Removing the Background by Adding the Background: Towards Background Robust Self-supervised Video Representation Learning

Jinpeng Wang¹, ²*, Yuting Gao²*, Ke Li², Yiqi Lin¹, Andy J. Ma¹†, Xing Sun²
¹ Sun Yat-sen University ² Tencent Youtu Lab

Abstract

Self-supervised learning has shown great potentials in improving the video representation ability of deep neural networks by constructing surrogate supervision signals from the unlabeled data. However, some of the current methods tend to suffer from a background cheating problem, i.e., the prediction is highly dependent on the video background instead of the motion, making the model vulnerable to background changes. To alleviate the problem, we propose to remove the background impact by adding the background. That is, given a video, we randomly select a static frame and add it to every other frames to construct a distracting video sample. Then we force the model to pull the feature of the distracting video and the feature of the original video closer, so that the model is explicitly restricted to resist the background influence, focusing more on the motion changes. In addition, in order to prevent the static frame from disturbing the motion area too much, we restrict the feature being consistent with the temporally flipped feature of the reversed video, forcing the model to concentrate more on the motion. We term our method as Temporal-sensitive Background Erasing (TBE). Experiments on UCF101 and HMDB51 show that TBE brings about 6.4% and 4.8% improvements over the state-of-the-art method on the HMDB51 and UCF101 datasets respectively. And it is worth noting that the implementation of our method is so simple and neat and can be added as an additional regularization term to most of the SOTA methods without much efforts.

Introduction

Convolutional neural networks (CNNs) have achieved competitive accuracy on a variety of video understanding tasks, including action recognition (Hara, Kataoka, and Satoh 2018), temporal action detection (Zhao et al. 2017) and spatio-temporal action localization (Weinzaepfel, Harchaoui, and Schmid 2015). Such success relies heavily on manually annotated datasets, which are time-consuming and expensive to obtain. Meanwhile, there are numerous unlabeled data that are instantly available on the Internet, drawing more and more researchers’ attention from the community to utilize off-the-shelf unlabeled data to improve the performance of CNNs by self-supervised learning.

One successful way of self-supervised learning is to resolve the designed pretext task on the unlabeled data. However, Li et al.(Li, Li, and Vasconcelos 2018a) and Girdhar et al.(Girdhar and Ramanan 2020) point out that the current commonly used video datasets usually exist large implicit biases over scene and object structure, making the temporal structure become less important and the prediction tends to have a high dependence on the video background. We name this phenomenon as background cheating problem, as is shown in Figure 1. For example, a trained model may classify an action as playing soccer simply because it sees the field, without really understanding the motion. As a result, the model is easily to overfit the training set, and the learned feature representation is likely to be scene-biased. Li et al.(Li, Li, and Vasconcelos 2018a) reduces the bias by resampling the training set, and Wang et al.(Wang and Hoai 2018) propose to pull actions out of context by training a binary classifier to explicitly distinguish action samples and conjugate samples that are contextually similar to human action samples but do not contain the action.

In this work, in order to tackle the background cheating problem and make the model generalize better, we present to reduce the impact of the background by adding the background with feature consistency regularization, which is termed as Temporal-sensitive Background Erasing (TBE). Specifically, given a video, we randomly select a static frame...
and add it to every other frames to construct a distracting video, as is shown in Figure 3. Then we force the model to pull the feature of the distracting video and the feature of the original video together with consistency regularization. In addition, since the added static frame may cause interference to the area of the moving subject, we take advantage of the symmetry of time to further constrain the motion pattern. Concretely, we reverse the frame order of the input video and restrict the extracted features to be symmetric in the temporal dimension, which enhance the temporal discrimination ability of the extracted representation.

Experimental results demonstrate that the proposed method can effectively reduce the influence of the background noise, and the extracted representation is more robust to the background bias and have stronger generalization ability. At the same time, the information of motion pattern is further highlighted. Our approach is simple but incorporate it as a regularization term into existing self-supervised video learning methods can bring significant gains.

In summary, our main contributions are twofold:

- We propose a simple but effective video representation learning method that is robust to the background.
- The proposed approach can be easily incorporated with existing self-supervised video representation learning methods as a regularization term, bringing further gains on UCF101(Soomro, Zamir, and Shah 2012) and HMDB51(Kuehne et al. 2013) datasets.

### Related Work

#### Self-supervised Video Representation Learning

Self-supervised learning is a generic learning framework that only relies on the pretext task of unlabeled data. The alternative signal in pretext task exploits labeling that comes for free. Representative alternative signals include predicting the rotation angle of image (Gidaris, Singh, and Komodakis 2018) and solving the jigsaw puzzle (Noroozi and Favaro 2016). Recent years, self-supervised learning has expanded into the video domain and has attracted a lot interests.

The majority of the prior work explored natural video properties as supervisory signal. Among them, temporal order is one of the widely-used property. For example, Wei et al.(Wei et al. 2018) classify the arrow of time, Misra et al.(Misra, Zitnick, and Hebert 2016) distinguish a real or-

### Methodology

In this section we introduce the proposed Temporal-sensitive Background Erasing (TBE) for action recognition. We first give an overall description, and then introduce the TBE in details.

### Overall Architecture

The framework of the proposed TBE is shown in Figure 2. For each input video, we first randomly crop a fixed-length clip from different spatial locations, denoted as $x_n$ and $x_t$. In this way, the input clips have different distribution in the pixel level but are consistent in the semantic level. Afterwards, $x_n$ is directly fed into the 3D backbone to extract the feature representation and we denote this procedure as $F(x_n; \theta)$, where $\theta$ represents the backbone parameters. For $x_t$, we first generate a distracting counterpart $x_d$ for it, which has the interference of added static frame noise but the semantics remains the same. Then we apply a temporal inversion operation to $x_d$ to obtain the final input. The output feature maps of $x_n$ and $x_t$ are represented by $f_{x_n}, f_{x_t} \in \mathbb{R}^{C \times T \times H \times W}$. $C$ is the number of channel and $T$ is the length of time dimension. $W$ and $H$ are spatial size. At last, the extracted features $f_{x_n}, f_{x_t}$ are fed into a regularization head and a pretext task head to guide the learning procedure. Specifically, in the regularization head, we use Temporal Consistency Regularization to minimize the distance between $f_{x_n}, f_{x_t}$. The Algorithm 1 summarizes the overall procedure of TBE.
Algorithm 1 Training with Temporal-sensitive Background Erasing.

Require: $F_0$: convolutional neural network with parameter $\theta$, DVG: distracting video generation method, TR: temporal reverse, TCR: temporal consistency regularization.

INPUT: unlabeled video $x$

OUTPUT: the parameter $\theta$

1: for each iteration do
2:  $x_o \leftarrow \text{random \_crop}(x)$
3:  $x_t \leftarrow \text{random \_crop}(x)$
4:  $x_d \leftarrow \text{DVG}(x_t)$
5:  $x_t \leftarrow \text{TR}(x_d)$
6:  Feature Representation:
7:  $f_{x_o} \leftarrow F_0(x_o)$ \> feature of the original video
8:  $f_{x_t} \leftarrow F_0(x_t)$ \> feature of the transformed video
9:  $f_{x_t} \leftarrow \text{TR}(f_{x_t})$ \> reverse the learned feature of the transformed video
10: Update Network:
11:  $\text{loss} \leftarrow \text{TCR}(f_{x_o}, f_{x_t})$ \> self-supervised loss
12:  update $\theta$ using SGD \> update network parameters
13: end for

Figure 2: The framework of the proposed method TBE. A video is first randomly cropped spatially, then we generate the distracting video by adding a static frame upon other frames followed by the temporal inversion. The model is trained by a pretext task together with the proposed temporal consistency regularization, with the goal of pulling the feature of the original video and the temporally flipped feature of the reversed distracting video closer. (Best viewed in color).

Background Erasing. In the video representation learning, sometimes the statistical characteristics of the background will drown out the motion features of the moving subject. Thus it is easy for the model to make predictions based only on the background information. Moreover, when solving self-supervised pretexts, the model sometimes judges by some tricky means, for example, when solving the rotation pretext, it may resort to the black border of the video (Gidaris, Singh, and Komodakis 2018; Kim, Cho, and Kweon 2019), neglecting the content changes. Therefore, the model is easy to overfit to the training set and has poor generalization on the new dataset. Such solutions are termed degenerated solution.

Background Erasing (BE) is proposed to remove the negative impact of the background by adding the background. Specifically, we randomly select one static frame $x^t_k$ from $x_t$ and add it as a spatial background noise to every other frames to generate a distracting video $x_d$ through:

$$x^j_d = (1 - \lambda) \cdot x^j_t + \lambda \cdot x^k_t, j \in [1, T]$$

where $\lambda$ is sampled from the uniform distribution $[0, \gamma]$, $j$ is the index of frame and $k$ denotes the index of the random selected frame. Compared to $x_t$, $x_d$ add background perturbation on the spatial dimension, but the motion pattern is basically not changed, as shown in Figure 3. Afterwards, we force the model to pull the feature of $x_o$ and $x_d$ closer under the temporal consistency regularization, which will be introduced in details later. Since $x_o$ and $x_d$ resemble each other in the motion pattern but differentiate each other in spatial, when the features of $x_o$ and $x_d$ are brought closer, the model will be promoted to suppress the background noise, yielding video representations that are more sensitive to motion changes. We have tried a variety of ways to add background noise, which are shown in the ablation study part of the experiment. Experimental results demonstrate that the intra-video static frame works best overall. However, simply approximate a static frame as background noise is not enough, since it may also affect the motion area. In order to overcome this side effect, we propose a temporal inversion operation to compensate the loss, which will be introduced in detail in the next section.

Temporal Inversion. Compared with the image representation learning, the most important factor in the video representation learning is the object motion, especially for action recognition task. In order to compensate the the negative effect of the background erasing on the motion patter, we further propose Temporal Inversion (TI) to constrain the motion pattern based on the symmetry of time. That is, restrng the feature being consistent with the temporally flipped feature of the reversed video.

Specifically, we first reverse the frame order of $x_d$ obtained from background erasing, then we flip the feature of reversed video in the temporal dimension. The entire procedure can be expressed by the formula:

$$f_{x_t} = \text{Flip}(F(\text{Reverse}(x_d, \theta))),$$

where $F$ denotes the feature extraction, and both Reverse and Flip operations are done in the temporal dimension.
The essence behind this operation is that, in a video, when a certain clip contains strong motion, its corresponding spatio-temporal representation will also have a higher response. After the video is inverted, although these clips are also reversed, strong motions are retained, so the corresponding features should also have high responses.

**Temporal Consistency Regularization**

In this section, we use a temporal consistency regularization term to pull the feature of $x_o$ closer to the feature of $x_d$, and make them consistent in the temporal dimension. Formally,

$$
\mathcal{L}_{tcr} = ||\psi(f_{x_o}) - \psi(f_{x_d})||^2
$$

(3)

where $\psi$ is an explicit feature mapping function that project features from $C \times T \times H \times W$ to $C \times T$. We use spatial global max pooling since $x_o$ and $x_d$ have different pixel distribution due to random cropping. In this fashion, we force the max response at each time dimension being consistent.

**Pretext Tasks**

TBE is supposed to be used as a regularization term. Using it solely for optimization will make the model fall into a trivial solution easily. Therefore, we integrate our method into the existing pretext tasks which require high-level discriminative representations.

Most pretext tasks can be formulated as a multi-category classification task and optimized with the cross-entropy loss. Specifically, each pretext will define a transformation set $R$ with $M$ operations. Given an input $x$, a transformation $r \in R$ is performed, then the convolutional neural network with parameters $\theta$ is required to distinguish which operation it is. The entire loss function is as follows,

$$
\mathcal{L}_p = -\frac{1}{M} \sum_{r \in R} \mathcal{L}(F(r(x); \theta), r),
$$

(4)

where $\mathcal{L}$ is Cross Entropy. Take Rotation Prediction (Gidaris, Singh, and Komodakis 2018) task as an example, a model is required to determine the rotation angle of an image. In particular, given a video, we randomly choose rotation degrees from set $R = \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$ and rotate all frames with the same degree. For other pretext tasks, such as the commonly used ST Puzzle (Kim, Cho, and Kweon 2019) and Clip Order (Xu et al. 2019), the only difference from the Rotation Prediction task is the definition of the set $S$.

**Temporal-sensitive Background Erasing**

By integrating our method into a pretext task, the final objective function becomes:

$$
\mathcal{L}_{tbe} = \mathcal{L}_p + \beta \mathcal{L}_{tcr},
$$

(5)

where $\beta$ is a hyperparameter that controls the importance of the regularization term. In our experiments, $\beta$ is set to 1.

In order to verify the generalization ability of our method, in this work, we explore three widely used pretext tasks, that is, Rotation Prediction, ST Puzzle and Clip Order.

**Experiments**

**Implementation Details**

In this section, we use two well-known video datasets, UCF101 and HMDB51. UCF101 is an action recognition dataset of realistic action videos, collected from YouTube with 13,320 videos of 101 action categories. HMDB51 is collected from various sources and contains 6,849 clips of 51 action categories.

We use PyTorch (Paszke et al. 2017) to implement the whole framework. In order to demonstrate the generality of our work, we use C3D (Tran et al. 2015), R3D (Hara, Kataoka, and Satoh 2018), R(2+1)D (Tran et al. 2018) and I3D (Carreira and Zisserman 2017) as backbones. For each model, the consistency regularization is performed before the global average pooling layer. We provide complete implementation details of each network in the Supplementary material. Final results are obtained through the following two steps:

**Step 1: Self-supervised pre-training.** We pre-train the network using TBE on the split 1 of the UCF101 dataset. The input clip consists of 16 frames and the temporal stride is 4 so that the adjacent frames have great visual difference. The choice of the temporal stride is analysed in the Supplementary material. In particular, given a video, we randomly choose rotation degrees from set $R = \{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$ and rotate all frames with the same degree. For other pretext tasks, such as the commonly used ST Puzzle (Kim, Cho, and Kweon 2019) and Clip Order (Xu et al. 2019), the only difference from the Rotation Prediction task is the definition of the set $S$.

**Step 2: Supervised fine-tuning.** After the pre-training stage, we transfer the learned parameters to the downstream task, i.e., action recognition, with the last fully connected layer randomly initialized. During the fine-tuning and testing stage, we follow the same protocol in (Xu et al. 2019) to
Table 1: Adding TBE as a regularization term to the previous pretext-based self-supervised video representation learning methods. We report the top-1 accuracy (%) on the UCF101 and HMDB51. TBE can bring significant improvement over three mainstream pretext tasks. Supervised methods are also listed for reference on the top of the table.

Table 2: The results of different distracting video generation methods on HMDB51 dataset with C3D backbone. Compared to the baseline, Intra-Video Mixup can bring a 2.6% improvement.

Ablation Analysis
In this section, we conduct experiments to explore the effectiveness of different distracting video generation methods, different component of TBE and different backbones. For simplicity, we employ Rotation Prediction pretext with C3D backbone as the baseline, and all the experiments are pre-trained on the split 1 of the UCF101 dataset.

Variants of Distracting Video Generation. One main operation in the background erasing is to generate a distracting video which has background noise, but the temporal semantics is retained. In order to explore whether adding a static frame is the most effective operation, we compare it with another four common ways: (a) Gaussian Noise: add an identical White Gaussian Noise on each frame. (b) Video Mixup (Hongyi Zhang and Lopez-Paz 2018): interpolate two videos frame by frame. (c) Video CutMix (Yun et al. 2019): randomly replace one region of each frame with a patch from another frame. (d) Inter-Video Mixup: randomly select one frame from another video, and add this static frame as noise to each frame of this video. (e) Our Intra-Video Mixup: randomly select one frame from the video itself, and add this static frame as noise to each frame of this video. The results are shown in Table 2 and three observations can be obtained:

i. Video Mixup and Video CutMix perform worse than the baseline. Notice that these two ways destroy the motion pattern of the original video, which demonstrates the importance of keeping semantics consistency.

ii. Gaussian Noise, Inter-Video Mixup and Intra-Video Mixup give positive improvement and are more suitable for action modeling since all of them preserve the motion semantics. Therefore, the idea of removing noise by adding noise is effective, but it is essential to make sure the intro-

Comparison with State-of-the-arts
In this section, we integrate TBE into several pretext tasks to verify the performance gains brought by TBE. Specifically, we conduct experiments on three pretext tasks, Rotation Prediction, ST Puzzles and Clip Order. In each video, 10 clips are uniformly selected for prediction, and the final result of the video is the average of 10 clip results. All the results shown in Table 1 are averaged over 3 dataset splits. We also report the result of the random initialized model and the result of the model pre-trained with all labels of ImageNet and Kinetics in a supervised manner for reference. It can be observed that TBE can bring prominent gains on three pretext tasks under the same setting. With the Clip Order pretext, TBE brings 4.8% improvement on the UCF101 dataset, and 6.4% improvement on the HMDB51 dataset.

provide a fair comparison. We fine-tuned the network for 45 epochs and the optimizer is set the same as the pre-training stage. The learning rate is initialized as 0.05 and decreases to 1/10 every 10 epochs.

| Method                           | Backbone | Pretrained   | UCF101(%) | HMDB51(%) |
|----------------------------------|----------|--------------|-----------|-----------|
| **Supervised**                   |          |              |           |           |
| Random Init                      | C3D      | -            | 60.5      | 21.2      |
| ImageNet Supervised              | C3D      | ImageNet     | 67.1      | 28.5      |
| Kinetics Supervised              | C3D      | Kinetics     | 96.8      | 74.5      |

| Method                           | Backbone | Pretrained   | UCF101(%) | HMDB51(%) |
|----------------------------------|----------|--------------|-----------|-----------|
| **Self-supervised**              |          |              |           |           |
| Shuffle & Learn (Misra, Zitnick, and Hebert 2016) [ECCV, 2016] | AlexNet | UCF101 | 50.2 | 18.1 |
| VGAN (Vondrick, Pirsiavash, and Torralba 2016) [NeuIPS, 2016] | VGN | UCF101 | 52.1 | - |
| OPN (Lee et al. 2017) [ICCV, 2017] | Caffe Net | UCF101 | 56.3 | 22.1 |
| Geometry (Gan et al. 2018) [CVPR, 2018] | Flow Net | UCF101 | 55.1 | 23.3 |
| Rotation Prediction (Gidaris, Singh, and Komodakis 2018) | C3D | UCF101 | 62.5 | 25.6 |
| Rotation Prediction + TBE | C3D | UCF101 | 66.4 (3.9↑) | 29.2 (3.6↑) |
| Rotation Prediction + TBE | C3D | Kinetics | 67.2 (4.7↑) | 31.4 (5.8↑) |
| ST Puzzles (Kim, Choi, and Rweon 2019) [AAAI, 2019] | C3D | UCF101 | 60.6 | 28.3 |
| ST Puzzles + TBE | C3D | UCF101 | 65.0 (4.4↑) | 31.7 (3.4↑) |
| Clip Order (Xu et al. 2019) [CVPR, 2019] | C3D | UCF101 | 65.6 | 28.4 |
| Clip Order + TBE | C3D | UCF101 | 70.4 (4.8↑) | 34.8 (6.4↑) |
In this part, we explore the effectiveness of BE and TI.

Effectiveness of BE and TI. In this part, we explore the effectiveness of each module in TBE. The results are reported in Table 3 and we can make two conclusions:

i. Each component can bring performance improvements on both HMDB and UCF101 benchmarks, indicating the effectiveness of these two components.

ii. Combining these two modules together can further boost performance, bringing 3.9% improvement on the UCF101 and 3.6% improvement on the HMDB51 compared to the baseline.

In the following section, we will perform visual analysis on BE and TI respectively.

Influence of backbone. We use 4 different backbones to verify the proposed method: (a). C3D, (b). R3D, (c). R(2+1)D, (d). I3D. The results are shown in Table 4, and it can be seen that TBE can bring significant improvements on the UCF101 and HMDB51 datasets with any backbone, which indicates TBE has strong generalization and generalization.

Table 3: The effectiveness of each component of TBE on the UCF101 and HMDB51. Both BE and TI can bring significant improvements, and the combination will further boost the performance.

| Method     | UCF101(%) | HMDB51(%) |
|------------|-----------|-----------|
| C3D        | 62.5      | 25.6      |
| C3D + TBE  | 66.4 (3.9↑)| 29.2 (3.6↑)|
| R3D        | 60.6      | 25.3      |
| R3D + TBE  | 63.2 (2.6↑)| 30.4 (5.1↑)|
| R(2+1)D   | 65.6      | 26.4      |
| R(2+1)D + TBE | 70.5 (4.9↑)| 31.9 (5.5↑)|
| I3D        | 64.3      | 27.2      |
| I3D + TBE  | 68.4 (4.1↑)| 32.2 (5.0↑)|

Table 4: Integrating TBE into three different network architectures all bring performance improvements.

| Method     | UCF101(%) | HMDB51(%) |
|------------|-----------|-----------|
| C3D        | 62.5      | 25.6      |
| C3D + TBE  | 66.4 (3.9↑)| 29.2 (3.6↑)|
| R3D        | 60.6      | 25.3      |
| R3D + TBE  | 63.2 (2.6↑)| 30.4 (5.1↑)|
| R(2+1)D   | 65.6      | 26.4      |
| R(2+1)D + TBE | 70.5 (4.9↑)| 31.9 (5.5↑)|
| I3D        | 64.3      | 27.2      |
| I3D + TBE  | 68.4 (4.1↑)| 32.2 (5.0↑)|

In this part, we further investigate how BE can boost the performance by visualizing the heatmaps with Class Activation Map (CAM) technique (Zhou et al. 2016) and the feature embedding space with t-SNE (Maaten and Hinton 2008). Specifically, we compare the effect of merely using the Rotation Prediction pretext with the effect of combining Rotation Prediction pretext with BE. In the experiments we adopt C3D as the backbone and pretrained the network on the split 1 of the UCF101 dataset.

Heatmap Visualization. Figure 4 compares the heatmaps of some HMDB51 samples pre-trained using Rotation Prediction pretext w/o and w/ BE.

Visualization Analysis of the Background Erasing

In this part, we perform visual analysis on BE and TI respectively.

Influence of backbone. We use 4 different backbones to verify the proposed method: (a). C3D, (b). R3D, (c). R(2+1)D, (d). I3D. The results are shown in Table 4, and it can be seen that TBE can bring significant improvements on the UCF101 and HMDB51 datasets with any backbone, which indicates TBE has strong generalization and generalization.
After integrating BE into Rotation Prediction pretext, the video features are more diverse after pre-training and are easier to distinguish after fine-tuning on the downstream task.

Visualization Analysis of the Temporal Inversion

In this experiment, we train three C3D networks under different configurations: (a). random initialization, (b). pre-trained on the split 1 of UCF101 with Rotation Prediction pretext in the self-supervised manner, (c). pre-trained on the split 1 of UCF101 with Rotation Prediction pretext combined with TI to constrain the feature symmetry in the time dimension. After pre-training, we further fine-tune these three C3D models on the split 1 of the HMDB51.

We select some videos from the test set of HMDB51 and visualize the corresponding temporal feature vectors extracted by the three networks above. Specifically, we select videos with aperiodic action and randomly crop them to the shape of $3 \times 64 \times 224 \times 224$, where 64 is the number of frames, 3 is the channel number and 224 is the scale of height and width. The scale of the feature representation output by the last convolutional layer before the global average pooling is $512 \times 8 \times 4 \times 4$. Spatial global max pooling is performed first and the average pooling along with all channels is followed, yielding a compressed temporal feature vector with a length of 8. Formally, we record the original video sample as $o$ and its reversed one as $r$, and the corresponding temporal feature is $f_o$ and $f_r$ respectively.

In order to measure the temporal symmetric of the high-level feature map between two videos, we define a measurement named $\text{sym}$ as follows:

$$
\text{sym}(o, r) = 1 - \text{MSE}(\text{norm}(f_o), \text{Reverse}(\text{norm}(f_r))),
$$

where $\text{norm}$ stands for normalization, $\text{Reverse}$ flips the feature vector in the temporal dimension, and $\text{MSE}$ is the mean square error. The stronger the symmetry of the model is in the temporal dimension, the closer the $\text{sym}$ is to 1.

The temporal feature vectors under three settings are all visualized in Figure 6. Compared with the random initialized model, we find the model pre-trained with Rotation Prediction pretext predicts more temporal symmetric feature, which indicates that solving the task of TI can enhance the model’s temporal modeling ability. In addition, we can observe that after using TI to constrain the symmetry in time, $\text{sym}$ is very close to 1, this means these feature vectors are highly symmetrical in time, as we expected.

Conclusion

In this paper, we propose a novel Temporal-sensitive Background Erasing (TBE) method for self-supervised learning. The proposed method minimizes the feature distance between the sample and sample variation constructed by Spatio-Temporal Transformation and Intra-Video Mixup. The proposed method is evaluated using different CNN backbones on two benchmark datasets. Experimental results show that the proposed TBE can be used as a regularization term with the current pretext tasks and outperforms existing methods for action recognition notably.

Our future work will study how to make use of other advanced consistency learning methods in the self-supervised fine-tuning on the downstream action recognition task, making videos more easily to be distinguished.
setting. On the other hand, besides action recognition, we will further develop self-supervised learning method based on the proposed TBE on other video applications like spatio-temporal action localization etc.

References

Benaim, S.; Ephrat, A.; Lang, O.; Mosseri, I.; Freeman, W. T.; Rubinstein, M.; Irani, M.; and Dekel, T. 2020. SpeedNet: Learning the Speediness in Videos. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 9922–9931. 2

Carreira, J.; and Zisserman, A. 2017. Quo vadis, action Net: Learning the Speediness in Videos. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 9922–9931. 2

Choi, J.; Gao, C.; Messou, J. C.; and Huang, J.-B. 2019. Why Can’t I Dance in the Mall? Learning to Mitigate Scene Bias in Action Recognition. In Advances in Neural Information Processing Systems, 853–865. 2

Diba, A.; Sharma, V.; Gool, L. V.; and Stiefelhagen, R. 2019. Dynamonet: Dynamic action and motion network. In Proceedings of the IEEE International Conference on Computer Vision, 6192–6201. 2

Gan, C.; Gong, B.; Liu, K.; Su, H.; and Guibas, L. J. 2018. Geometry guided convolutional neural networks for self-supervised video representation learning. In CVPR, 5589–5597. 2, 5

Gidaris, S.; Singh, P.; and Komodakis, N. 2018. Unsupervised Representation Learning by Predicting Image Rotations. In ICLR. 2, 3, 4, 5

Girdhar, R.; and Ramanan, D. 2020. CATER: A diagnostic dataset for Compositional Actions and TEmporal Reasoning. In ICLR. 1, 2

Han, T.; Xie, W.; and Zisserman, A. 2019. Video representation learning by dense predictive coding. In Proceedings of the IEEE International Conference on Computer Vision Workshops, 0–0. 2

Hara, K.; Kataoka, H.; and Satoh, Y. 2018. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In CVPR, 6546–6555. 1, 4

Hongyi Zhang, Moustapha Cisse, Y. N. D.; and Lopez-Paz, D. 2018. mixup: Beyond Empirical Risk Minimization. ICLR. 5

Kim, D.; Cho, D.; and Kweon, I. S. 2019. Self-supervised video representation learning with space-time cubic puzzles. In AAAI, volume 33, 8545–8552. 2, 3, 4, 5

Kuehne, H.; Jiang, H.; Stiefelhagen, R.; and Serre, T. 2013. Hmdb51: A large video database for human motion recognition. In High Performance Computing in Science and Engineering ?12, 571–582. Springer. 2

Lee, H.-Y.; Huang, J.-B.; Singh, M.; and Yang, M.-H. 2017. Unsupervised representation learning by sorting sequences. In ICCV, 667–676. 5

Li, Y.; Li, Y.; and Vasconcelos, N. 2018a. RESOUND: Towards Action Recognition without Representation Bias. In Proceedings of the European Conference on Computer Vision (ECCV). 1, 2

Li, Y.; Li, Y.; and Vasconcelos, N. 2018b. Resound: Towards action recognition without representation bias. In Proceedings of the European Conference on Computer Vision (ECCV), 513–528. 2

Luo, D.; Liu, C.; Zhou, Y.; Yang, D.; Ma, C.; Ye, Q.; and Wang, W. 2020. Video cloze procedure for self-supervised spatio-temporal learning. AAAI. 2

Maaten, L. v. d.; and Hinton, G. 2008. Visualizing data using t-SNE. Journal of machine learning research 9(Nov): 2579–2605. 6

Misra, I.; Zitnick, C. L.; and Hebert, M. 2016. Shuffle and learn: unsupervised learning using temporal order verification. In ECCV, 527–544. Springer. 2, 5

Noroozi, M.; and Favaro, P. 2016. Unsupervised learning of visual representations by solving jigsaw puzzles. In ECCV, 69–84. Springer. 2

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in pytorch. 4

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. 2

Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; and Paluri, M. 2015. Learning spatiotemporal features with 3d convolutional networks. In ICCV, 4489–4497. 4

Tran, D.; Wang, H.; Torresani, L.; Ray, J.; LeCun, Y.; and Paluri, M. 2018. A closer look at spatiotemporal convolutions for action recognition. In CVPR, 6450–6459. 4

Vondrick, C.; Pirsiavash, H.; and Torralba, A. 2016. Generating videos with scene dynamics. In NeurIPS, 613–621. 5

Vondrick, C.; Shrivastava, A.; Fathi, A.; Guadarrama, S.; and Murphy, K. 2018. Tracking emerges by colorizing videos. In Proceedings of the European conference on computer vision (ECCV), 391–408. 2

Wang, J.; Jiao, J.; Bao, L.; He, S.; Liu, Y.; and Liu, W. 2019. Self-supervised Spatio-temporal Representation Learning for Videos by Predicting Motion and Appearance Statistics. In CVPR, 4006–4015. 2

Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; and Van Gool, L. 2018. Temporal segment networks for action recognition in videos. IEEE transactions on pattern analysis and machine intelligence. 2

Wang, X.; Jabri, A.; and Efros, A. A. 2019. Learning correspondence from the cycle-consistency of time. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2566–2576. 2

Wang, Y.; and Hoai, M. 2018. Pulling actions out of context: Explicit separation for effective combination. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 7044–7053. 1, 2
Wei, D.; Lim, J. J.; Zisserman, A.; and Freeman, W. T. 2018. Learning and using the arrow of time. In CVPR, 8052–8060.

Weinzaepfel, P.; Harchaoui, Z.; and Schmid, C. 2015. Learning to track for spatio-temporal action localization. In ICCV, 3164–3172.

Xu, D.; Xiao, J.; Zhao, Z.; Shao, J.; Xie, D.; and Zhuang, Y. 2019. Self-Supervised Spatiotemporal Learning via Video Clip Order Prediction. In CVPR, 10334–10343.

Yao, Y.; Liu, C.; Luo, D.; Zhou, Y.; and Ye, Q. 2020. Video Playback Rate Perception for Self-Supervised Spatio-Temporal Representation Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 6548–6557.

Yun, S.; Han, D.; Oh, S. J.; Chun, S.; Choe, J.; and Yoo, Y. 2019. Cutmix: Regularization strategy to train strong classifiers with localizable features. ICCV.

Zhao, Y.; Xiong, Y.; Wang, L.; Wu, Z.; Tang, X.; and Lin, D. 2017. Temporal action detection with structured segment networks. In ICCV, 2914–2923.

Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; and Torralba, A. 2016. Learning deep features for discriminative localization. In CVPR, 2921–2929.