UrbanPipe Track on Fine-grained Video Anomaly Recognition Technical Report

Hao Wang, Jiahao Wang, Zhuojun Dong, Yuting Yang, Qianyue Bao, Fang Liu, Licheng Jiao
Key Laboratory of Intelligent Perception and Image Understanding
School of Artificial Intelligence Xidian University, Xi'an, China
{21171213809, 21171213808}@stu.xidian.edu.cn

Abstract

This technical report presents an overview of our solution used in the submission to ***ECCV DeeperAction Challenge 2022 UrbanPipe Track (Track 5) on Fine-grained Video Anomaly Recognition****. Video anomaly detection is important for industrial applications in the real world. In particular, the urban pipeline system is one of the most important infrastructures in a city and its robustness is a prerequisite for stable urban development.We sampled the input pipeline video clips at the specified frame interval, trained the pipeline videos using mmaction and PySlowfast framework, tried SlowFast, TSN, TPN, TAN, TimeSformer, Video-SwinB, Video-Swin-S, Video-SwinB-Ensembled, MVITv2-S and MVITv2-B, etc. The output results of different models are averaged to obtain the probabilities, and then trained by setting positive and negative samples to finally obtain the final category probability prediction results.*

1. Introduction

Video understanding is a popular area in deep learning research, and video anomaly detection is a subfield of video understanding.

Nowadays, the mainstream video anomaly detection methods can be roughly divided into several directions, one is the classification-based anomaly detection method [1] [2] [3], i.e., collecting a sufficient amount of normal anomaly samples and doing binary classification on this basis; the other is the feature reconstruction-based method [4] [5] [6], i.e., using a large number of normal samples to train the network, expecting the model to have good reconstruction ability for normal samples, and detecting anomalies by having large reconstruction errors when the model encounters anomaly samples.

This competition is actually a classification task. In pipeline video anomaly detection, for each video, there are one or more video-level labels, which means that for each video only the presence of anomalies can be determined without being able to locate a particular frame,, and if anomalies exist, the probability predictions for each anomaly category are output correspondingly.

2. Dataset

UrbanPipe is a fine-grained and multi-labeled video dataset.The UrbanPipe dataset consists of 9609 short videos containing 16 anomaly classes such as deposition, misalignment, obstacle, deformation, scum, leakage, and disjunction, of which 6399 videos are used for training and 3210 videos are used for testing. Each video is annotated with 1-5 tags per video due to the complexity of the pipeline situation where multiple defects often appear simultaneously.

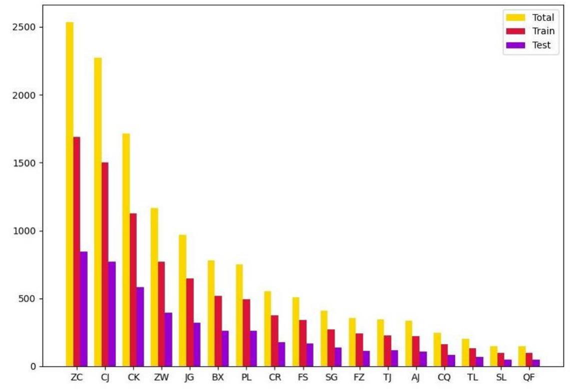

Figure 1. Histogram of the distribution of the number of 16 pipeline anomaly labels in the UrbanPipe dataset.

Figure 1 shows the histogram of the distribution of the number of occurrences of each label in the UrbanPipe dataset, from which it can be seen that the number of occurrences of the labels of heaving, leakage, tragic wall, branch pipe concealed connection, disconnections, flotsam, and tree roots is less.

3. Method

In this section, we describe our approach to video anomaly identification on the UrbanPipe dataset. First, some groundbreaking approaches in the field of video understanding and their innovations are presented. Then, we introduce the overall framework of our method for this task. Finally, we discuss learning long-tail category distribution strategies in video datasets and a multiple temporal resolution ensemble method for improving model generalization.

3.1. Mainstream method

I3D [7] takes a video clip as input, and forwards it through stacked 3D convolutional layers. A video clip is a sequence of video frames, usually 16 or 32 frames are used. The major contributions of I3D are: 1) it adapts mature image classification architectures to use for 3D CNN; 2) For model weights, it adopts a method developed for initializing optical flow networks in to inflate the ImageNet pre-trained 2D model weights to their counterparts in the 3D model. Hence, I3D bypasses the dilemma that 3D CNNs have to be trained from scratch.

SlowFast [8] is an efficient network with a slow pathway and a fast pathway. The network design is partially inspired by the biological Parvo- and Magnocellular cells in the primate visual systems. As shown in Figure 2, the slow pathway operates at low frame rates to capture detailed semantic information, while the fast pathway operates at high temporal resolution to capture rapidly changing motion.

Temporal Segment Network (TSN) [9] first divides a whole video into several segments, where the segments distribute uniformly along the temporal dimension. As shown in Figure 6, TSN randomly selects a single video frame within each segment and forwards them through the network. Here, the network shares weights for input frames from all the segments. In the end, a segmental consensus is performed to aggregate information from the sampled video frames.

Non-local [10] is a generic operation similar to self-attention [11], which can be used for many computer vision tasks in a plug-and-play manner. As shown in Figure 2, they used a spacetime non-local module after later residual blocks to capture the long-range dependence in both space and temporal domain, and achieved improved performance over baselines without bells and whistles.

TimeSformer [12], a new architecture for video understanding, is completely based on Transformer, using a unique mechanism called divided space-time attention to avoid complex computations between all sequences of image blocks. In addition, the scalability of TimeSformer makes it possible to train larger models on longer video clips. Finally, TimeSformer achieves SOTA results on several challenging behavior recognition datasets.

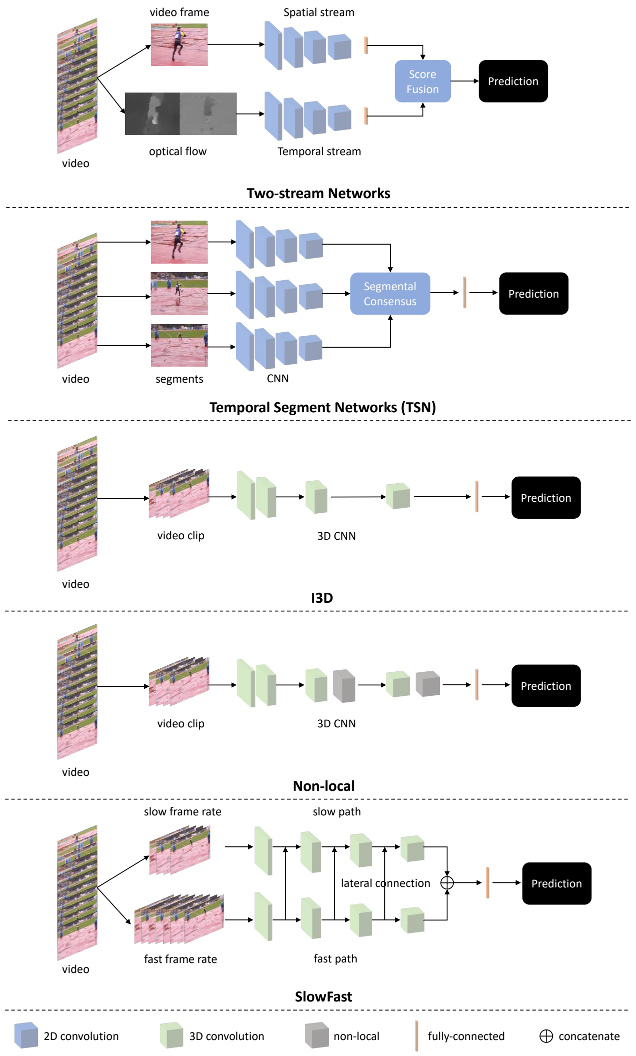

Figure 2. Workflow of five mainstream method: two-stream networks [9], temporal segment networks [9], I3D [7], Nonlocal [10] and SlowFast [8].

Video Swin Transformer [13] is initially described in ”Video Swin Transformer”, which advocates an inductive bias of locality in video Transformers, leading to a better speed-accuracy trade-off compared to previous approaches which compute self-attention globally even with spatial-temporal factorization.

Multiscale Vision Transformers (MViT) [14] serves as a general-purpose architecture for image and video classification and target detection. This optimized multi-scale ViT uses decomposed relative position embedding as well as residual truth connectivity. Since the modules in MViT can be easily migrated to the spatio-temporal domain, MViT can be easily used for video classification tasks.

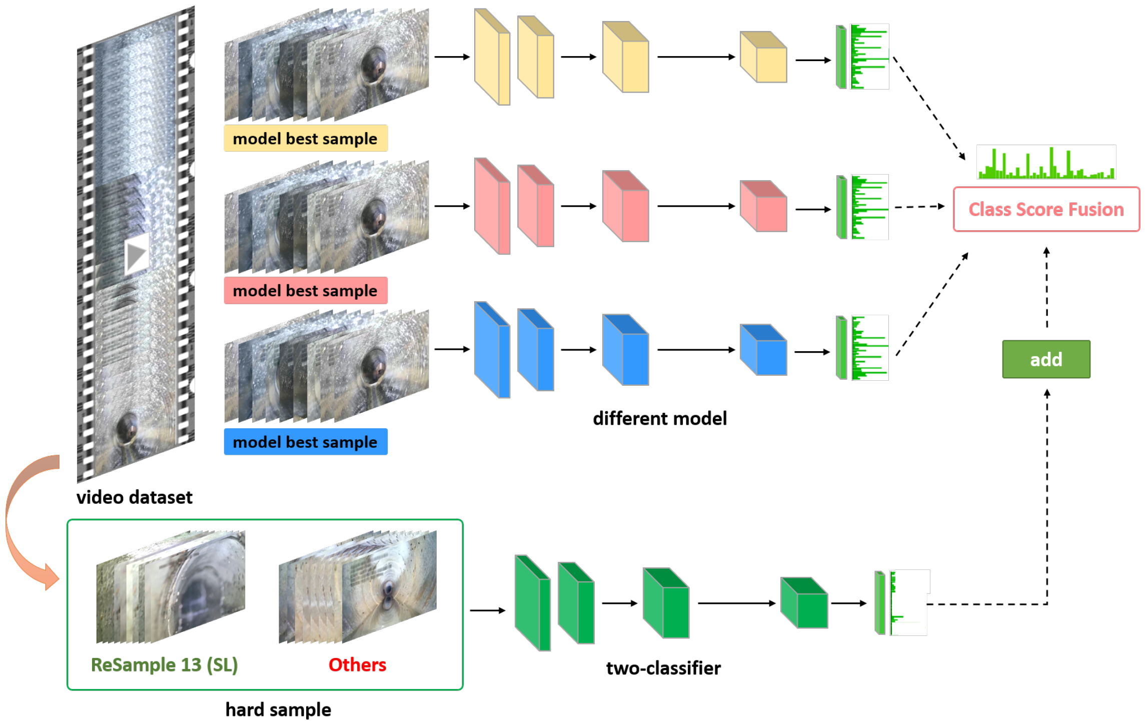

Figure 3. Workflow of our approach for video anomaly recognition.

3.2. Over Framework

Based on some previous work on video understanding, for the video anomaly multi-label classification task, we designed the entire pipeline, as shown in Figure 3. The framework is designed to detect the probability that the pipeline video corresponds to the class of all anomalies. Specifically, a video is first sampled at a specified frame interval for the input video clip. Sampling to a specified frame interval in an input video clip (experiments are needed to arrive at the most appropriate sampling method). For the Sigmoid probability outputs of the different models, we average all their results to obtain an integrated probability result. Then the data from a single category of them, i.e., the video has that label, is considered as a positive sample, and the rest of the data is used as a negative sample for training. Finally, a threshold is set for the prediction result of that category and added to the integrated probability result to get the prediction probability of the final anomaly category.

3.3. Long-tailed Learning

The number of each anomalous video in the training set varies from 98 to 1609, which reflects obvious long-tailed category distribution. The classes with fewer instances pose great challenges for deep learning based models on how to deal with the class imbalance problem.

We consider the decoupling representation learning strategy [15] to obtain the model that is capable of recognizing all classes well. Specifically, the training process is divided into two phases. In the first phase, we follow the normal training paradigm with standard randomly sampled data. In the second phase, we freeze all parameters of the model except the final classifier and retrain the classifier with class-balanced loss. For category balancing loss, the weights correspond to the log inverse of the number of categories. Such a strategy helps to further improve performance, especially on some classes with a small number of samples.

3.4. Ensemble on Multiple Temporal Resolutions

Integration learning can significantly improve the performance of the model in various tasks, where one of the cores of the variance reduction based approach is that different base learners are needed to learn different knowledge from the data, so as to improve the final generalization performance by the consensus of different base learners.

Bagging [16] is one of the representative algorithms.We start from the idea of Bagging, as shown in Figure 4, which differs from the original algorithm that trains a subset by random sampling, and we use different time-domain sampling rates to sample videos and obtain training sets with different time-domain resolutions, so as to train different base learners. Experiments show that our approach can significantly improve the integration results, and it also outperforms the traditional Bagging integration strategy because each base learner can use all the training videos and thus

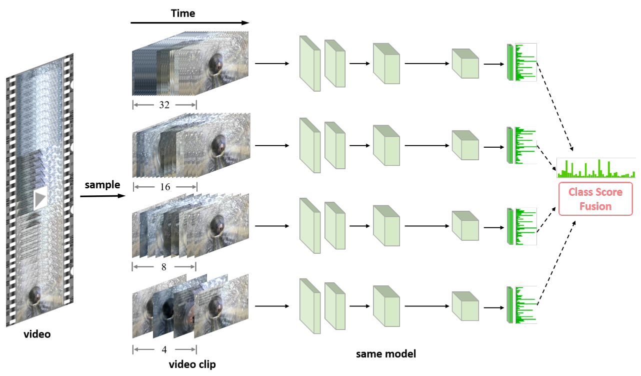

Figure 4. Single Fusion With Multiple Temporal Resolution.

achieve higher single-model performance.

3.5. Data Augmentation

Data augmentation is a commonly used method to improve model performance in deep learning, mainly used to increase the training data set and improve the generalization ability of the model, the data augmentation methods we used is described as follows.

Mixup [17]. Each time we randomly sample two examples (x_i, y_i) and (x_j, y_j). Then we form a new example by a weighted linear interpolation of these two examples:

$$\hat{x} = \lambda x_i + (1-\lambda)x_j \hat{y} = \lambda y_i + (1-\lambda)y_j \tag{1}$$

where $\alpha \in$ [0,1] is a random number.

Label Smoothing [18] is a regularization method in machine learning, usually used in classification problems, to prevent models from predicting labels too confidently during training and to improve poor generalization.

Label smoothing combines a uniform distribution with an updated label vector $\hat{y}_i$ to replace the traditional ont-hot encoded label vector y_{hot}:

$$\hat{y}_i = y_{hot}(1-\alpha) + \alpha/K \tag{2}$$

where K is the total number of categories for multicategorization and α is a small hyperparameter (generally taken as 0.1), i.e.

$$\hat{y}_i = \begin{cases} 1-\alpha, & i = \text{ target} \\ \alpha/K, & i \neq \text{ target} \end{cases} \tag{3}$$

In this way, the distribution after label smoothing is equivalent to adding noise to the true distribution to avoid the model being overconfident about the correct labels, making the difference between the output values of the predicted positive and negative samples less significant, thus avoiding overfitting and improving the generalization ability of the model.

RandAugment [20] allows the incremental sample space generated by data augmentation to be significantly reduced, so that it can be done in a bundle with the model training process, avoiding it being done as a separate pre-processing task. The RandAugment method is:

1) Set a set of operations, and the set of operations consists of multiple operations.

2) RandAugment has only two parameters: N and M. Where N is the number of operations used at each enhancement, and M is a positive integer, indicating that all operations are applied with magnitude M.

3) Use grid search, or a more high-end approach, to experiment on the complete data set, the complete network, and find the most suitable N and M.

3.6. Loss

We use BCEWithLogitsLoss to compute the multi-label class loss. This loss combines the Sigmoid layer and BCELoss in a single category, which performs better than using a sigmoid alone, followed by BCELoss. In the case of multi-label classification, assuming that there are N batches, each predicting n labels, the loss can be described as:

$$\begin{aligned} \ell_c(x, y) = L_c = \{l_{1,c}, l_{N,c}\}^{\top} \\ l_{n,c} = -w_{n,c}[p_c y_{n,c} \cdot \log \sigma(x_{n,c}) \\ + (1-y_{n,c}) \cdot \log(1-\sigma(x_{n,c}))] \end{aligned} \tag{4}$$

$l_{n,c}$ denotes the loss of a single sample on a class, w_n is the weight adjustment coefficient, p_c denotes the weight adjustment factor when a category takes the value of 1.

4. Experiments

The UrbanPipe dataset contains 6,399 videos in the training set. And there are 3,210 videos in the test set. Following the guidelines of the challenge, we use Average Precision (AP) to evaluate the recognition results on each defect category(we also treat normal class as a defect category). Then we average AP over all the categories to obtain mAP.

4.1. Implementation Details.

Training. The video classification was trained by the mmaction2 [19] and PySlowfast [21] framework, and all models were explored as classifiers in the development phase. The specific training details by mmaction2 are shown in Table 1. In Table 1, num clips, clip len, and frame interval are the configuration parameters for reading untrimmed videos. The num clips is defined as the number of clips the whole video will be divided into, clip len is defined as the number of frames actually selected for each clip, and frame interval is defined as the number of frames between each clip. Different configurations have an impact on the accuracy of the model. In video-swin-transformer (SwinB), for example, num clips is set to 1

model	bachbone	head	pre-train	train views	test views	test argument	dev map	test map
I3D	Resnet50	I3DHead	K400	32×2×1	32×2×10	ThreeCrop	58.8	57.1
SlowFast	Resnet50	SlowFastHead	K400	32×2×1	32×2×10	ThreeCrop	58.531	–
SlowFast	Resnet50	SlowFastHead	K400	32×2×1	32×2×10	ThreeCrop	59.633	–
SlowFast	Resnet101	SlowFastHead	K400	32×2×1	32×2×10	ThreeCrop	62.469	–
SlowFast	Resnet101	SlowFastHead	K400	32×2×1	32×2×10	TenCrop	62.534	60.574
TSN	Swin-B	TSNHead	K400	1×1×3	1×1×25	TenCrop	62.519	60.697
TAN	ResNet50	ClsHead	K400	1×1×8	1×1×8	TenCrop	62.037	60.097
TAN	ResNet101	ClsHead	K400	1×1×8	1×1×8	TenCrop	63.624	61.746
TPN	ResNet101	ClsHead	K400	32×2×1	32×2×6	TenCrop	62.72	61.136
TPN	ResNet101	ClsHead	K400	32×2×1	32×2×10	TenCrop	62.991	61.355
TimeSFormer jointST	VIT-B	TimeSformerHead	K400	8×32×1	8×32×1	ThreeCrop	59.905	–
TimeSFormer divST	VIT-B	TimeSformerHead	K400	8×32×1	8×32×1	TenCrop	61.042	–
TimeSFormer divST	VIT-B	TimeSformerHead	K400	8×32×1	8×32×4	TenCrop	63.176	61.704
TimeSFormer-HR divST	VIT-B	TimeSformerHead	K400	16×16×1	16×16×4	TenCrop	64.876	62.980
Video-Swin-Transformer	Swin-S	I3DHead	K400	32×2×1	32×2×1	TenCrop	64.834	63.079
Video-Swin-Transformer	Swin-B	I3DHead	K400	20×2×1	20×2×1	TenCrop	64.187	–
Video-Swin-Transformer	Swin-B	I3DHead	K400	32×2×1	32×2×1	TenCrop	66.364	64.881
Video-Swin-Transformer	Swin-B	I3DHead	SSv2	20×2×1	32×2×1	TenCrop	–	**66.169**

Table 1. **Training details for video classification networks based on mmaction2 [19]. The views represents clip_len × frame_iterval × num_clips** The "num_clips" is the number of clips the whole video will be divided into, the "clip_len" denotes the number of frames selected for clip, and the "frame_interval" is defined as the number of frames between each clip.

model	bachbone	head	pre-train	train views	test views	dev map	test map
MVITv2	VIT-S	TransformerBasicHead	K400	16 × 4	1 × 5	59.065	–
MVITv2	VIT-S	TransformerBasicHead	K400	16 × 4	3 × 10	66.766	65.235
MVITv2	VIT-S	TransformerBasicHead	SSv2	16 × 4	3 × 10	65.014	63.995
MVITv2	VIT-B	TransformerBasicHead	K400	32 × 3	1 × 5	–	63.624
MVITv2	VIT-B	TransformerBasicHead	K400	32 × 3	3 × 16	–	66.167
MVITv2	VIT-B	TransformerBasicHead	SSv2	32 × 3	3 × 16	–	65.217

Table 2. **Training details for video classification networks based on PySlowFast** [21].The "train views" is frame length× sample rate, The "test views" is crops × clips.

during training, and TTA (Test Time Augmentation) is performed by increasing num clips during testing to improve the test accuracy, which is used to cover the variable duration of untrimmed videos.

num_clips	Test MAP	Long-tailed Fine-tuning
8	64.663	65.034
16	65.734	66.161
20	66.169	**66.773**
32	65.964	66.556
Ensemble	–	**67.798**

Table 3. **The MAP of Video-SwinB at different num_clips with Long-tailed Learning Fine-tuning. This "Ensemble" is on Multiple Temporal Resolutions**.We used 8,16,20,32 resolutions of temporal sampling to integrate the model.

Table3 shows the Mean Average Precision of SwinB under different num clips in detail. We ensembled the models with four different temporal resolutions.

The specific training details based on the PySlowFast framework are shown in Table 2. In Table 2, the sampling_rate is defined as the frame sampling rate (interval between two sampledframes) and num_frames s defined as the number of frames to sample.

Model	Test MAP
SlowFast	60.574
TSN	60.697
TPN	61.355
TAN	61.746
TimeSFormer	62.980
MVITv2-S (K400, views=3x10)	65.235
MVITv2-S (SSv2, views=3x10)	63.995
MVITv2-B (K400, views=3x16)	66.167
MVITv2-B (SSv2, views=3x16)	65.217
Video-SwinB (nc=20)	66.773
Video-Swin-S (nc=32)	63.079
Video-SwinB-Ensembled	67.798
Avg Ensemble	**71.63**
Class Overlay	**72.923**

Table 4. Comparison between different backbones for video classification.

Inference. As shown in Table 4, we ensembled the SlowFast, TSN,TPN, TAN, TimeSformer, Video-SwinB, Video-Swin-S, Video-SwinB-Ensembled, MVITv2-S, and MVITv2-B models with different num clips parameters to determine the validation dataset mean average precision.

Overlay. Since the submitted results could only see the mean average precision, we considered that the poor results after averaging might be due to poor individual category results. Therefore, after observing the final ensemble results, we found that class 13 had a high prediction error rate in the test set, so we augmented that category with data resampling to obtain a new dataset, then froze the feature extraction process before the classification layer, trained only a two-category classifier, and added the probability of categories above a certain threshold in the prediction results to the integrated results. Performing the same process for categories 9, 12, and 16 can effectively improve the mean average accuracy on the test set.

5. Conclusion

In this work, we introduce the method designed for the DeeperAction competition, including long-tailed learning, ensemble on multiple temporal resolutions and resample two-classifier. We find that the model using the video clips for action recognition has a greater performance on proposals than the model using single frame. Increasing the segment size of video in inference can further improve the recognition mean accuracy precision in the QVPipe dataset. In the future we tend to go for an attention module designed to enhance the spacing between multi-tab classes rather than simply using integration to improve performance.

6. Acknowledgement

Throughout the writing of this dissertation I have received a great deal of support and assistance. I would first like to thank my tutors, for their valuable guidance throughout my studies. You provided me with the tools that I needed to choose the right direction and successfully complete my competition. I would particularly like to acknowledge my teammate, for their wonderful collaboration and patient support. Finally, I would not have been able to get in touch with this competition without the support of the organizer, DeeperAction, who provided a good competition environment and reasonable competition opinions.

Thanks to the support of the National Natural Science Foundation of China (No.62076192), Key Research and Development Program in Shaanxi Province of China (No.2019ZDLGY03-06), the State Key Program of National Natural Science of China (No. 61836009), the Program for Cheung Kong Scholars and Innovative Research Team in University (No. IRT 15R53), The Fund for Foreign Scholars in University Research and Teaching Programs (the 111 Project) (No. B07048), the Key Scientific Technological Innovation Research Project byMinistry of Education, the National Key Research and Development Program of China, and the CAAI Huawei MindSpore Open Fund.

References

[1] Yiru Zhao, Bing Deng, Chen Shen, Yao Liu, Hongtao Lu, and Xian-Sheng Hua. Spatio-temporal autoencoder for video anomaly detection. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 1933–1941, 2017.

[2] Waqas Sultani, Chen Chen, and Mubarak Shah. Real-world anomaly detection in surveillance videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6479–6488, 2018.

[3] Jia-Xing Zhong, Nannan Li, Weijie Kong, Shan Liu, Thomas H Li, and Ge Li. Graph convolutional label noise cleaner: Train a plug-and-play action classifier for anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1237–1246, 2019.

[4] Cewu Lu, Jianping Shi, and Jiaya Jia. Abnormal event detection at 150 fps in matlab. In *Proceedings of the IEEE international conference on computer vision*, pages 2720–2727, 2013.

[5] Bin Zhao, Li Fei-Fei, and Eric P Xing. Online detection of unusual events in videos via dynamic sparse coding. In *CVPR 2011*, pages 3313–3320. IEEE, 2011.

[6] Dan Xu, Elisa Ricci, Yan Yan, Jingkuan Song, and Nicu Sebe. Learning deep representations of appearance and motion for anomalous event detection. *arXiv preprint arXiv:1510.01553*, 2015.

[7] Joao Carreira and Andrew 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.

[8] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6202–6211, 2019.

[9] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016.

[10] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[12] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, volume 2, page 4, 2021.

[13] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3202–3211, 2022.

[14] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Mvitv2: Improved multiscale vision transformers for classification and detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4804–4814, 2022.

[15] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. *arXiv preprint arXiv:1910.09217*, 2019.

[16] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[17] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[18] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? *Advances in neural information processing systems*, 32, 2019.

[19] MMAction2 Contributors. Openmmlab's next generation video understanding toolbox and benchmark. `https://github.com/open-mmlab/mmaction2`, 2020.

[20] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

[21] Haoqi Fan, Yanghao Li, Bo Xiong, Wan-Yen Lo, and Christoph Feichtenhofer. Pyslowfast. `https://github.com/facebookresearch/slowfast`, 2020.