Video Classification of Farming Activities with Motion-Adaptive Feature Sampling

He Liu, Amy R. Reibman, Aaron C. Ault, and James V. Krogmeier
Department of Electrical and Computer Engineering, Purdue University, West Lafayette, USA

***Abstract*—Recently, video has been applied in different industrial applications including autonomous driving vehicles. However, to develop autonomous farming vehicles, the video analysis must be targeted for specific farming activities. So an important first step is to classify the videos into their specific farming activity. In this paper, we propose a video classification framework that includes two branches that process videos differently based on their motions. A gradient-based method is proposed for separating videos into two subsets which are then processed by different feature sampling strategies. The result shows that two motion-based feature sampling strategies provide more efficient features; thus better classification performances are achieved. We also discuss how the feature sampling strategy influences the classification accuracy and the computational efficiency. In addition to farming videos, this proposed system can also be applied to classify videos captured from various camera movements, such as hand-held or first-person cameras.**

***Index Terms*—video classification, feature sampling, farming, agriculture**

I. Introduction

Cameras are one of the most important sensors used in autonomous vehicles, because they can provide fast and accurate information about surrounding areas in real-time. In agriculture, there are different types of large moving machines working on the farm, such as tractors and combines. Towards the goal of automating farm vehicles, cameras like dashcam, are placed on these big machines to sense the environment. However, operating farming machines is not easy since operators need to pay attention to many aspects, such as controlling of the vehicle, controlling of any attachments, and detecting anything anomalous. These each depend on the exact farming activity, like planting, spraying, harvesting and tillage. For example, harvesting needs to monitor the on reel speed and orientation, but for tillage it is more critical to minimize field overlap and avoid trenching. To learn from the videos captured on the farming vehicles, one important step is to separate videos into different farming activities. But farmers cannot document the activities as they happen, because they must focus on getting their work done in a narrow time window. Therefore, we focus on activity classification with farming videos captured by dashcams.

Classifying these farming activities using dashcam videos poses three challenges. The first challenge is the unconstrained capturing environment. With a working machine in the field, operators could place the camera at different positions to get different views, in order to monitor various aspects of the activity. Such different camera positions can cause significant image structure variations. Also the plants in the field have very different colors and shapes, depending on weather and season. In addition, there are no rigid objects to track in the field, which means it is hard to find and match robust feature points.

This work was supported by the Foundation for Food and Agricultural Research Grant #534662 and the Open Ag Technologies and Systems (OATS) Group.

Fig. 1. Some farming video frames from vehicle mounted cameras: chopping corns (top), tillage (middle), harvesting (bottom). Images on the left are in the training data; images on the right are in the test data. Notice that these videos are captured with different camera positions and various angles.

Figure 1 provides some examples of images from our farming videos. Each row shows images from the same category but with different capture angles or colors. The first two images captured from two different fields while chopping corn silage. The second row shows two tillage images with two camera angles. The last row presents two harvesting images with different camera positions. The left column images are chosen from the training set, and the right column images are from the testing set. We show below that image-based features are inadequate alone to effectively classify our categories. Therefore, in this paper we focus on video attributes and consider both spatial and motion information.

The second challenge is the variety of very different video motions. Dashcam videos normally have three types of motion conditions: very low or no motion, object motion and camera motion. In many instances, motions can be very useful to distinguish activity categories, but not always. Camera movements can dominate motion features, for example when vehicle is slowing down. If these motions are used as features, all other activities which have a similar slowing down movement might be incorrectly classified. In this case, camera motion becomes

a problem in the feature extraction process.

The third challenge is the limited number of videos. To our knowledge, currently there are no large scale farming video datasets available in either the classification or the farming community. Because a farming season is relatively long, which means most farming activities take place only once a year and it is hard to collect a large amount of data in a short time period.

In this paper, we propose a two-branch video classification framework to deal with the unpredictable video motions from dashcams. In this framework, the videos are separated into two subsets using dense optical flow analysis. One subset of videos has camera motions only or no motion at all, while another set has motions generated by the activity or object motion. To extract positions from two subsets of video separately, we use dense sampling [1] and proposed the concept of Fixed Position Trajectories (FPT). Cuboid methods [2] are applied to build feature descriptors for both subsets, but the cuboid designs are different. In our designs, features for videos containing object motions are extracted more densely in the temporal domain, while the features for camera-motion or low-motion videos are extracted more densely from the spatial domain. By applying different sampling and extracting strategies on different videos, the generated features are more efficient and informative, which makes this framework more adaptive to videos with various motion situations.

We have three major contributions in this work. First we propose a video classification framework that classifies videos differently based on their motion situations. Secondly, we summarize the motion types of dashcam videos and develop a fast method to separate them. Thirdly, we introduce a new feature sampling strategy to specifically extract motion features. This paper is organized as follows. In section II, we introduce some previous methods and algorithms applied for video classification. Then we explain our proposed framework together with video motions and our feature extraction method in section III. Finally we present two experiments which compare our framework with previous approaches and analyze the feature efficiency in section IV.

II. Previous Work

Image classification has been studied for years. Spatial image features including color and texture, and robust feature points, are effective for classification problems. Recently, Convolutional Neural Networks (CNN) [3] show much better performance. But we show below that image based spatial features are not sufficient for our task.

In video feature extraction, there are many sampling methods to select interesting feature positions, such as the cuboid detector [2], Space-Time Interest Points (STIP) and dense trajectories (DT) [4]. STIP is designed to select points from a 3D volume by computing the gradient matrix of each position and thresholding the trace and determinant. Trajectory-based feature extraction methods were developed for studying on human action recognition, and each motion trajectory contains a series of feature points over time. In [4], interest points are traced based on Kanade Lucas Tomasi (KLT) trackers or dense optical flow and then merged as trajectories. But not all the features generated by such methods are useful, so methods have been developed to reject inefficient positions. Page rank and visual similarity graphs were designed in [5] to prune static features by finding regions of interest. In [6], the Improved Dense Trajectory (IDT) method was proposed with a trajectory sampling method that incorporates human detection and homography estimation.

After these interest positions are identified, local feature descriptors are used to encode information in 3D video volumes. Some widely-used descriptors are Histogram of Oriented Gradients (HOG) [7], Histogram of Optical Flow (HOF) [8] and Motion Boundary Histograms (MBH) [9]. HOG and HOF are directly extracted from video frames and dense optical flow, but feature descriptors like MBH and Histograms of Motion Gradients (HMG) [10] are computed based on either spatial gradient or temporal gradient of optical flow. Such gradient-based descriptors are more robust to camera motions.

Video classification methods include standard methods and approaches using CNN. Standard classification methods normally include three steps: feature extraction, feature encoding and classifier training, such as [6] [11] [12] . Widely-used encoding methods include Fisher encoding [13] and Vector of Locally Aggregated Descriptors (VLAD) [14], and the one-vs-rest Support Vector Machine (SVM) is one of the most commonly-used classifiers. Recently, more CNN-based approaches have appeared, such as the two-stream architecture [15] and [16] with its fovea stream and context stream.

Many video datasets are available for classification research. Human action recognition is a widely-studied topic and some related public datasets are UCF-101 [17], HMDB [18] and Sport-1M [16]. The Sport-1M dataset is the largest; it includes one million YouTube video clips with 487 different sports categories. However, there are no farming-related outdoor activity video datasets available. In the image classification field, there are some datasets for scene classification purposes. The SUN dataset [19] includes 899 different categories including indoor, urban and natural scenes. The Places dataset [3] is a much larger scene-centric dataset, and the Places-CNN [3] is trained on this for scene classification. Both datasets provide some farming-related categories, but their categories are not specific enough to separate our video frames into different farming activities.

III. Proposed Method

In this section, we first discuss the video motions and our proposed framework. Then we introduce our motion separation method and the feature extraction methods in detail.

A. Two-branch framework

Our two-branch framework is inspired by the motions of farming video. Our video clips normally have three types of motions: static, camera motions and object motion. Static videos have no motion at all or are slowly shifting or shaking. Motion-based feature extraction methods such as IDT are not effective on these static regions. The second type is

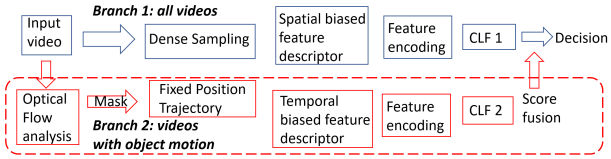

Fig. 2. Flow chart of the classification framework. Branch 1 (upper row) is primary and works on all videos, branch 2 (in red box) only works on object motion videos.

camera motion. The motion in these scenes are generated only by the camera movement or the vehicle movement. This occurs when the vehicle is driving and the outside objects are moving in the reverse direction. Camera motions are useful in detecting vehicle motion, but not effective for classifying farming activities. The third type of videos contains object motions that are generated by the objects themselves. These motions are robust to camera movements, and yet capture the movement pattern of the activity. Therefore, they are the most critical features.

Based on the video motion analysis, simply applying the same extraction methods on all types of videos cannot provide efficient features. So we propose two different classification branches as shown in Figure 2, where both branches (upper and lower rows) include a complete classification pipeline. While they have different feature extraction steps, the two classification branches both have these classification steps: Principal Component Analysis (PCA) feature dimension reduction, Fisher encoding and one-vs-rest SVM training. In this framework, the upper path is primary so all videos are processed by the first branch. The optical flow analysis is performed on the input video and used to determine if it contains enough object motion. This analysis performs spatial segmentation and produces spatial-temporal masks that indicate the object-motion regions. If no object-motion regions are found, the second (lower red) branch is disabled and the result of the first branch is final. If the video has enough object motions, the second branch is activated to process that video. Then both branches are enabled, and the final decision is determined by a fusion process. Ideally the fusion process should be a model trained using training videos as was done in [1] and [20]. Unlike [20] that trains a SVM classifier to merge results from different classifiers, we simply add the scores from two classifiers and choose the category with the highest score. Because the number of our training videos is limited, we avoid extra training steps that may cause overtraining.

B. Object motion separation

As discussed, object motion is the most critical motion feature for activity separation. Camera motions produce very strong features, yet they are not unique to a particular activity. The goal of the first optical flow analysis is to separate the potential object motion regions from the camera motions. In [6], a homography is estimated for consecutive frames to compensate for the camera motion. But in our agricultural videos, there are no robust feature points for homography estimation because there are no rigid objects in the field. The MBH feature is able to cancel camera motion by computing

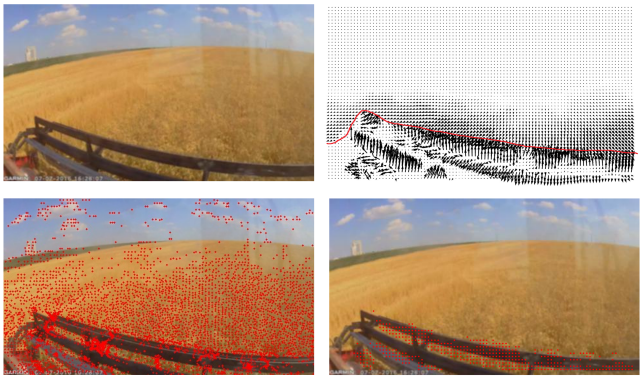

Fig. 3. The feature sampling method comparison on a harvesting video. Top left: the original image, top right: the optical flow at this frame, bottom left: the trajectories positions from IDT [6], bottom right: sampled positions using our motion separation method. In the optical flow image, the hand drawn red line partitions the object motions from the camera motions. In two lower images, red points represent the positions of trajectories for feature extraction.

a spatial gradient on optical flow, but it also removes useful motions and only leaves motion boundaries. So in our method, we compute gradient from both the spatial and temporal domain of optical flow. Then the regions are selected if they have large gradient in either the temporal or spatial domain in a volume of video.

Formally, for a video volume V with frame number T, the dense optical flow is estimated for every frame noted as OF_t where $t \in \{1, 2, ...T\}$. Then we convolve each OF_t with horizontal and vertical gradient kernels to obtain OF_DX_t and OF_DY_t. The temporal derivative OF_DT_t is computed as the difference between consecutive dense optical flows. Note that the shape of OF, OF_DX, OF_DY and OF_DT are all the same as V. With three derivative flow maps, the active pixel positions are selected using:

$$\begin{aligned} Active\_map = OR\big(&(OF\_DX > TH_{dx}), \\ &(OF\_DY > TH_{dy}), \\ &(OF\_DT > TH_{dt})\big) \end{aligned} \tag{1}$$

where TH_{dx}, TH_{dy} and TH_{dt} are threshold parameters and OR is logical operator OR. The $Active\_map$ is a 3D binary volume. Then object-motion regions are selected by summing $Active\_map$ over the time axis, resulting in a 2D map. Those positions $p(x, y)$ in the 2D map that have values that are larger than $T/2$ are selected as object-motion regions. As a result, the optical flow analysis provides a spatial mask which is used for all T frames in the volume. Finally, a spatial sampling step is added to pick positions between some minimum distance. This prevents large overlapping feature vectors if two sampled positions happen to be too close.

Figure 3 shows an example of the sampling result of a harvesting video. The upper two images are the original frame and the computed optical flow map, and the lower two are the sampled results from IDT [6] and our method. Notice that, in this video, the combine is driving forward and in the upper right optical flow figure, the flows above the hand-drawn red line are caused by the forward motions. These camera motions are not necessarily related to the harvesting category,

but the rotation motions below the red line are distinctly object motions. The bottom left IDT method selects forward motions as features and chooses features from the sky region. Our method only selects the object motion at the bottom region, which is more accurate and therefore will be more informative.

C. Feature extraction

The feature extraction process includes both a feature sampling step and a descriptor generation step. In the system shown in Figure 2, the two branches extract features differently. Branch 1 uses dense sampling with spatial-focused descriptors and Branch 2 uses FPT and temporal-focused descriptors.

Branch 1 is designed for classifying all video clips, including slow motion videos. Thus, its features should focus more on spatial information than temporal motion. We use the dense sampling method and include every position in the spatial frame into the feature vector. As shown in [21], the cuboid method has better performance than other methods to generate spatial-temporal local feature descriptors. Each input volume is first temporally cut into large temporal chunks with length T and each chunk is spatially segmented into a fixed-sized $M \times M \times T$ cuboid. Within each cuboid, different types of features are extracted in smaller blocks, and then concatenated to form a descriptor. The shape of a feature vector per video is fixed and determined by the volume shape only.

Branch 2 is designed only for object motion clips. Its features are extracted more densely in the temporal domain to capture motion information. In this part, the pre-computed object motion mask is applied and features are only extracted around positions in the mask. To describe motion features, a trajectory is better than individual points like STIP, because a motion should be traced as a series of points. But for dashcam videos, the spatial structures of consecutive frames are normally fixed across time. Inspired by optical flow stacking from [15], we extract features at same positions through time which we call a Fixed Position Trajectory (FPT). Unlike trajectories in [6], FPTs have no drifting problems and they can use long temporal windows. The length of FPTs need not be fixed, but here we set them to a fixed length to maintain dimensions for every feature descriptor. The neighborhood region to compute the descriptor is generated with the selected positions at center and lasts T frames. The blocks inside each neighborhood region have fewer frames (a smaller value of t) and this enables the descriptor to preserve more motion information.

IV. Experiments

In this section, we first introduce our videos and the implementation framework. Then in the first experiment, we compare our framework with previous methods. In the second experiment, we analyze feature efficiency for two sampling strategies with different feature sizes.

A. Experiment setup

All videos are collected from dashcams that are mounted on combines and tractors at local farms during the last two years. All the raw captured videos are uniformly sampled into short 5-second (150-frame) clips every 30 seconds. Each short clip is downsampled to 480×272 and only the middle 120 frames are used for feature extraction. These short videos are labelled with three basic farming categories: wheat harvesting, corn chopping and tillage. In total, we randomly select 2400 short clips from all different camera angles with 800 clips for each category. Within each category, 500 are for training and 300 are for testing. The training and testing videos are separated based on the camera angles. For example in Figure 1, the training frames on the left have very different angles and structures from the testing frames on the right.

TABLE I
Feature sampling strategy comparison. Numbers in parentheses are averaged values. Note *Places-CNN is fine tuned with our data.

	Method	Feature per video $N \times D$	Accuracy
Object motion videos	Proposed B1	480×396	0.808
	IDT [6]	$(78773.67) \times 396$	0.822
	Proposed B2	$(1596.17) \times 660$	**0.857**
All videos	*Places-CNN [3]	5×4096	0.696
	Uijlings et al. [1]	2090×594	0.755
	IDT [6]	$(46989.83) \times 396$	0.751
	Proposed B1	480×396	0.792
	B1 + B2	N/A	**0.840**

Note that the focus of this paper is on the feature sampling procedure. Therefore the rest of the experimental design follows that in [20] to process extracted features from all different methods. In the implementation, we use the optical flow from [22], because this method provides accurate dense flow although it is time consuming. We implement the Fisher encoding method based on [23] while the PCA and one-vs-rest SVM classifier are both from the Sklearn package [24]. Since our features have much smaller shape, the PCA process is not performed for our system.

B. Feature sampling comparison

In this experiment, we compare our two-classification framework with other algorithm that have feature sampling strategies, such as IDT [6]. Since the IDT method uses HOG, HOF and MBH, we apply these three features in our strategy for a fair comparison. Recall that the proposed system has two classification branches; Branch 1 is for all video clips and Branch 2 is only for clips which have object motions. In Branch 1, we design $32 \times 32 \times 30$ cuboids and $16 \times 16 \times 10$ blocks inside each cuboid. The cuboid temporal length is set to be one second (30 frames), since motion information is not the focus for Branch 1. Based on this setting, each descriptor has the same feature dimension (396) as the feature from IDT, with 8 bins for HOG and MBH and 9 bins for HOF. In Branch 2, all videos have strong motions; and features are extracted from positions that are computed from the optical flow analysis process shown in Figure 2. Then the neighboring cuboid size is smaller both temporally and spatially. The cuboid size is set to be $16 \times 16 \times 30$ and block size is $8 \times 8 \times 6$. A smaller duration for each block helps to capture more motion information. With this setting, each feature descriptor has a dimension of 660. The fusion stage adds the scores from the two classifiers and selects the most probable category.

Also, an image-based scene classification method Places-CNN [3] is compared. We select frames from both training and

TABLE II
Feature efficiency comparison between different feature designs. Upper four rows shows the dense sampling, lower four rows shows the FPT. Note that the cuboid size is fixed for all designs.

	Block shape $m \times m \times t$	Block number	Scale	Feature per video $N \times D$	HOG (8)	HOF (9)	MBHX (8)	MBHY (8)	HMG (8)	All merged	All Separate
Object motion videos	(8,8,10)	12	1	(2903.48*492)	0.711	0.858	0.599	0.488	0.77	0.819	0.677
	(8,8,6)	20	1	(2903.48*820)	0.718	0.841	0.58	0.305	0.49	0.824	0.635
	(4,4,10)	48	1	(2903.48*1968)	0.665	0.637	0.599	0.258	0.635	0.734	0.763
	(8,8,6)	20	2	(4317.44*820)	0.74	0.853	0.603	0.295	0.717	0.777	0.594
All videos	(16,16,10)	12	1	480*492	0.682	0.7	0.651	0.713	0.707	0.802	0.694
	(16,16,6)	20	1	480*820	0.768	0.698	0.643	0.677	0.485	0.782	0.645
	(8,8,10)	48	1	480*1968	0.618	0.714	0.681	0.688	0.47	0.745	0.766
	(16,16,10)	12	2	572*492	0.654	0.707	0.701	0.695	0.502	0.778	0.736

testing videos at one frame per second (5 images per video). Inspired by [16], we treat the CNN as a feature extractor and fine-tune the top layer. The AlexNet model with pre-trained PLACE dataset is the underlying model. The outputs of the last fully connected layer (FC7) are collected and these 4096-dimensional features are input to a one-vs-rest SVM classifier. In total, 7500 images selected from the trained videos are used to train the classifier and 4500 images are tested.

Two groups of videos are compared for different sampling methods. By optical flow estimation, 478 out of 900 testing clips have significant object motions. So we first compare the performance on 478 videos with large object motions and then compare all 900 videos.

The results are shown in Table I. The first group (upper 3 rows) considers only 478 object-motion videos and we compare our two branches B1 and B2, together with the IDT method. Among all three methods, the dense sampling method (Branch 1) has the worst performance since no feature sampling methods are used. Both IDT and our Branch 2 apply motion-based sampling strategies, and Branch 2 has better accuracy. Apart from accuracy, the feature shapes of each video from the three methods are very different. IDT has the most feature descriptors N, because it extracts more trajectories (see Figure 3) spatially.

The second group (lower 5 rows) considers the entire group of videos, which includes videos with all motion types. Notice that image based method Places-CNN has lower performance than the motion-based methods. Among motion based methods, the accuracy of the IDT method decreases relative to the first test because this method is not accurate for videos that have only small motions or purely camera motions. Our Branch 1 applies a dense extraction method with cuboid, that are uniformly sampled from every position in the video volume. The method in [1] is also densely sampled, but it has much smaller cuboids and blocks, which makes both feature number N and dimension D larger than our design. But considering the performance, our output accuracy is slightly higher, which means denser sampling might not always provide better accuracy. The last row demonstrates that our whole framework outperforms other methods. This final result directly applies results from Branch 1 for videos without object motion, while for object motion videos, it combines the probabilities from two classifiers.

C. Sampled feature efficiency

Sampling methods such as STIP, IDT and the proposed FPT provide interest positions in the video volume, which can either be individual points or a series of points (trajectory). But when generating the feature descriptors from such positions, normally descriptors are built from cuboid regions. One interesting question is how to design the cuboid and block size, in order to get better feature efficiency with different feature sampling methods.

In a video sequence, each cuboid contributes to one feature vector and the whole video is summarized as a feature vector with shape $N \times D$. The dimension of each descriptor D is determined by the number of blocks inside the cuboid, multiplying the number of bins of the feature histogram. The number of the descriptors or cuboids per video N is determined by the sampling strategy. In dense sampling methods, N is normally the ratio between video size and cuboid size, so a smaller cuboid produces a large N. On the other hand, in STIP or trajectory-based methods, the sampling threshold decides N, and it varies based on video content and motions. Such N can be increased by dense cuboid sampling on the same volume or sampling cuboids from more than one spatial scale space. To explore this problem in our scope, we perform video classifications on the farming videos with multiple choices of N and D. Two groups of methods are tested with different videos. Dense sampling is tested for all videos, and FPT is only tested on object-motion clips.

The result is provided in Table II. Notice that the results from Table II are not comparable to Table I, since here we use more feature types than in the previous experiment. In the table, the upper half rows show the test for all videos and the lower four rows are for object motion videos only. We test features individually first with HOG and HOF, together with MBH and the recently proposed HMG [10]. The number after each feature in the table is the number of histogram bins of that feature. The last two columns show two different orders of feature encoding and merging. In *all merging* order, the system merges all different features together as one long feature vector and then uses a Fisher vector to encode. On the other hand, *All separate* encodes different feature types individually and then concatenates all encoded vectors. All designs in the table have the same cuboid shape and each row represents one different design for the block size. *Block number* shows the number of blocks in the fixed-sized cuboid and *scale* means the number

of scale spaces used to extract features. All the processing procedures are the same except the feature extraction methods. The classification accuracy is computed to be the measure of feature efficiency.

The upper half of Table II shows the results of the FPT sampling method. The HOF feature achieves the best performance among all feature types, and even outperforms the results from all features together. The lower half shows the results for the dense sampling method. The first three rows have the same number of feature vectors, but the dimension increases. The best accuracy is achieved with all five feature types with *all merged* order. Also, this design has the minimum number of features and the smallest feature dimension.

There are some observations we can draw from the table. Firstly, neither using denser sampled cuboids nor increasing the blocks inside the cuboid to get larger feature vector improves the accuracy for either the dense sampling method or the FPT method. Also simply applying more types of features does not improve accuracy, unless a late score fusion stage is well trained to merge all decisions, like [20]. However, it is not efficient to use larger features or multiple feature types, because this adds computational burden for the encoder and classifier training process. Secondly, the order of encoding after merging five features (*all merged*) allows the GMM training stage to choose the most representative centroids in the high dimensional feature space. Therefore it improves the encoded feature efficiency to some degree. In contrast, when merging individual encoding features together *all separate*, all feature types have the same contribution to the final feature vector. In this case, the final decision could be influenced by ineffective features. Thirdly, sampling methods such as FPT perform a spatial selection process before feature extraction, and the gradient-based features such as MBH or HMG might not provide as good performance as features that operate on the original image or optical flow, such as HOG and HOF.

V. Conclusion

In this paper, we propose a two-branch framework to separate videos into different farming activities with a limited collection of data. These videos have very different motion characteristics, and we apply two branches and treat videos separately based on their motions. In our system, low motion videos and pure camera motion videos are considered as non-object motion videos, and object motion videos refer to those whose motions are independent of camera movement. Two different sampling strategies, dense sampling and FPT, are performed individually on the two types of videos. The results show that our method uses fewer feature vectors from the videos and achieves better classification accuracy.

This framework is designed to solve the problem of classifying videos with different motion conditions. Not only farming videos, but also hand-held or first-person cameras introduces a large amount of video motions mixed with camera movements and object motions. Our proposed two-branch system adapts to such types of videos and provides more efficient features for further analysis.

References

[1] J. Uijlings *et al.*, "Video classification with densely extracted HOG/HOF/MBH features: an evaluation of the accuracy/computational efficiency trade-off," *International Journal of Multimedia Information Retrieval*, vol. 4, no. 1, pp. 33–44, 2015.

[2] P. Dollár *et al.*, "Behavior recognition via sparse spatio-temporal features," in *2005-2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2005, pp. 65–72.

[3] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[4] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 3169–3176.

[5] J. Liu *et al.*, "Recognizing realistic actions from videos in the wild," in *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1996–2003.

[6] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *2013 IEEE International Conference on Computer Vision (ICCV)*, pp. 3551–3558.

[7] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. 886–893.

[8] I. Laptev *et al.*, "Learning realistic human actions from movies," in *2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008, pp. 1–8.

[9] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *European Conference on Computer Vision (ECCV)*, 2006, pp. 428–441.

[10] I. C. Duta *et al.*, "Histograms of motion gradients for real-time video classification," in *14th International Workshop on Content-Based Multimedia Indexing (CBMI)*, pp. 1–6.

[11] L. Shao *et al.*, "Spatio-temporal Laplacian pyramid coding for action recognition," *IEEE Transactions on Cybernetics*, vol. 44, no. 6, pp. 817–827, 2014.

[12] J. C. Niebles *et al.*, "Modeling temporal structure of decomposable motion segments for activity classification," in *2010 European Conference on Computer Vision (ECCV)*, pp. 392–405.

[13] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the Fisher kernel for large-scale image classification," in *2010 European Conference on Computer Vision (ECCV)*, pp. 143–156.

[14] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms," http://www.vlfeat.org/, 2008.

[15] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 568–576.

[16] A. Karpathy *et al.*, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 1725–1732.

[17] K. Soomro *et al.*, "UCF101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

[18] H. Kuehne *et al.*, "HMDB: a large video database for human motion recognition," in *2011 IEEE International Conference on Computer Vision (ICCV)*, pp. 2556–2563.

[19] J. Xiao *et al.*, "SUN database: Large-scale scene recognition from abbey to zoo," in *2010 IEEE conference on Computer vision and pattern recognition (CVPR)*, pp. 3485–3492.

[20] X. Peng *et al.*, "Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice," *Computer Vision and Image Understanding*, vol. 150, pp. 109–125, 2016.

[21] H. Wang *et al.*, "Evaluation of local spatio-temporal features for action recognition," in *2009-British Machine Vision Conference (BMVC)*. BMVA Press, 2009, pp. 124–1.

[22] C. Liu, "Beyond pixels: exploring new representations and applications for motion analysis," Ph.D. dissertation, Massachusetts Institute of Technology, 2009.

[23] J. Krapac, J. Verbeek, and F. Jurie, "Modeling spatial layout with Fisher vectors for image categorization," in *2011 IEEE International Conference on Computer Vision (ICCV)*, 2011, pp. 1487–1494.

[24] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.