (see the blue line with stars), where we see the computed perturbations successfully fool the action recognition networks (for example, in the first case, a clean video with label of Bench Press is identified as Lunges after adding perturbations). Correspondingly, the original frame-level labels (red dotted line) are also misclassified as wrong labels (black dotted line). By contrast, the Mean Absolute Perturbation (MAP) value of each frame is reduced significantly along with the time. In the last few frames, they fall into almost zeros. That's to say, although few perturbations are added on these frames, the perturbations from the previous frames propagate here, and help fool the DNNs. As a comparison, we also list the results of Eq.(1) with l_2 norm in Fig. 4 (see the magenta line with circles). In this figure,

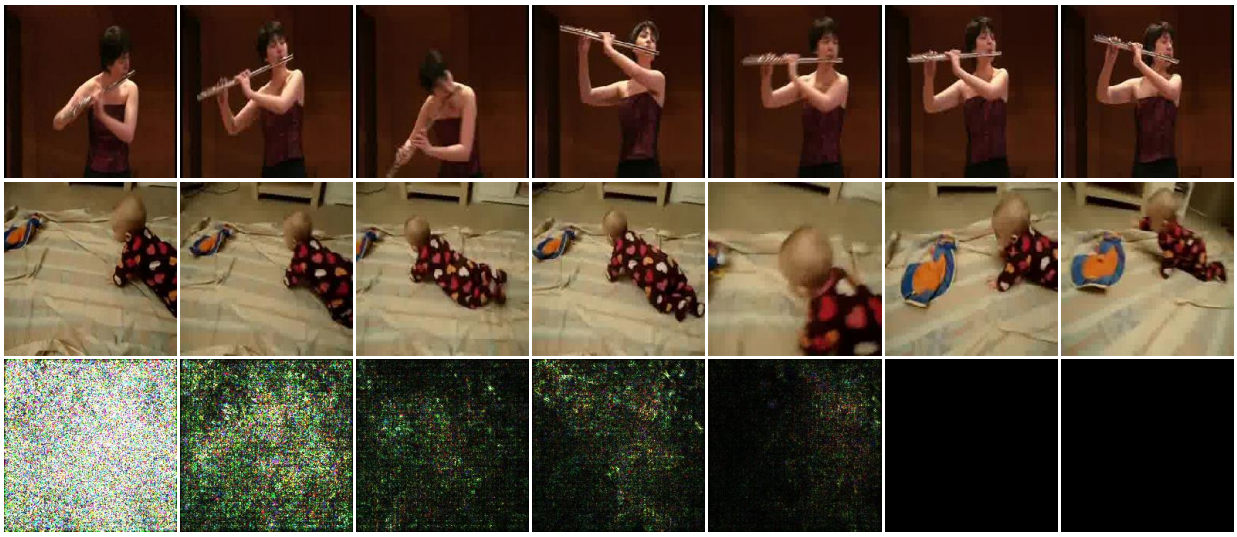

Figure 3: The computed adversarial videos (top two rows) and their corresponding perturbations (bottom row) using Eq.(2) with $l_{2,1}$ norm. We better visualize the perturbations with increasing their amplitudes with $\times$ 255. For more discussions, please see the texts.

the MAP value is also reduced across frames, which further demonstrates the perturbation propagation. The difference is, the output of $l_{2,1}$ norm is sparse, which reveals that the frames ranking behind the video line actually need few (even no) perturbations to fool DNNs with the help of propagation. But l_2 norm cannot show this property.

Inspired by the sparsity of $l_{2,1}$ norm, we directly enforce perturbations not to be added on the frames ranking behind the video line. To this end, we add the temporal mask during the optimization process using Eq.3. Here we only select the top 8 frames to compute their perturbations, and let the other frames be clean. The experimental results on the same videos are listed in Fig. 5. We find that the frames are still predicted as wrong labels. Furthermore, the MAP values of these frames also show a decreasing trend. It further demonstrates the propagation of perturbations. Otherwise, these clean frames cannot be predicted as wrong labels. Note that in the forth case in Fig. 5, the final 4 frames have correct labels, which shows perturbations will reduce its effect along with the time, and cannot propagate forever.

We also randomly select some frames to align perturbations to perform the video attack (i.e., we randomly set the elements of M to 1, and other elements as 0). Four experimental results under different sparsities are given in Fig. 6, where we see that the video attack still succeeds to fool the DNN, and the perturbations show good propagation under different sparsities.

Table 1: The results of fooling rates versus different sparsities.

S	0%(40)	80%(8)	90%(4)	97.5%(1)
F	100%	100%	91.8%	59.7%
P	0.0698	0.6145	1.0504	1.9319

We now gradually enlarge the sparsity S in Eq.(3), and observe the change of Fooling ratio F in the testing set on UCF101 dataset. High sparsity S means more clean frames, and less adversarial frames in the video. We give the quantitative results of fooling rates versus different sparsities in Table. 1. In the table, we list four sparsities (S) and their corresponding Fooling rates (F) as well as perceptibility scores (P). Taking $90\%(4)$ as an example, $90\% = 1 - \frac{4}{40}$, where 4 is top four polluted frames, and 40 is the total number of frames in the video. The results in Table 1 show that even only one frame is polluted ($S = 97.5\%$), the Fooling rate can also reach 59.7%. To achieve the 100% fooling rate, the least number of polluted frames is 8 ($S = 80\%$) on the used dataset. We also see that the perceptibility score is gradually increasing with the rise of sparsity score, and reaches the top in $S = 97.5\%$. This is reasonable because large perturbations can spread to more frames. The polluted top one frames in $S = 97.5\%$ and their corresponding clean frames are illustrated in Fig. 7, where we see that despite the largest $P = 1.9319$, the adversarial frames are the same to the clean frames, which are not perceptible to human eyes.

Adversarial Video based on Propagation Thanks to the perturbation propagation, we don't need to compute perturbations based on the whole video. Instead, we can compute perturbations on a shorten version video, and then adapt them to the long version video. In this way, the computation cost is reduced significantly. We report the time of computing perturbations for various frames in Table 2, where we see the computing time is linearly reduced with the rise of sparsity, showing that computing perturbations on a shorten version video can reduce computation cost.

Specifically, to fool the action recognition network for a given video, we first choose the top N frames $\{F_1, ..., F_N\}$ from the original video , and then use Eq.(1) (for a single

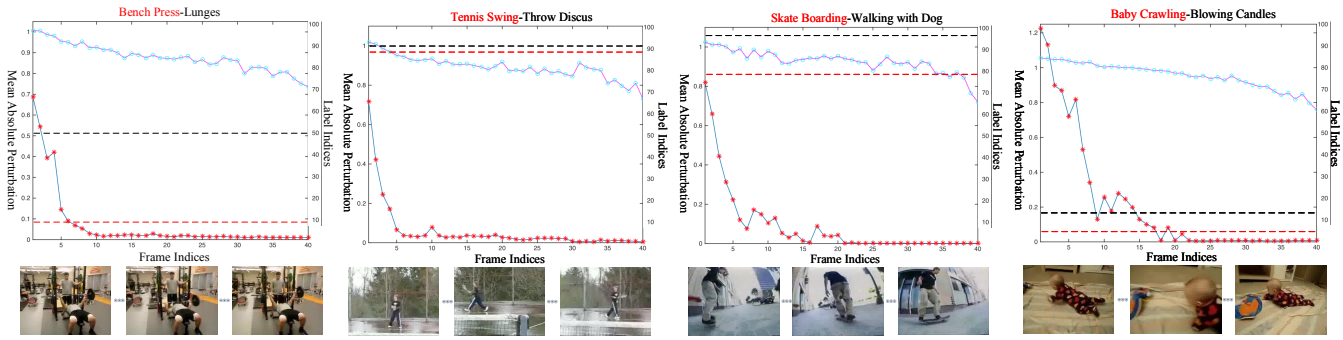

Figure 4: Four examples for showing perturbation propagation on UCF101 dataset. The x-axis denotes the frame indices in a video. The left y-axis denotes the Mean Absolute Perturbation (MAP) value of each frame's perturbations, and the right y-axis is the label indices. The blue line with stars is the curve of MAP values with $l_{2,1}$ norm, and magenta line with circles is the result with l_2 norm. The red dotted line is the predicted frame-level label indices for the clean video, and black dotted line is the predicted frame-level label indices for the adversarial video, both by the action recognition networks (the video-level labels are listed in the top of each figure with the same color). In the bottom of each figure, we give the corresponding video frames. For detailed discussions, please see the texts.

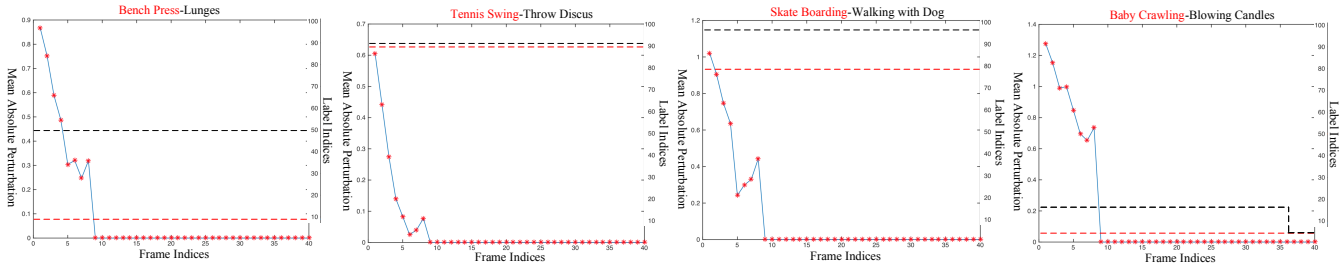

Figure 5: Four examples of showing perturbation propagation on UCF101 dataset. The difference with Fig. 4 lies in the integration of temporal mask proposed in Eq.(3). For detailed discussions, please see the texts.

video) or Eq.(2) (for getting universal perturbations) with l_2 norm to compute their adversarial frames $\{\hat{F}_1, ..., \hat{F}_N\}$. Finally, we replace $\{F_1, ..., F_N\}$ with $\{\hat{F}_1, ..., \hat{F}_N\}$ in the original video. This modified video is then input to the action recognition networks. Note that, we here don't use the $l_{2,1}$ norm. Because the $l_{2,1}$ norm will result in the sparse perturbations during these N frames, which are not good for further propagation to the rest clean frames. We plot the comparisons between l_2 and $l_{2,1}$ norm in this setting in Fig. 8. In this figure, we see the performance of l_2 norm is advantageous to $l_{2,1}$ norm. In the next section, we will give the detailed evaluations and discussions of this method. In default, we set $N = 20$ and use l_2 norm in the following experiments.

Table 2: Time for computing perturbations in one iteration.

S	0%	50%	75%	87.5%	97.5%
Time	2.853s	1.367s	0.612s	0.346s	0.0947s

Performance and Transferability

In this section, we evaluate the performance and transferability of the propagation based method.

Transferability across Models We firstly evaluate the transferability of computed perturbations. Because the transferability of CNN networks has been studied in many literatures, we here mainly explore the RNN networks, including Vanilla RNN, LSTM, and GRU. Besides, the results of CNN + Average Pooling (removing the RNN layer in Fig. 2) are also reported. The Fooling rates in different settings are given in Table 3, where we use the networks in rows to generate perturbations, and networks in columns evaluate the transferability. Form the table, we draw the following conclusions: 1. The diagonals have largest values. It is reasonable because they perform the white-box attack in this setting. 2. In the off-diagonals, the values are all above 65%, which shows perturbations in videos have good transferability, especially in the RNN models. 3. In the off-diagonals, the $Pooling$ column has the poor performance. Pooling method has no memory like LSTM or GRU, and thus, the perturbations cannot propagate to other frames, resulting in the poor performance. 4. By contrast, the GRU and $LSTM$ columns have better performance than VanillaRNN. As we known, GRU and LSTM can represent long memory, this demonstrates long memory is favor to the propagation of perturbations, and thus GRU and LSTM are easier to be attacked than VanillaRNN.

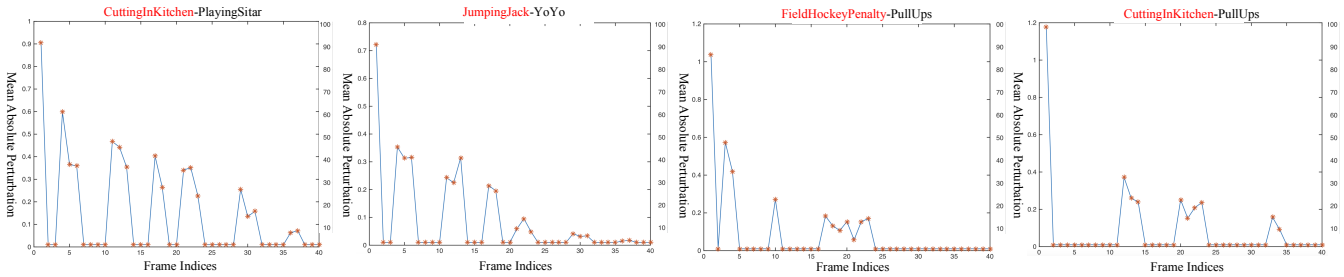

Figure 6: Perturbations can be added on any discrete frames with a random margin. These are four adversarial examples under different sparsities conducted on UCF101 dataset. For detailed discussions, please see the texts.

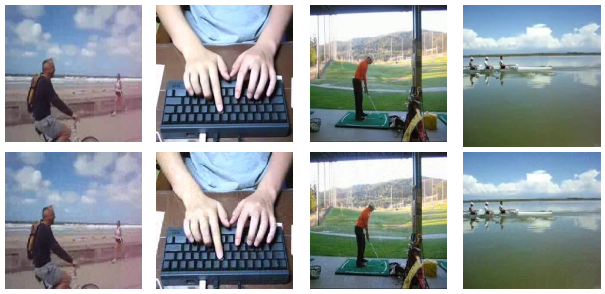

Figure 7: Four examples of the polluted top one frame in $S = 97.5\%$. The top row is the original clean frames, and the bottom row is the adversarial frames.

Table 3: Fooling rates in different settings on UCF101 dataset.

Models	VanillaRNN	LSTM	GRU	Pooling
VanillaRNN	95.2%	95.2%	95.2%	71.0%
LSTM	84.1%	100%	97.1%	76.8%
GRU	81.8%	92.4%	100%	66.7%
Pooling	84.1%	96.8%	95.2%	87.3%

Transferability across Videos We also evaluate the transferability of perturbations across videos. Th universal perturbations are computed using Eq.(2) on training set, and then are added to the testing videos to evaluate their transferability. The visualization of universal perturbations can be found in Fig. 3. The performance (Fooling rate) is listed in Table 4, where shows the results of our method has good transferability across videos (achieving the 95.2% fooling rate on the testing set). In other words, the universal perturbations can make new videos fool the action recognition networks.

Table 4: Performance of the cross-videos attack.

Metric	Training set	Testing set
Fooling rate (F)	100%	95.2%

Conclusions

In this paper, we explored the adversarial perturbations for videos. An $l_{2,1}$-norm based optimization algorithm was proposed to solve this problem. The $l_{2,1}$ norm applied the l_1 norm across frames, and thus, could ensure the *sparsity* of perturbations. A serial of experiments conducted on UCF101 dataset demonstrated that our method had better transferability across models and videos. More importantly, our method showed the *propagation* of perturbations under the $l_{2,1}$ constraint within the CNN+RNN architecture. According to this observation, we further presented the efficient method for adversarial videos based on the perturbation propagation. In the future, we will look into the defense methods for video attacks. Because adversarial perturbations may be added on any frame in a video, the greedy algorithm, that can efficiently deal with all the frames in a video, is a reasonable and reliable choice for the defense. For that, we will investigate an efficient denoising method, and put it as the pre-processing step before video classification networks.

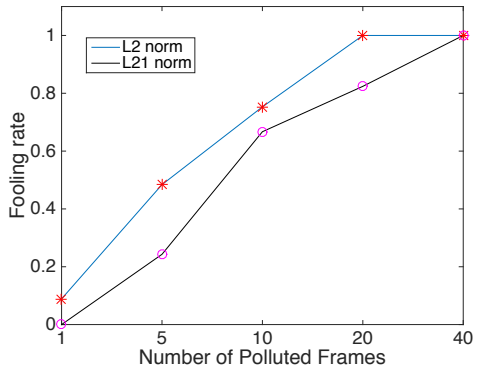

Figure 8: Comparisons between l_2 and $l_{2,1}$ norm versus Fooling rate on UCF101 dataset. We here report the results when $N = 1, 5, 10, 20, 40$, respectively. The total number of frames is 40.

Acknowledgements

This work was supported by the National Key Research and Development Program of China (No.2017YFA0700904), the NSFC Projects (Nos. 61806109, 61806111, 61620106010, 61621136008, 61332007), Project funded by China Postdoctoral Science Foundation, the MIIT Grant of Int. Man. Comp. Stan (No. 2016ZXFB00001), Tsinghua Tiangong Institute for Intelligent Computing, the NVIDIA NVAIL Program and a Project from Siemens.

References

Baluja, S., and Fischer, I. 2017. Adversarial transformation networks: Learning to generate adversarial examples. *arXiv preprint arXiv:1703.09387*.

Carlini, N., and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (SP)*, 39–57.

Donahue, J.; Anne Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; and Darrell, T. 2017. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2625–2634.

Dong, C.; Loy, C. C.; He, K.; and Tang, X. 2014. Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision*, 184–199.

Dong, Y.; Liao, F.; Pang, T.; Su, H.; Zhu, J.; Hu, X.; and Li, J. 2018. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 9185–9193.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hosseini, H.; Xiao, B.; and Poovendran, R. 2017. Deceiving google's cloud video intelligence api built for summarizing videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 1–5.

Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; and Fei-Fei, L. 2014. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 1725–1732.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*.

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

Ma, K.; Xu, Q.; and Cao, X. 2019. Robust ordinal embedding from contaminated relative comparisons. In *AAAI Conference on Artificial Intelligence*.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; Fawzi, O.; and Frossard, P. 2016. Universal adversarial perturbations. *arXiv preprint arXiv:1610.08401*.

Moosavi-Dezfooli, S.-M.; Fawzi, A.; and Frossard, P. 2016. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2574–2582.

Nguyen, A.; Yosinski, J.; and Clune, J. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 427–436.

Poppe, R. 2010. A survey on vision-based human action recognition. *Image and vision computing* 28(6):976–990.

Simonyan, K., and Zisserman, A. 2014. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, 568–576.

Soomro, K.; Zamir, A. R.; and Shah, M. 2012. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*.

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

Wang, N., and Yeung, D.-Y. 2013. Learning a deep compact image representation for visual tracking. In *Advances in neural information processing systems*, 809–817.

Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; and Van Gool, L. 2016. Temporal segment networks: Towards good practices for deep action recognition. In *European Conference on Computer Vision*, 20–36. Springer.

Wright, J.; Yang, A. Y.; Ganesh, A.; Sastry, S. S.; and Ma, Y. 2009. Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence* 31(2):210–227.

Xie, C.; Wang, J.; Zhang, Z.; Zhou, Y.; Xie, L.; and Yuille, A. 2017. Adversarial examples for semantic segmentation and object detection. In *IEEE International Conference on Computer Vision*.

Yang, J.; Wright, J.; Huang, T. S.; and Ma, Y. 2010. Image super-resolution via sparse representation. *IEEE transactions on image processing* 19(11):2861–2873.

Yue-Hei Ng, J.; Hausknecht, M.; Vijayanarasimhan, S.; Vinyals, O.; Monga, R.; and Toderici, G. 2015. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4694–4702.