國立臺灣大學電機資訊學院資訊工程學系

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

以對抗式物件合成訓練輔助動作辨識模型以減輕其偏差

Towards Robust Action Recognition via Adversarial Object Synthesis Training

劉哲宇

Zhe-Yu Liu

指導教授：徐宏民博士

Advisor: Winston Hsu, Ph.D.

中華民國 109 年 7 月

July, 2020

doi:10.6342/NTU202003008

國立臺灣大學碩士學位論文
口試委員會審定書

以對抗式物件合成訓練輔助動作辨識模型以減輕其偏差

Towards Robust Action Recognition via Adversarial Object Synthesis Training

本論文係劉哲宇君（學號 R07922031）在國立臺灣大學資訊工程學系完成之碩士學位論文，於民國 109 年 7 月 16 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

（指導教授）

陳永昇　　葉梅珍

系 主 任　莊永裕

摘要

一個好的動作辨識模型需要對人類或其他物體的移動模式有良好的了解。然而我們發現，即使是許多目前表現最好的模型，都會一定程度的利用周遭環境中的靜止物件來判斷當下發生的動作，而非使用該動作本身當作判斷依據。這種對周遭特定物件的依賴性，使得模型在應用到擁有不同物件分佈的環境中時，無法維持原來的表現，因爲許多動作像是「拿取」，不會跟固定的物件做連結。在此篇論文中，我們將上述問題稱爲物件謬誤依賴 (Fallacious Object Reliance, or FOR)，並且詳盡地討論了關於物件特徵偏差（object representation bias）在許多動作辨識資料集中造成的影響。我們提出了數個量化方法來測量 FOR 問題的嚴重性，並且提出了一個「對抗式物件合成訓練（AdvOST）」的方法來減輕模型的 FOR 問題。AdvOST 方法訓練了一個神經網路合成器，把各種物件的圖片合成到訓練資料集的影片中，並且該合成器在需要混淆動作辨識模型的同時做出合理的生成來通過另一個神經網路鑑別器的偵測。此方法驅使動作辨識模型去忽略無關的靜止物件線索，以此減輕 FOR 問題。我們的實驗發現 AdvOST 方法可幫助 I3D 跟 SlowFast 等頂尖的動作辨識模型在 EPIC-KITCHENS 與 HMDB51 資料集上獲得更好的表現。

關鍵字： 動作辨識、資料集偏差、對抗式訓練

doi:10.6342/NTU202003008

Abstract

The action recognition task requires agents to understand the motion performed by humans or objects. However, we found that recognition models tend to predict the action based on the surrounding static objects instead of the action itself. This dependency may hurt the robustness of such models when applied to new environments with different object distribution as many action classes could be associated with different subjects (e.g. ”take”). In this paper, we regard this problem as the *Fallacious Object Reliance (FOR)* issue and discuss the role that the object representation bias plays in different datasets. Based on the observation, we propose several metrics to measure the severity of the FOR issue. Moreover, we propose a new training procedure called Adversarial Object Synthesis Training (AdvOST) to mitigate this issue. AdvOST trains a synthesizer pasting objects onto training videos to obfuscate the classification model and uses a discriminator that regularizes the synthesizer to generate natural synthesis. This method forces the action model to ignore unrelated object clues and successfully reduces the FOR issue. Finally, we obtain decent accuracy improvement on the validation sets of the EPIC-KITCHENS using the state-of-the-art I3D and SlowFast after applied AdvOST. We also acquire consistently accurate improvement on the three splits of HMDB51 using I3D.

Keywords: Action Recognition, Dataset Bias, Adversarial Training

doi:10.6342/NTU202003008

Contents

doi:10.6342/NTU202003008

doi:10.6342/NTU202003008

List of Figures

doi:10.6342/NTU202003008

List of Tables

doi:10.6342/NTU202003008

Chapter 1

Introduction

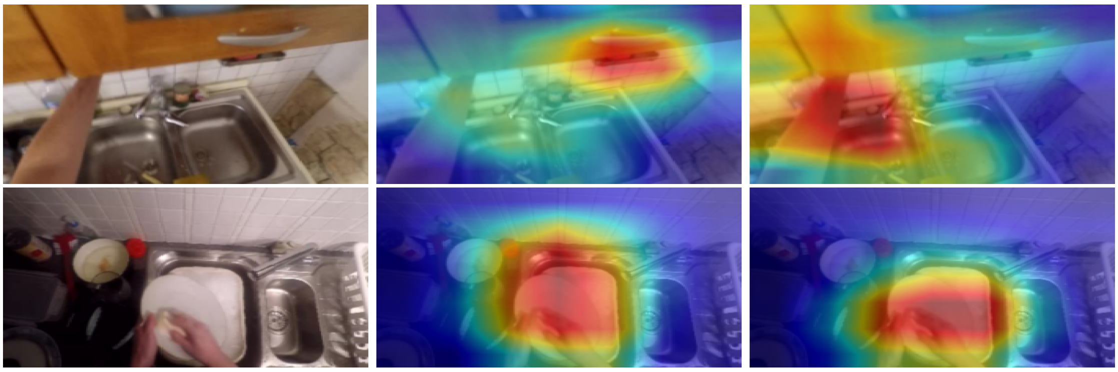

(a) Input Video (b) CAM of I3D (c) CAM of I3D with AdvOST

Figure 1.1: The grad-CAM [27] visualization of I3D model with and without our proposed AdvOST that helps alleviate the Fallacious Object Reliance. In the first row, the hand on the left is taking something behind the cabinet. The I3D model without AdvOST notices the handle-like object and predicts this action as ”close”, while I3D with AdvOST focuses on where the motion occurs and successfully predicts this action as ”take”. In the second row, although both models correctly predict the action as ”wash”, the one without AdvOST pays more attention on the appearance of the sink instead of the hands washing the dishes.

Deep learning has achieved significant progress in the image domain since the introduction of Convolutional Neural Network (CNN). Videos, as another commonly used data format in our daily lives, have also attracted lots of research attention in recent years. Similar to image classification, action recognition is one of the most fundamental problems among the various video understanding tasks like action localization [17, 14, 31] and captioning [21, 42]. Hence, lots of benchmarks [22, 32, 19, 28, 2, 5, 13] and deep models [29, 37, 34, 2, 9] are proposed for the action recognition task.

However, even for the state-of-the-art method I3D [2] and SlowFast [9], we still found

doi:10.6342/NTU202003008

action models sometimes utilize supposedly unrelated hints to infer action classes that are not tied to specific objects (e.g. ”take”). Two examples of the training set visualization in Fig. 1.1 suggest that the model concentrates on unrelated regions to make predictions. In our observation, this behavior diminishes the robustness of the trained model because it is highly likely that in new environments, these specific objects will not appear or will be associated with different actions.

A possible reason for this phenomenon is that spatio-temporal 3D CNN architectures focus more on **static clues** than **motion structures**. Authors of [35, 26, 9] also pointed out that the temporal dimension is essentially not symmetrical to the spatial dimensions, so it’s not ideal to simply add an extra dimension on the 2D CNN kernel to handle action recognition tasks. Hence, recent deep action models that achieve outperforming results usually have more carefully designed temporal modeling to process the complex and long-range temporal information [35, 26, 9, 16], but they still highly depend on static clues.

Besides the insufficient temporal modeling problem, Li *et al.*[23] also showed that there exist biases about objects, scenes, and people in the commonly used action datasets including HMDB51, UCF101, ActivityNet, and Kinetics, etc. These biases lead to erroneous conclusions when the dataset is not well-calibrated. Given the two facts that SOTA model architectures tend to focus on static images and datasets often have object biases, it becomes more of an issue for models to learn the motion essence of actions instead of the invalid dependencies of biased objects in the training set.

In this paper, we analyze this characteristic of action recognition models and propose a measurement called Object Reliance Level (ORL) to quantify this phenomenon. We also design a method to evaluate the object bias difference between training and testing sets and propose the Fallacious Object Reliance (FOR) score to measure the severity of wrong object-action association that hinders the robustness of action models in new environments (see 3.2 for details).

In addition to the above analysis, we further propose a training procedure called Adversarial Object Synthesis Training (AdvOST) to address the FOR issue. AdvOST augments the object diversity of training videos in an adversarial approach so the action recognition

doi:10.6342/NTU202003008

model could be trained to reduce FOR. Our method is tested on several benchmarks and proved to improve the performance and mitigate the issue we observed.

Overall, this paper makes two major contributions. First, we discuss and propose methods to measure the Object Reliance Level and FOR score of action models. Second, we propose a training structure called AdvOST to alleviate the FOR issue and obtain decent performance improvement.

doi:10.6342/NTU202003008

Chapter 2

Related work

2.1 Temporal Modeling of Action Recognition

Videos can be modeled as three-dimensional data containing 2 spatial and 1 temporal dimensions. A few works extract the temporal information using iDT [10] or optical flow [29] as auxiliary information with spatial data to make predictions. C3D [34] propose to use 3d convolutional kernels to process video data without pre-processing. Proposals that factorize C3D in order to solve its large parameter number issue are presented [26, 35, 40]. As long-range and complex temporal modeling gets more and more attention, network designs with different strategies to handle the temporal dimension have also been developed [41, 9, 38]. Our goal in this paper is to enhance the temporal modeling of action models, but we approach the task from the perspective of augmenting training data in an adversarial way.

2.2 Datasets Bias

Exploiting unintended dataset bias and mitigating the consequent issues are crucial for machine learning. For instance, the ethnic or gender bias in the datasets is studied for fairness [15, 1]. Li *et al.*[23] discusses the representation bias (in objects, scenes, and people) in action recognition datasets and mitigates the bias by resampling existing datasets. Instead, our proposed method forces action models to ignore irrelevant clues by dynamically

doi:10.6342/NTU202003008

increasing the object diversity without recollecting new data. Choi *et al.*[4] addresses the scene bias problem in action recognition by encouraging models to learn representation which is not able to predict scene types.

2.3 Cut-and-paste Synthesis

Synthetic labels are useful due to the expensiveness of human annotation. There are researches using cut-and-paste approaches to synthesize labels for object detection or tracking tasks [8, 7, 36, 11, 20]. The proposed method also uses this cut-and-paste approach; however our intention is not to produce labels but encourage models to ignore irrelevant objects.

2.4 Adversarial Learning

Adversarial training [33] was used to augment the diversity of training data with adversarial examples, which may increase the robustness of the model. Therefore, lots of tasks construct the network with such a learning framework like image synthesis, generative sampling and synthetic data generation [3, 6, 24, 30]. A-Fast-RCNN [39] modifies the features by spatial dropout to mimic occlusion and deformations. The ST-GAN approach [24] generates compositing images with geometric corrections for the purpose of warping the foreground image fitting the background image. [36] employs an adversarial learning paradigm to train their 3-way competition networks. We follow their adversarial manner to ensure the realistic of generated images are realistic. However, we composite uncorrelated objects onto input images to eliminate the effects caused by object reliance issue within datasets, instead of generating new or meaningful data for training.

doi:10.6342/NTU202003008

Chapter 3

Analysis of the Fallacious Object Reliance (FOR) Issue

As we have illustrated in Fig. 1.1, the main observation that motivates us to proposed AdvOST is that action models focus on static and unrelated object clues. We regard this phenomenon as the Fallacious Object Reliance (FOR) issue. In this section, we will discuss the role that dataset bias plays in this problem, measure how action models rely upon object hints, and find out when the object reliance property of models becomes problematic and hurt the robustness.

3.1 Object Reliance Level (ORL)

We believe the bias in action datasets is the source that encourages models to rely on unrelated clues. Fig. 3.2 shows one example of a repeated co-occurrence of an object and action in the training videos. This kind of association has been discussed in [23] and defined as a representation bias. Take objects in a dataset of an action dataset D as an example, they define the object representation bias:

$$B_{\text{obj}} = \log \frac{P(D, M_{\text{obj}\rightarrow\text{act}}(\theta_{\text{obj}}(D)))}{P_{\text{rand}}(D)} \tag{3.1}$$

Here, $\theta_{\text{obj}}(D)$ is the object representation of samples in D extracted by a pre-trained object classifier, and $M_{\text{obj}\rightarrow\text{act}}$ is another model trained on dataset D with $\theta_{\text{obj}}(D)$ as input. $P(D, M_{\text{obj}\rightarrow\text{act}}(\theta_{\text{obj}}))$ is the accuracy of $M_{\text{obj}\rightarrow\text{act}}$, and $P_{\text{rand}}(D)$ is the random guess accuracy of D. B_{obj} indicates how we can use only object information to predict action labels.

Using this formula, we can compare object representation biases among different datasets. However, it does not reveal if an action model $M_{\text{vid}\rightarrow\text{act}}$ actually depends on the bias. Therefore, we propose a new measurement that uses the **performance alignment** between $M_{\text{obj}\rightarrow\text{act}}$ and $M_{\text{vid}\rightarrow\text{act}}$ to quantify the *Object Reliance Level (ORL)* of an action model. The idea is that if the action model $M_{\text{vid}\rightarrow\text{act}}$ heavily relies on objects, its behavior will be similar to $M_{\text{obj}\rightarrow\text{act}}$, which uses only object representation as its input.

To compute performance alignment, we have to break the performance measure P in eq. 3.1 into per-group performance P_k (e.g. grouped by action class and compute the f1 score for each action), where k denotes the group index. Then, we pick an alignment measurement method A. For example, one good choice is the Pearson Correlation Coefficient:

$$A_{\text{corr}}(x, y) = \frac{\sum_{k=1}^{K}(x_k - \overline{x})(y_k - \overline{y})}{\sqrt{\sum_{k=1}^{K}(x_k - \overline{x})^2(y_k - \overline{y})^2}} \tag{3.2}$$

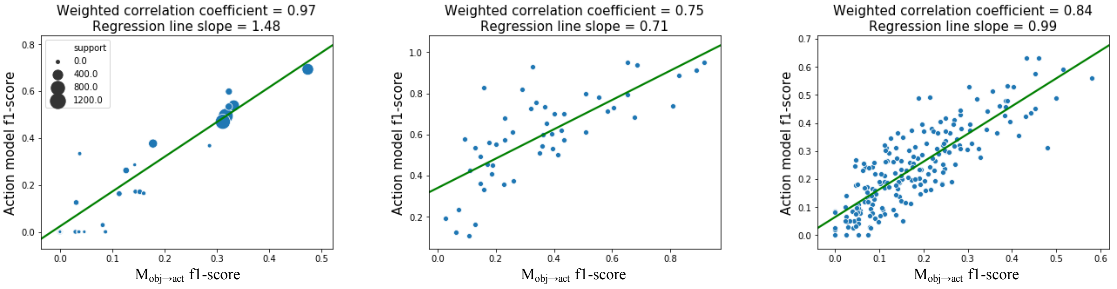
where K denotes the number of groups. Another reasonable alignment method choice is the slope of the least-square fit regression line:

$$A_{\text{slope}}(x, y) = \frac{\sum_{k=1}^{K}(x_k - \bar{x})(y_k - \bar{y})}{\sum_{k=1}^{K}(x_k - \bar{x})^2} \tag{3.3}$$

Finally, ORL is formulated as:

$$ORL(M_{\text{vid}\rightarrow\text{act}}) = A(P(D, M_{\text{vid}\rightarrow\text{act}}), P(D, M_{\text{obj}\rightarrow\text{act}})) \tag{3.4}$$

Fig. 3.1 visualizes the strong ORL of I3D in three different datasets. Here per-class f1 scores are used to compute the per-group performance P_k, and a linear regression classifier is chosen as the model $M_{\text{obj}\rightarrow\text{act}}$, which means it uses only the linear combination of extracted object features θ_{obj} to predict the action.



(a) EPIC-KITCHENS (b) HMDB51 split 1 (c) Moments in Time

Figure 3.1: The Object Reliance Level(ORL) of I3D on three different datasets. We can see the strong ORL for all the tested datasets, indicating that high ORL is a prevailing property across action datasets even for the state-of-the-art action model I3D. Note that because the EPIC-Kitchens dataset has imbalanced label distribution, we calculate the weighted version of both A_{corr} and A_{slope} and visualize the support number of each class using different point sizes.

3.2 Fallacious Object Reliance

It should be noted that it is not intrinsically wrong for action models to have a high object reliance level. If the captured object bias is universal over different datasets, we should consider the object as an essential part of that action. For instance, it is true that action ”playing piano” does associate with the object ”piano”.

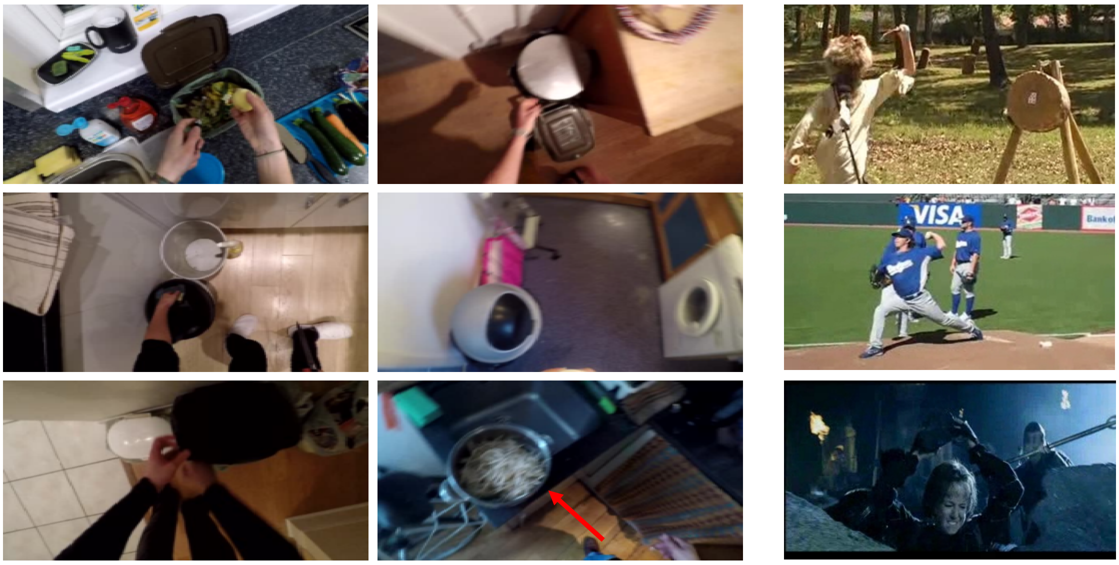
However, object biases in the training set D_{train} and testing set D_{test} are not guaranteed to be the same. In this situation, action models with heavy object reliance will learn the wrong object-action association and make inaccurate predictions in the testing set (see Fig. 3.2).

Therefore, we propose a new method to inspect the **discrepancy of object representation bias** between D_{train} and D_{test}. The idea is to calculate the performance drop of $M_{\text{obj}\rightarrow\text{act}}$ when trained on D_{train} and test on D_{test}:

$$\begin{aligned} B_{\text{diff}}(D_{\text{train}}, D_{\text{test}}, k) =& P_k(D_{\text{train}}, M_{\text{obj}\rightarrow\text{act}}(\theta_{\text{obj}}(D_{\text{train}}))- \\ & P_k(D_{\text{test}}, M_{\text{obj}\rightarrow\text{act}}(\theta_{\text{obj}}(D_{\text{test}})), \qquad (3.5) \\ \text{where } M_{\text{obj}\rightarrow\text{act}} =& \underset{M'}{\text{argmax}}\, P(D_{\text{train}}, M'(\theta_{\text{obj}}(D_{\text{train}})) \end{aligned}$$

In our experiments, we choose $M_{\text{obj}\rightarrow\text{act}}$ as a simple linear regression classifier and P_k as the per-class f1 scores, hence equation 3.5 can be interpreted as the object distribution

doi:10.6342/NTU202003008



(a) (b)

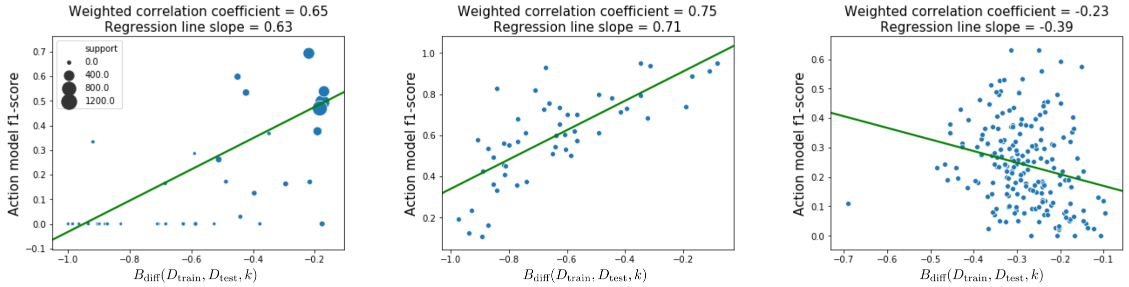
Figure 3.2: Examples of Fallacious Object Reliance with the action class ”throw”. (a) in the EPIC-KITCHENS dataset, ”throw” is often tied to ”trash can”, so the model relies on trash cans to predict the throwing action. However, in the testing set, ”throw” may be associated with other things like the highlighted pot. (b) in HMDB51, ”throw” is related to totally different objects from EPIC-KITCHENS. Action recognition models should capture the motion part of actions and avoid these FOR problems.

divergence of D_{train} and D_{text} given an action k.

If an action model performs worse on groups whose object bias discrepancy is large, it indicates the model is using the wrong object-action association to predict actions. Using this idea, we further propose our *Fallacious Object Reliance (FOR) measurement*, which is used to evaluate the alignment between the action model performance on D_{test} and the negative of object bias discrepancy.

$$FOR(M_{\text{vid}\longrightarrow\text{act}}) = A(P(D_{\text{test}}, M_{\text{vid}\longrightarrow\text{act}}), -B_{\text{diff}}(D_{\text{train}}, D_{\text{test}}, k)) \tag{3.6}$$

Fig. 3.3 shows the FOR scores of I3D on three different datasets. Note that because $M_{\text{obj}\longrightarrow\text{act}}$ has almost 100% accuracy on the D_{train} of HMDB split 1, $B_{\text{diff}}(D_{\text{train}}, D_{\text{test}}, k)$ is close to (1$-$ the performance on D_{test}), so Fig. 3.3 (b) is almost the same to Fig. 3.1 (b).



(a) EPIC-KITCHENS

(b) HMDB51 split 1

(c) Moments in Time

Figure 3.3: The Fallacious Object Reliance (FOR) scores of I3D on three different datasets. As we can see, although I3D has strong ORL on all the three datasets, it remains high FOR scores only on EPIC-KITCHENS and HMDB51. This indicates that either Moments in Time dataset is better calibrated so that it contains enough object diversity or it has similar object-action associations between its training and test sets.

doi:10.6342/NTU202003008

Chapter 4

Proposed Method

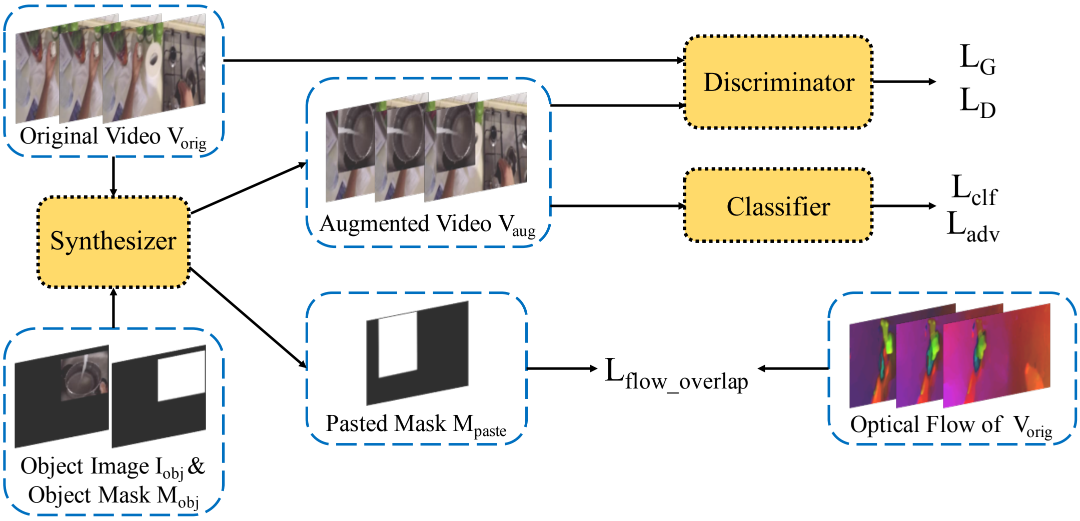


Figure 4.1: The overall architecture of AdvOST. AdvOST is composed of three different sub-networks: (a) a synthesizer S that affinely transforms the given object image and pastes it onto the original video to form an augmented video, (b) the classifier C to predict the action class given the augmented video at the training stage, and (c) the discriminator in charge of judging whether the input video is original or augmented and providing training signals for the synthesizer to produce natural synthesis. Additional regularization term called flow overlap loss is added to prevent the synthesizer paste on where motion occurs.

4.1 AdvOST

We show our overall training architecture called AdvOST in Fig. 4.1. This architecture consists of a synthesizer S, a classifier C, and a discriminator D.

For the synthesizer S, given an original video $v_{\text{orig}} \in V_{\text{orig}}$ and an object image $i \in I_{\text{obj}}$,

S will infer an affinement matrix to apply on the object image. The affined object image is pasted onto v_{orig} and produce a augmented video v_{aug}. Its network structure is illustrated and described in Fig. 4.2.

The classifier's target is to predict the action given v_{aug} in the training stage and v_{orig} in the testing stage. Its architecture can be any action recognition models, therefore the AdvOST is *model-agnostic*.

The final sub-network D acts just like the discriminator in the traditional structure of the generative adversarial network [12]. In each batch, v_{orig} and v_{aug} are fed to D, and it has to judge if the input video is authentic (v_{orig}) or synthesized (v_{aug}). Its goal is to prevent the synthesizer S from pasting objects in unnatural positions or at weird angles. Otherwise, it will be easy for the classifier to ignore the unnatural parts and make our purpose less effective.

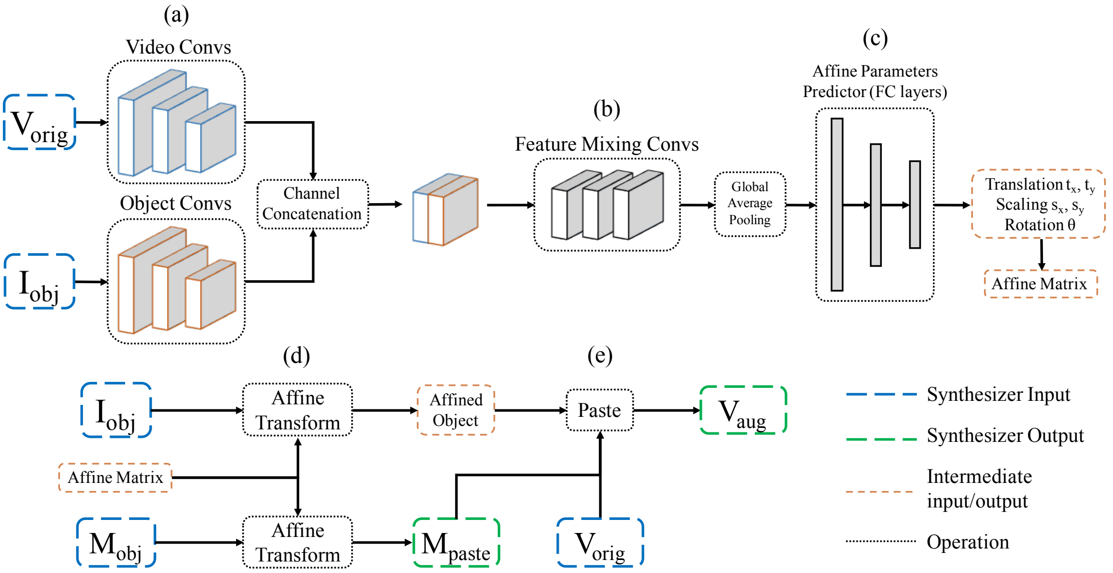


Figure 4.2: The network architecture of our synthesizer. (a) The video convs block and the object convs block extract the features of given videos and objects. The extracted features are concatenated in the channel dimension and processed by (b) the feature mixing convs block, where the output is then pooled by a global average pooling layer. The pooled feature is then fed into (c) the affinement parameters predictor to predict 5 parameters of the affinement matrix. The 5 degrees-of-freedom includes 2 translation, 2 scalings, and 1 rotation. We then apply the (d) affinement transformation on the object image I_{obj} and its mask M_{obj} and use them to (e) paste the affined object onto the original video V_{orig} and produce the augmented video V_{aug} and an affined object mask M_{paste}.

4.2 Loss Design

We'll discuss the losses we design one by one in this section.

Classification Loss. A cross-entropy loss is used as our classification loss to train our classifier C. Given an augmented video and the corresponding action label $(v, y) \in (\mathbf{V}_{aug}, \mathbf{Y})$ from k different action classes, the loss is:

$$L_{\text{clf}} = -\mathbb{E}_{(v,y)\sim(\mathbf{V}_{\text{aug}},\mathbf{Y})} \sum_{k=1}^{N} y_k \log C(v) \tag{4.1}$$

Adversarial Loss. To make the classifier more robust to unrelated objects, an adversarial loss is added during the optimization of the synthesizer. It's formulated as the negative cross-entropy loss:

$$L_{\text{adv}} = \mathbb{E}_{(v,y)\sim(\mathbf{V}_{\text{aug}},\mathbf{Y})} \sum_{k=1}^{N} y_k \log C(v) \tag{4.2}$$

Realness Losses. We use the original GAN loss designed by Goodfellow *et al.*[12] to implement our realness losses L^G_{real} and L^D_{real}. For the generator's objective, we apply the non-saturating version:

$$L^D_{\text{real}} = -\mathbb{E}_{v\sim V_{orig}}[\log D(v)] - \mathbb{E}_{v\sim V_{orig}}[1 - \log D(S(v)] \tag{4.3}$$

$$L^G_{\text{real}} = -\mathbb{E}_{v\sim V_{orig}}[\log D(S(v)] \tag{4.4}$$

Flow Overlap Penalty. Because of L_{adv}, the synthesizer may learn to paste on where the motion occurs when it's the most discriminative area, which is against our goal. Therefore, we come up with the Flow Overlap Penalty to penalize the overlapping area of the paste mask M_{mask} and the corresponding optical flow F_v of V_{orig}. Adding this penalty has an extra benefit on preventing a trivial solution for the synthesizer: enlarge the pasted object to occupy the whole video, because, in this situation, no unnatural sign can be found by the discriminator, and no clue can be used by the classifier to predict the action. The penalty is simply calculated by elementwisely multiply M_{paste} and F_v for T frames with height H and width W as follows.

$$L_{\text{flow\_overlap}} = \sum_{t}^{T} \sum_{x}^{W} \sum_{y}^{H} M_{\text{paste}}(x, y) \times F_v(t, x, y) \tag{4.5}$$

4.3 Optimization

The classifier, and the discriminator are trained using L_{clf}, L^{D}_{real}, respectively. The synthesizer is trained using $\lambda_a L_{\text{adv}} + \lambda_r L^{G}_{\text{real}} + \lambda_f L_{\text{flow\_overlap}}$, where λ_a, λ_r, and λ_f are the hyper-parameters during training.

doi:10.6342/NTU202003008

Chapter 5

Experiment

5.1 Datasets

Here we will introduce the action datasets on which we test AdvOST and the object source datasets we use for I_{obj}.

Action datasets. EPIC-KITCHENS [5] is the main action recognition testbed we use to test our analysis and AdvOST. This dataset consists of 39,594 segments in 432 videos, where 125 daily kitchen activities labels like cooking, mixing, and cutting are provided. This dataset is collected in the 31 different participants' kitchens, therefore the environment difference among each kitchen must be considered. We manually split part of the released training data into two validation sets, **seen** and **unseen**. The seen validation set has 12% randomly sampled segments of videos in kitchens that have appeared in the training set, while the unseen validation set consists of videos only in kitchens of participant 05, 06, and 07, where 9.19% videos and 7.43% segments are included. The unseen set is more difficult since the object and action distribution is very different from the training set.

Besides, we also test on the benchmarks HMDB51 [22] and UCF101 [32]. HMDB51 includes 6.8K videos of 51 actions, while UCF101 is composed of around 13K videos of 101 actions. Both dataset organizers provided 3 splits, and each split has its own training and testing data.

doi:10.6342/NTU202003008

Object source datasets. our method makes use of object images to augment the training videos with our synthesizer. Since EPIC-KITCHENS dataset has another detection track for common kitchen objects, we use the bounding box annotation to crop object images, which are used as I_{obj} when experimenting on EPIC-KITCHENS's action track. When testing on other action datasets, we use the objects cropped from COCO detection dataset [25].

5.2 Experiment Details

The synthesizer loss weights λ_a, λ_r, and λ_f are all set to 1. The only exception is when using TSN as our backbone, λ_f is empirically set to 0.1.

The I3D network we use is based on the InceptionV1 backbone[2] and pretrained on Kinetics-400. For SlowFast, we choose 8×8, R50 and use its official pretrained weights. The TSN network we use is with the BNInception backbone [18] pretrained on ImageNet. The sampled frame numbers for I3D, TSN, and SlowFast are 16, 16, and 32, respectively.

When training, we first resize the input video such that the shorter edge becomes 256 pixels length. Then, we randomly crop the videos into 256x256 and resize it to 224x224. No other data augmentation technique is used. At testing, to make the comparison simple, we do not include any test time augmentation.

We use the Adam optimizer with betas as 0.5 and 0.999 for all the backbones and our sub-networks. The learning rate is set to 0.0001 and decay by 0.1 for every 5 epochs. when training the baselines. When training AdvOST learning rates for the synthesizer, classifier, and discriminator are set to 0.00015, 0.0003, and 0.00015, respectively. The final scores we report for all experiments use the epoch that acquires the highest mean of top 1 and top 5 accuracy of the validation set.

5.3 Results

EPIC-KITCHENS. Table 5.1 show the results of our AdvOST method and the Fallacious Object Reliance metrics (FOR) on the three different backbones, i.e. I3D, TSN, and

doi:10.6342/NTU202003008

Table 5.1: Performance of seen validation set of EPIC-KITCHENS dataset. After applying AdvOST, the baselines constantly get higher performance and have lower Fallacious Object Reliance scores in the most case.

Split	Method	Top1 Acc.↑	Top5 Acc.↑	FOR_{corr} ↓	$\text{FOR}_{\text{slope}}$ ↓
Seen	I3D	47.59%	79.70%	0.5346	0.8701
	I3D+AdvOST	**49.21%**	**80.58%**	**0.5145**	**0.8095**
	TSN	40.68%	**79.64%**	0.4865	0.7112
	TSN+AdvOST	**41.02%**	**79.64%**	**0.4610**	**0.6735**
	SlowFast	**56.48%**	82.53%	0.5984	**0.9548**
	SlowFast+AdvOST	56.05%	**82.56%**	**0.5879**	1.024
unseen	I3D	43.05%	**74.24%**	0.8231	1.122
	I3D+AdvOST	**43.43%**	74.14%	**0.8230**	**1.114**
	TSN	33.36%	71.59%	**0.7383**	**0.9422**
	TSN+AdvOST	**34.31%**	**72.21%**	0.7674	0.9490
	SlowFast	49.57%	77.83%	0.8747	**1.193**
	SlowFast+AdvOST	**51.60%**	**78.02%**	**0.8632**	1.297

SlowFast. For the seen and unseen validation sets of EPIC-KITCHENS, all backbones have intermediate or strong FOR scores, indicating that even the recent state-of-the-art action models have the FOR issue. Also, the FOR scores in the unseen validation set is much higher than the ones in the seen validation set. It's because the unseen validation set contains much different object-action joint distribution, which demonstrates the FOR metrics we propose can exploit and reflect the issue we found.

After applying our AdvOST architecture, almost all the three backbones have Top1 accuracy improved compared to the ones without AdvOST, showing our proposed training procedure does enhance the robustness of backbones.

For the FOR scores, we found that in the seen dataset, most of the backbones have decreased FOR scores after using AdvOST. However, in the unseen dataset, only the two SOTA models I3D and SlowFast have reduced FOR scores, while the TSN ones increase. This result may imply that the TSN model cannot achieve high accuracy as other models in the unseen dataset because it does not learn enough valid object association.

Moment in Time. As the Moment in Time dataset has a better variety of objects in the training data, the FOR issue is less severe. Still, after applying our AdvOST, we still get

Table 5.2: Performance of Moments in Time dataset.

Method	Top1 Acc.↑	Top5 Acc.↑	FOR_{corr} ↓	$\text{FOR}_{\text{slope}}$ ↓
I3D	25.60%	51.38%	-0.2410	-0.4197
I3D+AdvOST	**27.05%**	**53.73%**	-0.2262	-0.3968

Table 5.3: Performance for the HMDB51 dataset. Our method could improve both the action recognition accuracy and FOR scores in most cases.

Split	Method	Top1 Acc. ↑	Top5 Acc. ↑	FOR_{corr} ↓	$\text{FOR}_{\text{slope}}$ ↓
1	I3D	59.80%	**88.69%**	0.7392	0.6797
	I3D+AdvOST	**60.26%**	88.23%	**0.7239**	**0.6796**
2	I3D	60.84%	87.40%	0.8221	0.7698
	I3D+AdvOST	**61.11%**	**87.64%**	**0.7675**	**0.6868**
3	I3D	60.91%	87.45%	0.8236	**0.8457**
	I3D+AdvOST	**61.50%**	**88.16%**	**0.7845**	0.8626

accuracy improvement as AdvOST forces the model to learn more about the motion itself while staying low for the object dependency.

HMDB51. We also test AdvOST on a backbone I3D using HMDB51. Table 5.3 shows that for the three testing sets, AdvOST consistently improves the performance in terms of the Top1 accuracy and mitigates the FOR issue.

Table 5.4: Ablation study on the EPIC-KITCHENs dataset with I3D as backbone.

Synthesizer	Discriminator	$L_{\text{flow\_overlap}}$	Top1 Acc.↑	Top5 Acc.↑	FOR_{corr} ↓	$\text{FOR}_{\text{slope}}$ ↓
			47.59%	79.70%	0.5346	0.8701
✓	✓		48.27%	79.67%	**0.4823**	0.8386
✓		✓	47.35%	80.01%	0.5097	0.8619
✓	✓	✓	**49.21%**	**80.58%**	0.5145	**0.8095**

5.4 Ablation Study

To validate each component of AdbOST, we conducted an ablation study on the EPIC-KITCHENs dataset with I3D as a backbone. From Table 5.4 we could observe that all the three components are necessary for AdbOST. Without the discriminator, the synthesizer will generate unreasonable augmented videos so that the performance has a significant drop. Also, without the flow overlap loss, it is trivial for the synthesizer to fool the model by blocking the motion part.

Chapter 6

Discussion

6.1 Per-class Improvement and Confusion Matrix

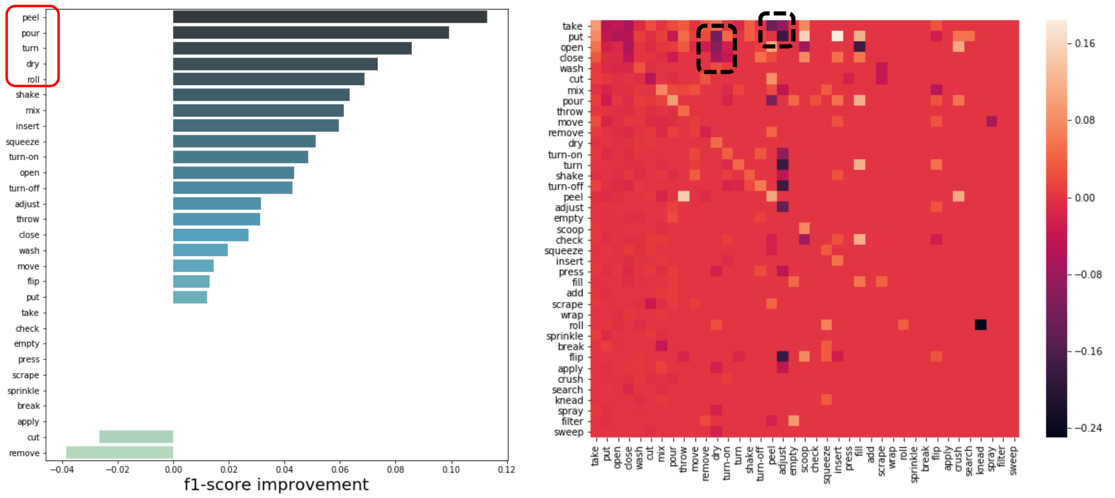
In order to dig deeper into what AdvOST contributes, we visualize the per-class score improvement and the difference of confusion matrix of model I3D after applying AdvOST in Fig. 6.1. The per-class improvement figure in (a) demonstrated the power of AdvOST, especially for those actions that often occur in particular locations or with specific objects such as ”peel”, ”pour”, ”dry”, and ”roll”, because for these classes, the original I3D may spot the surrounding objects repeatedly and learn to make use of these hints.

The confusion matrix difference before and after applying AdvOST can be seen in 6.1 (b). This figure reveals more details hidden in (a). For example, in the left black dashed box, we understand the improvement f1 score of action ”dry” is due to the decreased misclassification to ”put”, ”open”, and ”close”, the actions that can take place in more general scenes. The right black dashed box exposes similar information that after applying AdvOST, the fallacious association of objects and actions is alleviated. Note that we merge the seen and unseen validation sets in the two figures and have filtered out those classes with less than 20 examples in the validation set for clearer visualization.

doi:10.6342/NTU202003008

6.2 Grad-CAM Comparison

We show several grad-CAM visualizations in Fig 6.2 to compare the interior behaviors of pure I3D and I3D with AdvOST and we could see the effectiveness of AdvOST that guides action models to put more awareness on motion.

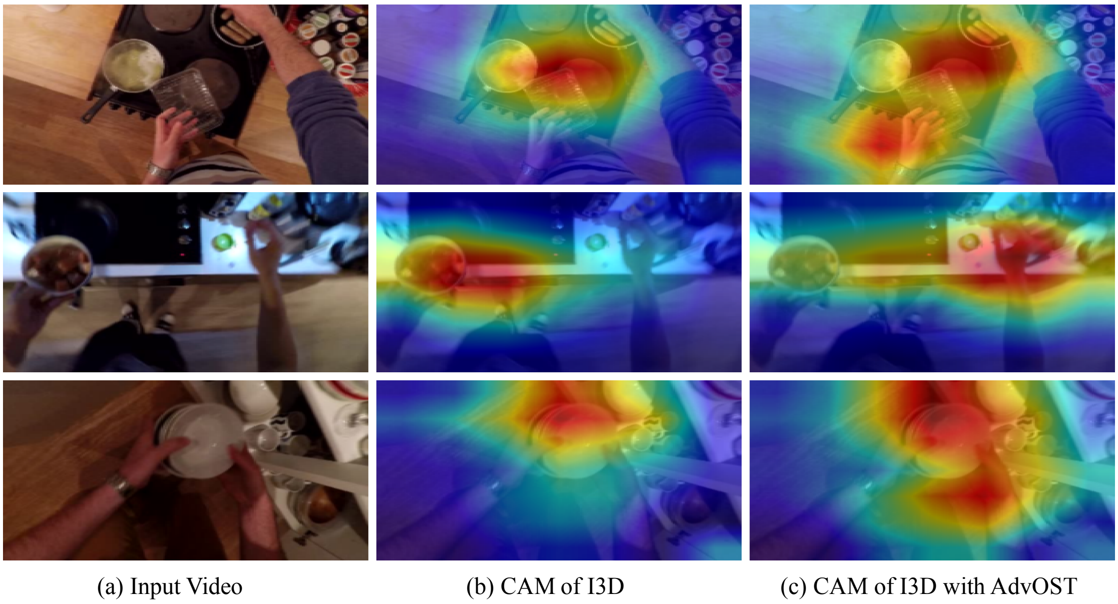


(a) Per-class Improvement

(b) Comfusion Matrix Difference

Figure 6.1: (a) The per-class f1 improvement and (b) the changes of the confusion matrix after applying AdvOST on I3D with EPIC-KITCHENS validation sets. (a) shows AdvOST helps our classifier improves most classes, especially for those actions subject to certain places or particular objects. The changes in the confusion matrix after applying AdvOST (b) demonstrate where the improvement comes from in detail. Please refer session 6.1 for in-depth discussion.

國立臺灣大學 · 人學國愛 · 學勵品敦



(a) Input Video (b) CAM of I3D (c) CAM of I3D with AdvOST

Figure 6.2: Grad-CAM visualizations of pure I3D and I3D with AdvOST. Compared to pure I3D, I3D with AdvOST focuses more on the hands performing that action instead of the subjects of the action. Besides, as we can see in the first two rows, I3D with AdvOST can pay attention on both hands if they are present, while pure I3D can not.

doi:10.6342/NTU202003008

Chapter 7

Conclusion

In this paper, we propose the Object Reliance Level and Fallacious Object Reliance (FOR) to measure action recognition models' erroneous dependency on objects in videos, based on our observations on the dataset's object bias and CNN model's invalid object-dependent behavior. Furthermore, we propose a novel model-agnostic approach, Adversarial Object Synthesis Training (AdvOST), to reduce the models' FOR score by increasing the object diversity of the training dataset with an object synthesizer. Experiments on the EPIC-KITCHEN and HMDB51 datasets suggest that our method could effectively improve the accuracy of SOTA action recognition models including TSN, I3D, and SlowFast.

doi:10.6342/NTU202003008

Bibliography

[1] J. A. Buolamwini. *Gender shades: intersectional phenotypic and demographic evaluation of face datasets and gender classifiers*. PhD thesis, Massachusetts Institute of Technology, 2017.

[2] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[3] C.-H. Chen, A. Tyagi, A. Agrawal, D. Drover, R. MV, S. Stojanov, and J. M. Rehg. Unsupervised 3d pose estimation with geometric self-supervision. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[4] J. Choi, C. Gao, J. C. Messou, and J.-B. Huang. Why can't i dance in the mall? learning to mitigate scene bias in action recognition. In *Advances in Neural Information Processing Systems*, pages 851–863, 2019.

[5] D. Damen, H. Doughty, G. Maria Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 720–736, 2018.

[6] D. Drover, R. MV, C.-H. Chen, A. Agrawal, A. Tyagi, and C. Phuoc Huynh. Can 3d pose be learned from 2d projections alone? In *The European Conference on Computer Vision (ECCV) Workshops*, September 2018.

doi:10.6342/NTU202003008

[7] N. Dvornik, J. Mairal, and C. Schmid. Modeling visual context is key to augmenting object detection datasets. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 364–380, 2018.

[8] D. Dwibedi, I. Misra, and M. Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1301–1310, 2017.

[9] C. Feichtenhofer, H. Fan, J. Malik, and K. He. Slowfast networks for video recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6202–6211, 2019.

[10] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1933–1941, 2016.

[11] G. Georgakis, A. Mousavian, A. C. Berg, and J. Kosecka. Synthesizing training data for object detection in indoor scenes. *arXiv preprint arXiv:1702.07836*, 2017.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[13] R. Goyal, S. E. Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *ICCV*, volume 1, page 5, 2017.

[14] C. Gu, C. Sun, D. A. Ross, C. Vondrick, C. Pantofaru, Y. Li, S. Vijayanarasimhan, G. Toderici, S. Ricco, R. Sukthankar, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6047–6056, 2018.

doi:10.6342/NTU202003008

[15] L. A. Hendricks, K. Burns, K. Saenko, T. Darrell, and A. Rohrbach. Women also snowboard: Overcoming bias in captioning models. In *European Conference on Computer Vision*, pages 793–811. Springer, 2018.

[16] N. Hussein, E. Gavves, and A. W. Smeulders. Timeception for complex action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 254–263, 2019.

[17] H. Idrees, A. R. Zamir, Y.-G. Jiang, A. Gorban, I. Laptev, R. Sukthankar, and M. Shah. The thumos challenge on action recognition for videos “ in the wild ” . *Computer Vision and Image Understanding*, 155:1–23, 2017.

[18] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

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

[20] A. Khoreva, R. Benenson, E. Ilg, T. Brox, and B. Schiele. Lucid data dreaming for object tracking. In *The DAVIS Challenge on Video Object Segmentation*, 2017.

[21] R. Krishna, K. Hata, F. Ren, L. Fei-Fei, and J. Carlos Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 706–715, 2017.

[22] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: a large video database for human motion recognition. In *2011 International Conference on Computer Vision*, pages 2556–2563. IEEE, 2011.

[23] Y. Li, Y. Li, and N. Vasconcelos. Resound: Towards action recognition without representation bias. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 513–528, 2018.

doi:10.6342/NTU202003008

[24] C.-H. Lin, E. Yumer, O. Wang, E. Shechtman, and S. Lucey. St-gan: Spatial transformer generative adversarial networks for image compositing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9455–9464, 2018.

[25] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[26] Z. Qiu, T. Yao, and T. Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *proceedings of the IEEE International Conference on Computer Vision*, pages 5533–5541, 2017.

[27] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[28] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, pages 510–526. Springer, 2016.

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

[30] Y. Song, R. Shu, N. Kushman, and S. Ermon. Constructing unrestricted adversarial examples with generative models. In *Advances in Neural Information Processing Systems*, pages 8312–8323, 2018.

[31] K. Soomro and A. R. Zamir. Action recognition in realistic sports videos. In *Computer vision in sports*, pages 181–208. Springer, 2014.

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

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

[34] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[35] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018.

[36] S. Tripathi, S. Chandra, A. Agrawal, A. Tyagi, J. M. Rehg, and V. Chari. Learning to generate synthetic data via compositing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 461–470, 2019.

[37] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016.

[38] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[39] X. Wang, A. Shrivastava, and A. Gupta. A-fast-rcnn: Hard positive generation via adversary for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2606–2615, 2017.

[40] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321, 2018.

[41] B. Zhou, A. Andonian, A. Oliva, and A. Torralba. Temporal relational reasoning in videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 803–818, 2018.

[42] L. Zhou, C. Xu, and J. J. Corso. Towards automatic learning of procedures from web instructional videos. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

doi:10.6342/NTU202003008