CS229: Action Recognition in Tennis

Aman Sikka
Stanford University
Stanford, CA – 94305
asikka@stanford.edu

Rajbir Kataria
Stanford University
Stanford, CA – 94305
rkataria@stanford.edu

1. Motivation

As active tennis players we are always trying to improve our game by observing the professionals. By observing their forehand, backhand and serve techniques, one can easily bring changes and perfect their own form. Unfortunately videos are not annotated with info stating the type of action the player is performing. Our goal for this project is to recognize the action a player is performing, so searching for such an action is plausible.

2. Introduction

This report discusses the findings of our project, Action Recognition in Tennis. It discusses our dataset, key-point detector and descriptor, the experiments we performed to determine the sampling mechanism and our overall results.

3. Dataset

Our dataset for this project consisted of video clips from 3 match videos. We collected 200 videos from the first match, and 100 videos each from the second and third matches and limited the videos to right-handed male tennis players. Due to great variation of camera angles, we only looked at clips shot from a bird's eye view as shown in Figure 1.

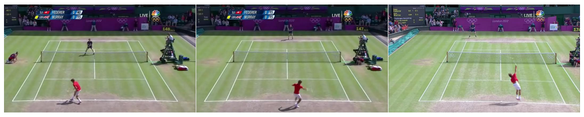


Figure 1: Backhand, Forehand and Serve

Since majority of the tennis match is shot from the bird's eye view angle, it provided us with sufficient data for our learning algorithm. We manually noted the start and end of action frames and extracted the clips from the match. This resulted in isolating the actions that were of interest to us, i.e. backhands, forehands, and serves. A video annotation tool (vatic – Video Annotation Tool from Irvine, California) was used to create bounding boxes around the tennis players that were performing the action of interest. We manually drew bounding boxes around the tennis players for each of the clips. Figure 2 below shows the bounding box we drew on each frame.

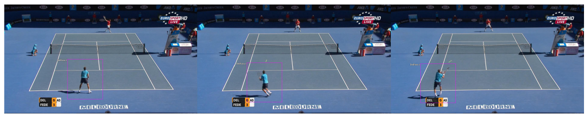


Figure 2: *Bounding Box around Player of interest*

Extracting the region of interest enclosed within the bounding box removed most of the background and ensured that we weren't capturing any other actions, i.e. actions of the umpire.

Figure 3 below shows our final processed data (in the form of video clips) that was fed into our algorithm to perform action recognition.

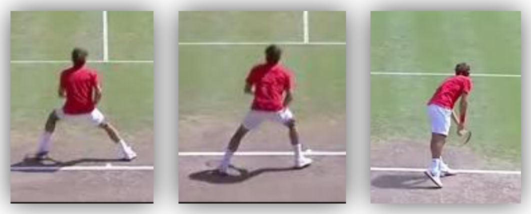


Figure 3: *Extracted region of interest*

Due to limited amount of time, our pipeline initially used one dataset. We tried to generalize it with more datasets near the end of the project, but realized our third dataset had erroneous data. As shown in Figure 4, quite a few clips contained part of the score. Due to this we required a larger dataset so to avoid the score from becoming an important key-point.

All our analysis excluded the third dataset.

Figure 4: *Example of Noisy Dataset*

4. Algorithm

The first step in our algorithm involved detecting and describing important key-points in a clip. We used Dense Sampling as key-point detector and HOG3D as a descriptor.

4.1 Dense Sampling

Dense sampling extracts video blocks at regular positions and scales in space and time. There are 5 dimensions to sample from: (x, y, t, σ, τ), where σ and τ are the spatial and temporal scale, respectively. In this project, the minimum size of a 3D patch is 8×8 pixels and 6 frames. Spatial and temporal sampling is done with 50% overlap. These numbers correspond to the one resulting best result on UCF50 dataset.

Once the key-points are selected, they must be described. The standard describer (or rather descriptor) for videos is HOG3D.

4.2 HOG3D Descriptor

For a given spatio-temporal block, HOG3D splits it into M x M x N sub-regions (M for spatial and N for temporal splits). Within each region, this method counts how many 3D gradients are roughly aligned with each of the directions of a polyhedral structure. This computation is done efficiently using basic geometric operations. In [3] the authors claim that, for BoW, the best performance in the validation set of the UCF50 dataset was obtained with an icosahedron (i.e., 20 orientations), M = 4 and N = 3, giving a total of 960 dimensions. These numbers were used for our testing as well. The figure below gives a quick overview of the descriptor computation.

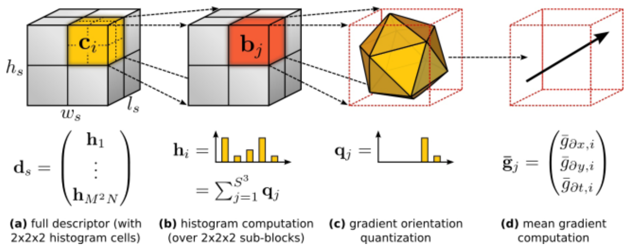

$\mathbf{c}_i$
$\mathbf{b}_j$
h_s
w_s
l_s
$\mathbf{d}_s = \begin{pmatrix} \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{M^2N} \end{pmatrix}$
$\mathbf{h}_i =$
$= \sum_{j=1}^{S^3} \mathbf{q}_j$
$\mathbf{q}_j =$
$\bar{\mathbf{g}}_j = \begin{pmatrix} \bar{g}_{\partial x,i} \\ \bar{g}_{\partial y,i} \\ \bar{g}_{\partial t,i} \end{pmatrix}$
(a) full descriptor (with 2x2x2 histogram cells)
(b) histogram computation (over 2x2x2 sub-blocks)
(c) gradient orientation quantization
(d) mean gradient computation

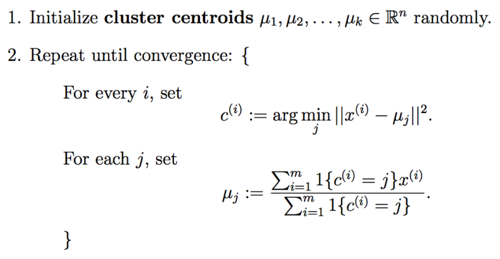
Figure 5: *HOG3D overview*

Once the points are described, a dictionary of visual code words is created using the k-means clustering algorithm.

4.3 K-means

K-means is a method of cluster analysis that aims to partition 'n' observations into 'k' clusters in which each observation belongs to the cluster with the nearest mean.

The k-means algorithm is described in the Figure 6 below.



```
1. Initialize cluster centroids $\mu_1, \mu_2, \ldots, \mu_k \in \mathbb{R}^n$ randomly.
2. Repeat until convergence: {
       For every $i$, set
           $c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2.$
       For each $j$, set
           $\mu_j := \frac{\sum_{i=1}^m 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)} = j\}}.$
   }
```

Figure 6: *K-means Clustering Algorithm*

Through research [15], we determined the optimal number of clusters to be between 1000-4000. We experimented with a range of values between 1000-4000, but they didn't seem to have much if any effect on our results. If the number of clusters picked is too small, then the visual words in our dictionary may not be representative of all the patches. However if the number of clusters is too large, then we could be over-fitting. Since the number of clusters didn't seem to have any effect, we picked the number to be 1000 in decrease computation time.

Once our dictionary was created, we employed the bag-of-words model to generate a histogram representation of our videos.

4.4 Bag-of-Words

The bag-of-words model, in the field of computer vision, can be applied to image classification, by treating image features as words. BoW finds its roots in document classification, where BoW is a sparse vector of occurrence counts of words; that is, a sparse histogram over the vocabulary. In computer vision, a bag of visual words is a sparse vector of occurrence counts of a vocabulary of local image features. [2]

The code-words obtained after k-means a mentioned in section 4.3 provided us with the vocabulary of local video features. Figure 7 summarizes our entire process including BoW approach when applied to images and is similar to video data.

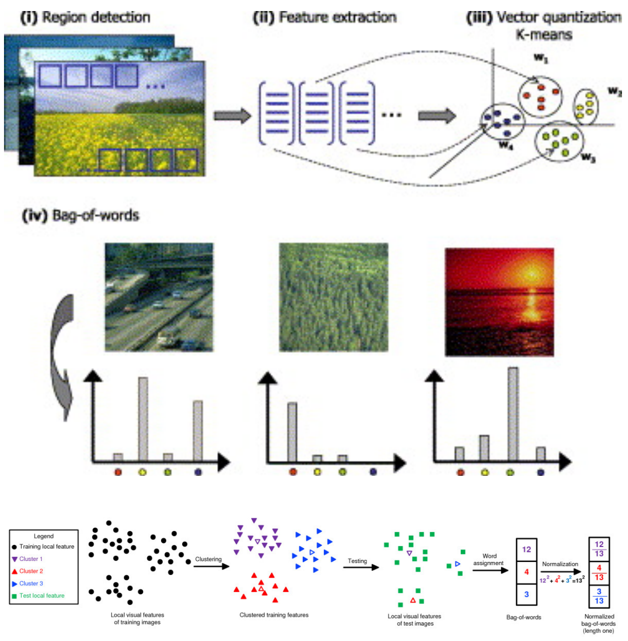


Figure 7: *Illustration of Bag-of-Words Model [17]*

4.5 Classification

After generating the histograms for each action clip we were ready to classify the data as Forehand, Backhand or Serve. For classification we focused on Support Vector Machines (SVMs). Based on our research [3][7]], SVMs are the most commonly used classifier used in recognition-based tasks. Our data could fall into one of three classes, either a forehand, backhand, or serve, so we tried a 1-vs-all approach. This was somewhat effective, but didn't give us information on the misclassifications and required three different models. The models required one of the actions as positive data, and the remaining actions as negative data. Once we discovered a multi-class classifier, we abandoned the 1-vs-all approach in favor of the multi-class.

We explore two kernels, linear and RBF in our SVM model

The Radial Basis Function (RBF) is usually represented by a Gaussian function. As a result it maps the data to an infinite dimensional space. The function depends on the euclidean distance between x and z where either x or z is the support vector, and the other is the test point. It is controlled by a the σ parameter. A larger value of σ indicates that the support vector has strong influence over a larger area, and hence gives a smoother decision boundary. The figure below gives an equation of the Radial Basis Function Kernel [16].

$$k(\boldsymbol{x}, \boldsymbol{z}) = \exp\left(\frac{-\|\boldsymbol{x} - \boldsymbol{z}\|^2}{c}\right)$$

Figure 8: Equation of the RBF Kernel

5. Results

The linear kernel was used as a baseline and worked quite well, but the RBF kernel outperformed the linear kernel for both of our datasets. Figure 9 and 10 show how accuracy is affected with the change of 'cost' and 'gamma' parameters for linear and RBF kernel respectively.

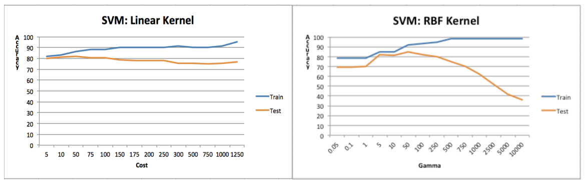


Figure 9: *Accuracy comparison between Linear and RBF kernel for Dataset 1*

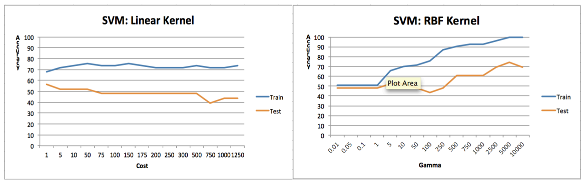


Figure 10: *Accuracy comparison between Linear and RBF kernel for Dataset 2*

For both of our datasets, the best result given by the RBF kernel is at least 10% better than the linear kernel. The confusion matrix shown in Table 1 outlines the test classification errors for each class for dataset 1.

		Predicted		
		Forehand	**Backhand**	**Serve**
Actual	**Forehand**	66.66%	0%	33.33%
	Backhand	0%	100%	0%
	Serve	0%	0%	100%

Table 1: Confusion Matrix using RBF Kernel (Dataset 1)

We performed a 70%-30% split between training and testing examples, and each action used the same number of examples for training.

6. Observations and Conclusions

The process of collecting the dataset (identifying the frames containing the action, annotating the bounding box and generating a dictionary) was very a time consuming task due to which we were limited to the amount of data we could collect. We collected 200 videos for the first dataset, and 100 videos each for the second and third datasets. Some of our interesting observations are noted below.

1. Actions learnt on player A from one set of match videos do not generalize well for the same player in a different match. This is probably due to slight variations in camera angle. The accuracy dropped from over 85% to 60%. This implies that multiple clips from different matches are required to better generalize the actions.
2. The accuracy was 50% when trained on men matches and tested on woman. I would suspect this to be the case due to the fact that woman play at a slower pace.
3. One of the training sets had large illumination variation, since the match was shot on a cloudy day, there were strong shadows casted throughout the game. For such situation it makes sense to do some kind of background removal.
4. Since both the HOG3D descriptor and the BoW representations are based on histograms, one of the most appropriate kernels is the RBF. Due to a high accuracy of our results, the RBF kernel did not make any difference. On testing on the minimal data available from the other two matches, RBF did seem to perform around 5-10% better than linear kernel. More data would be required to confirm the results.

7. References

[1] Video Annotation Tool from Irvine California, VATIC, http://mit.edu/vondrick/vatic/

[2] Bag of Words http://en.wikipedia.org/wiki/Bag_of_words_model_in_computer_vision

[3] A.Kl¨aser, M. Marszałek, and C. Schmid. A spatio-temporal descriptor based on 3D-gradients. In British Machine Vision Conference, pages 995–1004, sep 2008.

[4] L. Gorelick, M. Blank, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(12):2247–2253, December2007.

[5] I. Laptev and T. Lindeberg. Space-time interest points. In ICCV, 2003.

[6] D. G. Lowe. Distinctive image features from scale-invariant keypoints. Int Journal of Computer Vision, January 2004

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In Proc IEEE Conf on Computer Vision and Pattern Recognition, San Diego CA, June 20-25, 2005.

[8] H. Uemura, S. Ishikawa, and K. Mikolajczyk. Feature tracking and motion compensation for action recognition. In British Machine Vision Conference, 2008.

[9] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In Proc of the ACM Multimedia, Augsburg, Germany, September 23–28 2007.

[10] H. Wang, M. M. Ullah, A. Kl¨aser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In Proc 20th British Machine Vision Conf, London,Sept 7-10, 2009.

[11] A. A. Efros, A. C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In Proc 9th Int Conf on Computer Vision, Nice, France, Oct 13-16, pages 726–733, 2003.

[12] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge (VOC) Results. http://www.pascalnetwork.org/challenges/VOC/voc2009/workshop/, 2009.

[13] C. Harris and M.J. Stephens. A combined corner and edge detector. In Alvey Vision Conference, 1988.

[14] G. Willems, T. Tuytelaars, and L. Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In ECCV, 2008

[15] k-means clustering, http://en.wikipedia.org/wiki/K-means_clustering

[16] Using a Radial Basis Function as a kernel, http://svr-www.eng.cam.ac.uk/~kkc21/thesis_main/node31.html

[17] Image and Vision Computing, http://www.sciencedirect.com/science/article/pii/S0262885606002253