WHU-NERCMS AT TRECVID2019: INSTANCE SEARCH TASK

Longxiang Jiang[1], Jingyao Yang[1], Erxuan Guo[1], Fan Xia[1]
Ruxing Meng[2], Jingfeng Luo[2], Xiangyu Li[2], Xinyi Yan[2], Zengmin Xu[1,2,3], Chao Liang*
[1]National Engineering Research Center for Multimedia Software, School of Computer Science, Wuhan University
[2]Anview.ai
[3]School of Mathematics and Computing Science, Guilin University of Electronic Technology
cliang@whu.edu.cn

ABSTRACT

This paper presents the framework and results of our participation in the TRECVID 2019 Instance Search task. This year the instance search task aims at retrieving specific persons doing specific actions. In fact, this retrieval target can be divided into two parts, that is, person and action retrieval. So we tried two schemes: step retrieval and overall retrieval. On the one hand, we try to get the final retrieval results by retrieving the peolple and actions respectively and fusing them later. The advantage of this method lies in the independence of retrieval and the simplicity of later fusing. On the other hand, we try to get the final retrieval results by retrieving, tracking specific people and retrieving their actions. The advantage of this method is that it adopts targeted action retrieval for specific people to ensure the integrity of retrieval. The two retrieval schemes are complementary. With the proposed methods, our first scheme get 0.016 mAP and our second scheme get 0.017 mAP in automatic task according the evaluation, ranks 4 among all instance search teams.

1 Introduction

From 2010-2015 the instance search task tested systems on retrieving specific instances of objects, persons and locations. Recently in 2016-2018, it also retrieved specific persons in specific locations [1]. While in 2019, the new query type is to find the video of specific persons doing specific actions in all the video clips and make a record submission. Given a collection of test videos, a master shot boundary reference, a collection of topics that delimit a person in some example images and videos, and a set of predefined actions with example images or videos, locate for each topic up to 1000 shots most likely to contain a recognizable instance of the person doing one of the predefined actions [2]. As is shown in figure 1, the system is asked to retrieval shots with person Phil holding phone. Comparing with previous tasks, the difficulty of the new task is to recognize the action of a specific person in a continuous video frames. Based on the separability of retrieval task, we propose two schemes: step retrieval and whole retrieval.

For one scheme, we retrieval person and action respectively first and fuse them together later. Independent retrieval of people and actions makes the retrieval task clearer and easier, so that we can complete complex task in a simple way. When retrieving person, we adopted face recognition model to get person score. When retrieving action, we adopted 3D convolutional networks to extract the spatiotemporal features from videos and measured similarity with queries to get action scores. When fusing scores, we exploited weighting based balance and person identity based filter to combine the results of person retrieval and action retrieval to obtain the ranking result.

For the other scheme, we retrieval, track specific people and the retrieval their actions. Through the action retrieval of specific people, the action retrieval of non-specific people is excluded, so that the retrieval process is more targeted and effective. To find a specific action of a person, we adopted face recognition to determine the face ID of all the characters appearing in the video and bind the track ID of the track detected by the object tracking first. And then we adopted action recognition of consecutively tracked person target frames to get Action ID, so that each action of each character in all the given video clips has been identified and recorded separately. Finally, we use the specific action of the specific character that the task needs to get final results according to the record.

[*] Corresponding author

For person retrieval, our model is fine-tuned with a mixed dataset, which involved YouTube Face [3], IJB-A [4] and UMDFaces Datasets [5]. For action retrieval, our model is based on pre-trained kinetics [6] dataset which contains 400 action classes and a self-built dataset named Anview.ai Actions26, which consists of 8805 videos with 26 classes. And the combination of our approaches are:

- F_M_E_A_WHU_NERCMS_1 which is based on person retrieval, action retrieval and weight fusion
- F_M_E_A_WHU_NERCMS_2 which is based on person retrieval, action retrieval and filter fusion
- F_M_E_B_WHU_NERCMS_3 which is based on face recognition, object detection, object tracking and action recognition

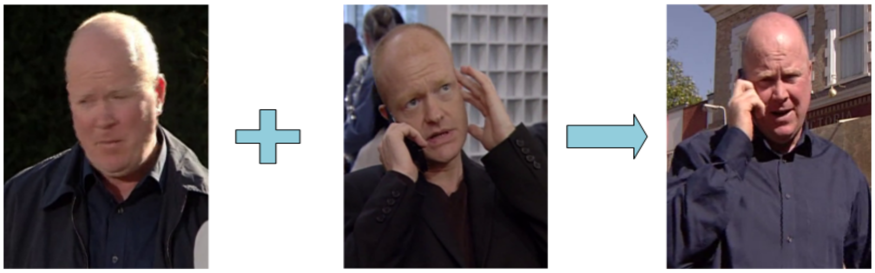


Figure 1: One of the topics of Trecvid 2019 INS, asked to retrieval *Phil holding phone*. (programme material copyrighted by BBC)

2 Our Framework

2.1 The first scheme

The proposed framework of our first scheme contains three parts as shown in Figure 2. The first is person retrieval module, which is based on face recognition, it focus on processing keyframes on which we can detect face box of persons. The second is action retrieval module, which is based on spatiotemporal feature extracted using 3D convolutional networks. With the two modules, we get both person and action retrieval scores. The third is result fusing module, we fuse the scores together to get the ranking result. The details of each module and related key technologies are demonstrated as following.

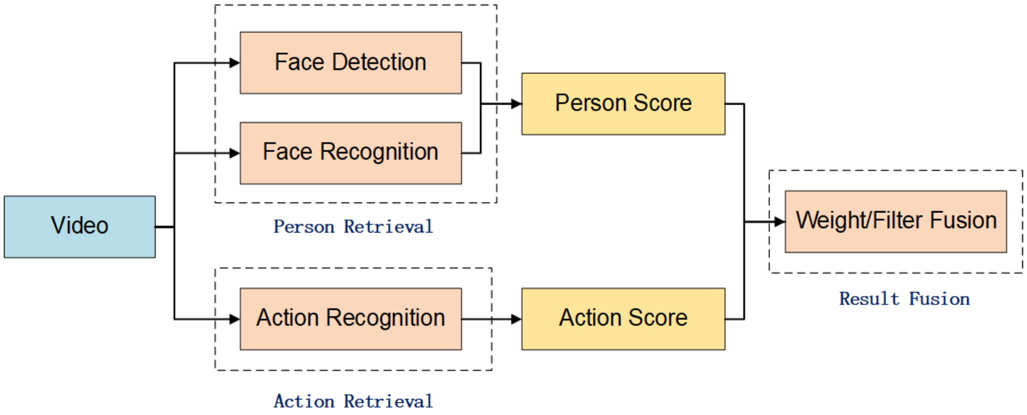


Figure 2: Our framework

2.1.1 Person retrieval

The different postures, illumination conditions, scales and other factors often affect the quality of face images in movies and TV plays, which poses a huge challenge to face recognition in movies and TV plays. To solve this problem, we do two steps in the person retrieval module. Firstly, we use the MTCNN model proposed in [7] to detect faces. Secondly, we use the method based on Center-loss proposed in [8] to build a face recognition model.

For the dataset, to match the target faces, we built our own face image library, which has 2664 images, a part of it is shown in Figure 3. In order to improve recognition accuracy, we collected a large number of actors of TV series images on the Internet, including different ages, postures and so on (these images are not appeared in this EastEnders TV series) as our reference dataset. In order to make it more consistent with the video face data distribution, we use the filtered mixed face datasets to fine-tune the pre-trained model, which involved YouTube Face [3], IJB-A [4] and UMDFaces Datasets [5].

For the face detection. We use MTCNN [7] face detection model trained on large-scale face detection dataset wilder face [9]. The model has several advantages. Firstly, the dataset contains the high variability of scale, posture and occlusion in the sample images, so it has strong robustness to the influencing factors. Secondly, MTCNN adopts cascade network structure, which reduces the computational load and ensures the detection accuracy, so it is conducive to large-scale data processing. Furthermore, MTCNN uses joint face detection and alignment multitask learning, which improves the accuracy of face detection.

For the face recognition. In order to decrease the internal-class discrepancy of deep facial features, we use the center loss + Softmax cost function proposed in [8] to build a face recognition model. The network has two convolution layers, three cascaded ResNet blocks, followed by a full connection layer which outputs a 512-dimensional feature vector. In the process of feature representation, firstly the features of the original face image and its horizontal flip image are extracted, and then the features are connected together to form 1024-dimensional feature vectors to represent the face.

With the above parts, we get the similarity matrix is obtained by querying each image in the reference data set first, and then the identity is determined according to the maximum similarity. Since an identity in a reference dataset contains multiple face images, we decrease the discrepancy of inner-class for the identity representation based post-processing method.

Figure 3: Part of our face library

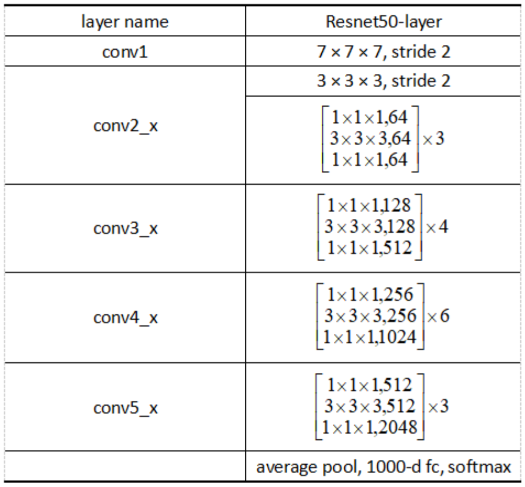
2.1.2 Action retrieval

Action retrieval is an important part of the task. In this part, we used C3D [10], a 3D Convolutional Networks, to learn spatiotemporal features. Our network is a residual network with different parameter layers. Similar with the traditional Resnet50 model [11], it mainly contains a convolution layer, a max pool layer, four bottleneck blocks, followed by an average pool layer and a fully connected layer. The most difference between our network and conditional Resnet50 is all the kernels are 3-dimensional. The network architecture is presented in figure 4.

To extract C3D feature, we use the Resnet50 model which is pre-trained on kinetics dataset [6] to initialize the network. For the dataset preparation, we split each video into 16 frame long clips with a 8-frame overlap between two consecutive clips. These clips are passed to the C3D network to get output before the final fc layer. In order to compute similarity fastly with matrix multiplication operation and eliminate the influence of feature scale, we refined the feature with L2 normalization and centralization. Final we use these features to compute the action similarity scores based cosine distance.

2.1.3 Result fusion

In result fusing module, we propose two different fusing methods. One is weight fusing and the other is filter fusing. With the person retrieval and action retrieval, we get two similarity score list for every topic, let them f_face



layer name	Resnet50-layer
conv1	7 × 7 × 7, stride 2
conv2_x	3 × 3 × 3, stride 2 $\begin{bmatrix}1\times1\times1,64\\3\times3\times3,64\\1\times1\times1,64\end{bmatrix}\times3$
conv3_x	$\begin{bmatrix}1\times1\times1,128\\3\times3\times3,128\\1\times1\times1,512\end{bmatrix}\times4$
conv4_x	$\begin{bmatrix}1\times1\times1,256\\3\times3\times3,256\\1\times1\times1,1024\end{bmatrix}\times6$
conv5_x	$\begin{bmatrix}1\times1\times1,512\\3\times3\times3,512\\1\times1\times1,2048\end{bmatrix}\times3$
	average pool, 1000-d fc, softmax

Figure 4: 3D Resnet-50 structure

Table 1: Fusing Method

Score lists	Fusing method	Method index	Submit index
$f\_face + f\_action$	weight	A	INS.fusion_v1_main INS.fusion_v1_progress
$f\_face + f\_action$	filter	B	INS.fusion_v2_main INS.fusion_v2_progress

and f_action respectively. We exploited weighting based and person identity filter based method to combine the results to get the ranking result. Our submitted 4 results based on our first scheme are generated by combining score lists with the fusing methods, which are listed in table 1.

Before fusing, we normalize all the score lists range from 0 to 1 by formula 1

$$f = \frac{f - min(f)}{max(f) - min(f)} \tag{1}$$

For method A, we fuse f_face and f_action together using weight fusion, the formula is shown in 2 and the weight α is set to 0.5.

$$f = \alpha * f\_face + (1 - \alpha) * f\_action \tag{2}$$

For method B, we adopt face filter method to fuse the scores. Note that the face library has all the actors appeared in the TV series, the detected face must belong to a certain actor. According to the largest score, we assign detected face to a actor. However, the face is not always recognized correctly due to the complex situation in the video, thus we refined the actor ID of the face which tack smooth, that is to say, we combine the bounding boxes toghter which have large Intersection over Union and have large possibility to be the same person, and then we assigned actor ID to all the face boxes. Thus, given a shot and a target actor, we can conclude whether the actor appeared in the shot. Based on above, we filter out all the shots without target face, and then according to the f_action, we rank the remained shots to get fusing result.

2.2 The second scheme

The specific process of the second framework we proposed is shown in Figure 5. It consists of four modules. The first module is face recognition, the second module is object detection, the third module is object tracking, and the fourth module is action recognition. Through face recognition, object detection and object tracking to determine a person's name and location, then we send their continuous multi-frame video clips into the action recognition module for identification, finally for each person and its actions in each video can be get the corresponding result.

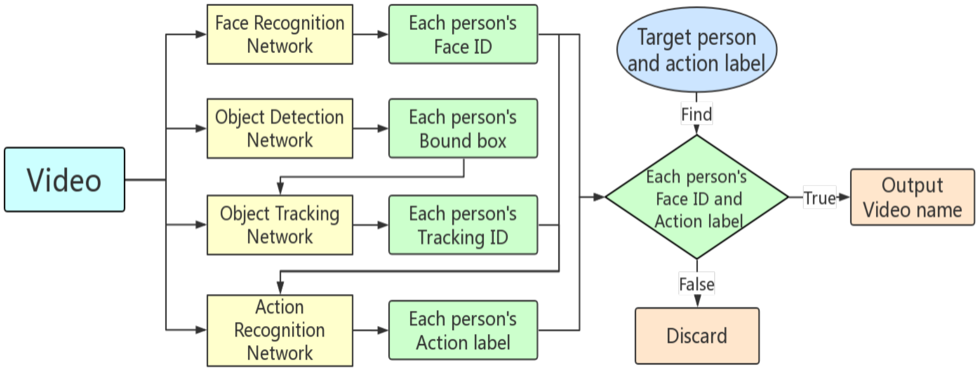


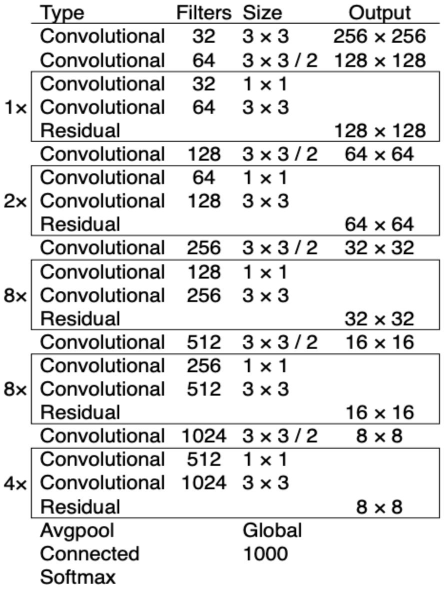
Figure 5: Our framework

2.2.1 Face Recognition

In the face recognition module, we adopted face recognition using the same method with the person retrieval of our first scheme.

2.2.2 Object Detection

In the object detection module, a video frame is used as an input of the neural network, and the position of each character target bounding box and the associated category detected by the neural network are returned to the output layer.



	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 6: Darknet-53 Structure

In the forward propagation process of the object detection algorithm, the tensor size transformation is realized by changing the step size of the convolution kernel. The backbone will reduce the output feature map to $1/32$ of the input, so the input picture is required to be multiple of 32.

The algorithm uses the Darknet-53 network [12]. The specific structure is shown in Figure 6. In addition to the final average pooling, the connection classification is performed. There is no pooling layer and full connection layer in the whole network structure, including the add layer 23 layer (mainly used for res_block). Composition, each res_unit needs an add layer, a total of $1 + 2 + 8 + 8 + 4 = 23$ layers), the number of Batch Normalization layer and

Leaky ReLU layer is exactly the same (72 layers), the performance in the network structure is: A layer of BN will be followed by a layer of LeakyReLU. The convolution layer has a total of 75 layers, of which 72 layers will be followed by a combination of BN + LeakyReLU to form the basic component DBL. Both upsampling and concat are 2 times, and each res_block is padded with a zero, for a total of 5 res_blocks.

The Darknet-53 outputs 3 feature maps of different scales with a depth of 255 and a side length ratio of $13 : 26 : 52$. The algorithm divides the entire input graph into nn cells, each cell predicts 3 boxes, each box needs to have five basic parameters (x, y, w, h, confidence), and because of training The COCO data set has a probability of 80 categories, so the depth is $255 = 3(5 + 80)$. Because the feature maps of different scales of the output require concat stitching, it is necessary to use $(2, 2)$ upsampling to ensure that the scale of the feature map is consistent.

When the final output of the target box is performed, the three boxes predicted by the network are respectively adapted to the set anchor, and the logistic regression is used to perform objectness score on the content surrounded by each anchor, and the anchor prior is selected according to the score to perform the predict output the forecast box.

The target frame of all characters detected in the video frame is filtered out and passed to the object tracking module.

2.2.3 Object Tracking

In the object tracking module, the object block detected by the above module object is used as an input, and each object character is tracked by an algorithm, and a unique Track ID and a predicted bounding box are given.

In the object tracking algorithm [13], the Kalman filter is used to calculate the object tracking $\hat{x}_t^-$ and covariance matrix Σ_t^- generated by the object detection frame of the previous frame, as shown in formula 3.

$$\begin{aligned} \hat{x}_t^- &= F\hat{x}_{t-1} + u_t \\ \Sigma_t^- &= F\Sigma_{t-1}F^T + Q \end{aligned} \tag{3}$$

The covariance matrix Σ_{t-1} of the target frame of the previous frame is the coordinate $\hat{x}_{t-1}$ of the target frame of the previous frame, F which is the state transition matrix, F^T which is the transposition of the state transition matrix F , u_t which is the update matrix and Q is the process noise.

Calculate the IOU of the object tracking frame generated by the previous frame and the object detection frame of the current frame, obtain the largest unique match of the IOU through the Hungarian algorithm, and then remove the matching pair whose matching value is less than the preset threshold. The Kalman filter is updated with the object detection frame matched in the current frame, the Kalman gain, the state update and the covariance update are calculated, and the state value update value is output as the object tracking frame of the current frame. Calculate the Kalman gain, as in formula 4:

$$K_t = \Sigma_t^- H^T (H\hat{x}_{t-1}H^T + R)^{-1} \tag{4}$$

Wherein, H for the observation matrix, the transposition H^T of the permutation matrix H, R is state transition noise, and the covariance Σ_t^- of the target frame of the current frame prediction is the object detection frame coordinate $\hat{x}_{t-1}$ of the $t-1$ time. Then, according to the calculated Kalman gain K_t, the coordinates $\hat{x}$ of the current frame object detection frame and the covariance matrix Σ_t are calculated, $\hat{x}_t^-$ which is the value of the object tracking frame at time t, and the coordinate formula $\hat{x}_t$ of the object tracking frame is as formula 5

$$\hat{x}_t = \hat{x}_t^- + K_t(y_t - H\hat{x}_t^-) \tag{5}$$

The covariance Σ_t update formula is as formula 6:

$$\Sigma_t = (1 - K_t H)\,\Sigma_t^- \tag{6}$$

Then, according to the covariance matrix Σ_t of the current prediction and the coordinates $\hat{x}_t$ of the current frame object detection frame, the coordinates $\hat{x}_{t+1}$ and covariance matrix Σ_{t+1}^- of the object detection frame of the next second are predicted, F is a state transition matrix, F^T is a transposition of the state transition matrix F, u_{t+1} which is an update matrix, Q is the process noise, the prediction is shown in formula 7:

$$\begin{aligned} \hat{x}_{t+1}^{-} &= F\hat{x}_t + u_{t+1} \\ \Sigma_{t+1}^{-} &= F\Sigma_t F^T + Q \end{aligned} \tag{7}$$

Reinitialize the tracker for targets that do not match in the current frame. The consecutive frames of the target pedestrians with the same number are combined into a object tracking queue. After the pedestrian object tracking queue length reaches the set frame number threshold, all object tracking frames are connected in series to form a object tracking area, and the object tracking area is sent to the action recognition module.

2.2.4 Action recognition

In the action recognition module, continuous frames of each person's object tracking area are taken as input, and different action types are classified by the neural network.

To train the action recognition model, we built a dataset named Anview.ai Actions26, which consists of 8805 videos with 26 classes, some of which are shown in Figure 7.

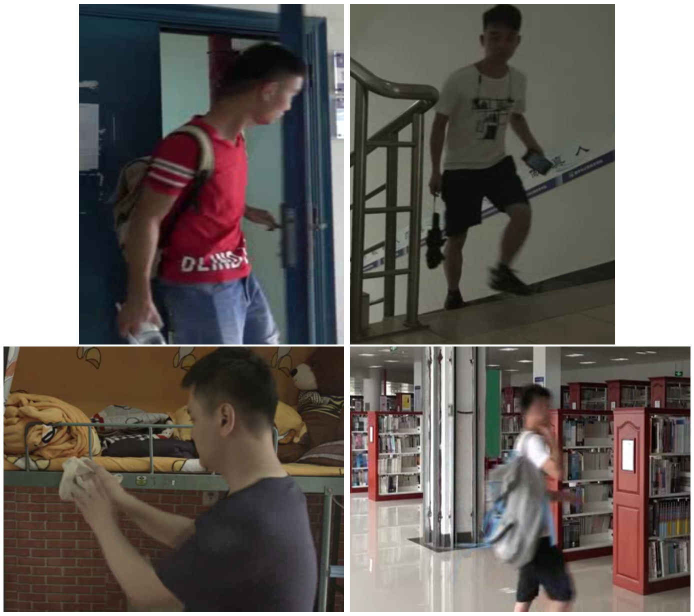


Figure 7: Anview.ai Actions26 Dataset

The action recognition neural network mainly uses the Multi-fiber Unit [14] as the basic network construction unit. As shown in Figure 8, the Multi-fiber Unit slices the entire residual module into multiple parallel and independent branches according to the channel, and uses the Multiplexer module to use Combine information between fibers in the form of residual connections.

The overall network structure is based on the ResNet-18 baseline. The specific structure is shown in Figure 9. After the video frames are input into the network, they are first convolved and pooled, and then passed through 16 3D Multi-fiber Units to obtain the feature map. Use the Global Average Pool compression feature to finally fully connect the output classification.

3 Results and Analysis

The final results of our submitted runs on Instance Search task of TRECVID 2019 are shown in Table 2. Through the results, our analysis and summary are as follows:

- The performance of tracking a specific person continuously and detect their actions is better than to detect tasks and actions separately and then fuse them later.

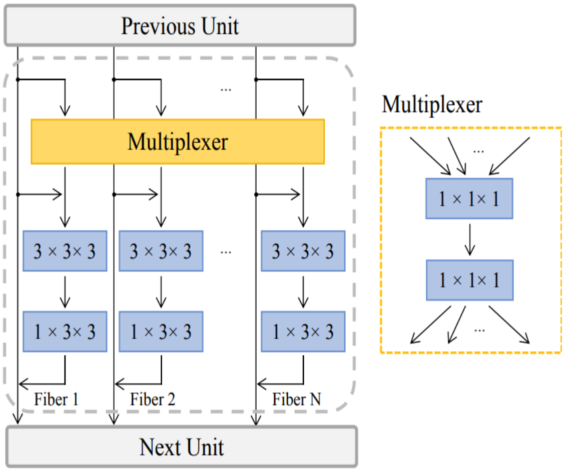


Figure 8: The internal structure of each Multi-fiber Unit

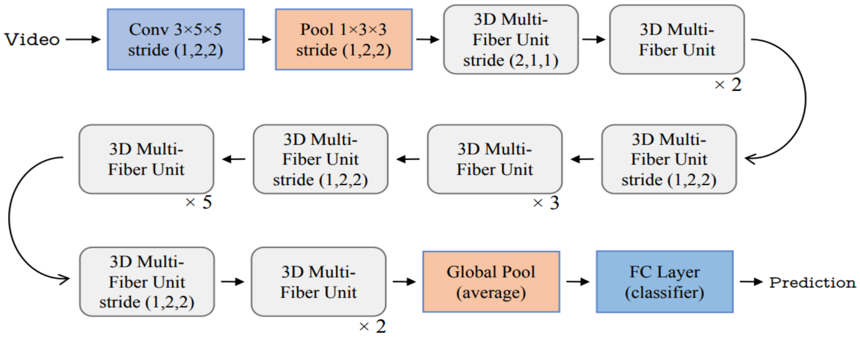

Video
Conv 3×5×5 stride (1,2,2)
Pool 1×3×3 stride (1,2,2)
3D Multi-Fiber Unit stride (2,1,1)
3D Multi-Fiber Unit
× 2
3D Multi-Fiber Unit stride (1,2,2)
3D Multi-Fiber Unit
× 3
3D Multi-Fiber Unit stride (1,2,2)
3D Multi-Fiber Unit
× 5
3D Multi-Fiber Unit stride (1,2,2)
3D Multi-Fiber Unit
× 2
Global Pool (average)
FC Layer (classifier)
Prediction

Figure 9: The overall architecture of 3D Multifiber Network

- In the score fusing module, the weighting fusing method and the method of person identity based filter have similar effect. While the exploration of the more effective fusing method should not stop.

- Through the test, we found that the accuracy of character retrieval is higher than that of action in our work. Considering the reason for it, we only used pre-training model to extract features instead of using data to fine-tune the network. It turns out that the model used off-the-self can not suit well in INS dataset, thus need retrain or fine-tuning before making use of, how to train CNNs with unsupervised or weakly supervised way is worth considering in our future work.

Table 2: Automatic Result

Num	ptype	exampleSet	mAP	Runid	Scheme	Fusion method
1	F	E	0.016	F_M_E_A_WHU_NERCMS_1	1	A
2	F	E	0.016	F_M_E_A_WHU_NERCMS_2	1	B
3	F	E	0.017	F_M_E_B_WHU_NERCMS_3	2	-

Acknowledgement

Thanks for the great support to our work by associate professor Chao Liang, who guided us from the begin to the end of the INS task. Thanks for Zengmin Xu in School of Mathematics and Computing Science, Guilin University of Electronic Technology, who provides us technical guidance. We also thank BBC for the material.

References

[1] Dongshu Xu, Longxiang Jiang, Xiaoyu Chai, Jin Chen, Li Jiao, Jiaqi Li, Shichen Lu, Han Fang, and Chao Liang. Whu-nercms at trecvid 2018: Instance search task.

[2] George Awad, Asad Butt, Keith Curtis, Yooyoung Lee, Jonathan Fiscus, Afzal Godil, Andrew Delgado, Alan F. Smeaton, Yvette Graham, Wessel Kraaij, and Georges Quénot. Trecvid 2019: An evaluation campaign to benchmark video activity detection, video captioning and matching, and video search & retrieval. In *Proceedings of TRECVID 2019*. NIST, USA, 2019.

[3] L. Wolf, T. Hassner, and I. Maoz. Face recognition in unconstrained videos with matched background similarity. In *Computer Vision and Pattern Recognition*, 2011.

[4] Brendan Klare, B. Klein, E. Taborsky, A. Blanton, J. Cheney, K. Allen, P. Grother, A. Mah, and A. K. Jain. Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a. 2015.

[5] Ankan Bansal, Anirudh Nanduri, Carlos Castillo, Rajeev Ranjan, and Rama Chellappa. Umdfaces: An annotated face dataset for training deep networks. 2016.

[6] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, and Andrew Zisserman. The kinetics human action video dataset. 2017.

[7] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Qiao Yu. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016.

[8] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *European Conference on Computer Vision*, 2016.

[9] Shuo Yang, Luo Ping, Change Loy Chen, and Xiaoou Tang. Wider face: A face detection benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[10] Tran Du, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *IEEE International Conference on Computer Vision*, 2015.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Sun Jian. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[12] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. 2018.

[13] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. 2016.

[14] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. Multi-fiber networks for video recognition. 2018.