# Summarization of sports videos based on unsupervised deep learning[*]

Sandeep Sripada
ssandeep@stanford.edu

Venu Gopal Kasturi
venuk@stanford.edu

Gautam Kumar Parai
gkparai@stanford.edu

## ABSTRACT

Analyzing videos based on hand crafted features is a tedious task and is highly dependent on the type of data. These methods cannot be easily extended to account for other sensor modalities like audio, texts. The aim of the project is to be able to use both video and audio in efficiently and effectively recognizing events of interest in sports videos. Using unsupervised deep learning we learn the features from the data itself and try to identify a set of predefined classes in sports videos (Goal, Penalty, Foul, FreeKick, Corner). We built a baseline system using hand crafted vision features like HOG3D and compared it against the peformance of the unsupervised learning method (2-3 hidden layered ISA). The accuracy for multi-class recognition was 69.4% in the baseline system and 58.4% in the unsupervised method. We also summarized new videos using the model learnt to generate highlights.

## General Terms

Video analysis, Summarization, HOG3D, Unsupervised Deep Learning

## 1. INTRODUCTION

One of the most common and popular procedures for feature description in visual recognition is to use hand-designed methods like SIFT [7], [8], HOG [2],GLOH [9] and SURF [4]. A major disadvantage with these methods is that they are not easy to extend to other modalities like audio, text. This is where unsupervised feature learning methods like Deep Belief Nets [3], Stacked Autoencoders [11], Sparse Coding [10] take precedence as they are more generic because of the way they learn features directly from the provided data.

In this project we used an unsupervised feature learning method which is based on the use of Independent Subspace Analysis (ISA) and also incorporates two important ideas in convolutional neural networks [6] i.e. convolution and stacking. ISA learns features that are robust to local translational changes while being selective to frequency, rotational and velocity changes. However, this could be very slow but by incorporating ideas mentioned above, the training phase could be made faster[1]. Features are learnt with small input patches and then convolved with a larger region of the input data and then used as inputs to the next layer.

We follow the work in [5] and based on their finding that there is no universally best feature for all datasets and that dense sampling with HOG/HOF works well, we implement a system with HOG3D and dense sampling. We compare this to the ISA algorithm and measure the accuracy.

The paper is organized as follows: section 2 describes the data accumulation and annotation procedure, section 3 describes the implementation details of the two systems developed, section 4 talks about the results obtained and goes through the analysis of the results and finally section 5 mentions possible future work.

## 2. DATA DESCRIPTION

We have downloaded videos from Youtube[2] and annotated them manually for the following classes.

- Goal
- Foul
- Freekick
- Penalty
- Corner

We annotated a total of 725 clips from ∼80 Youtube videos of soccer matches. These clips were 5 seconds long on an average and had both crowd and commentator audio. Howvever, not all videos were in English (it was a multilingual dataset including chinese, spanish and arabic commentary). The videos were around 25-30 fps and had a fixed resolution of 320x240. All the videos were initially downloaded from Youtube in ‘flv’ format and were later converted to ‘avi’ and resized using ffmpeg[3]. We then extracted audio from each

[*] CS 229: Final Project

[1] This algorithm was proposed by the Stanford AI team and is currently under review at CVPR.

[2] www.youtube.com

[3] http://www.ffmpeg.org/

**Table 1: Distribution of clips**

| Class | Number |
|---|---|
| Attempt on goal | 13 |
| Corner | 40 |
| Foul | 45 |
| FreeKick | 114 |
| Goal | 261 |
| None | 95 |
| Penalty | 157 |
| Total | 725 |

of these videos using jAudio[4] (we modified jAudio to handle dynamic generation of MFCC co-efficients). We automated this entire process and obtained videos for annotation. The distribution of final annotated clips is as shown in Table 1.

We also downloaded 2 full matches from the same source to try summarization/highlight generation. The full matches were again resized to 320x240 and were 2 hours 44 minutes in overall playing time.

# 3. IMPLEMENTATIONS

This section describes both the systems that we used in detail. System 1 uses work in [1] and based on the flow as mentioned in [5]. The second system is the ISA as designed by the Stanford AI team.

## 3.1 HOG3D+Dense sampling

This system uses HOG3D based on [1] and can be seen as an extension to SIFT [7]. This descriptor combines both shape and motion information at the same time. A 3D patch is divided into $n_x$x$n_y$x$n_t$ cells. The corresponding descriptor concatenates gradient histograms of all cells and is then normalized. We used the executable from the authors' website[5] and use the following parameter values while sampling points for clustering:

- XY stride: 20
- XY max scale: 2
- T stride: 10
- T max scale: 2
- Scale overap: 1 (side-by-side)
- T cells: 2
- XY cells: 2

The outline of the algorithm is as follows:

1. Use HOG3D to represent patches in the video
2. Use KMeans to cluster the patches and create the codebook
3. Use histogram based representation for video
4. Use MFCC to represent audio
5. Histogram + MFCC for audio to train the classifier

Using the HOG3D settings mentioned above, we generate samples from all of our training samples and pass them to step 2. Using the kmeanslite program, we obtained 4000 clusters after running it for 100 iterations. These clusters act as codebook and we examined all the training samples and then extracted overlapping patches from the segments (based on our annotations) and calculated the histograms. For each patch, we calculated the closest centroid and incremented the count for that centroid and used the final count for all centroids as our feature representation for that clip. This process was repeated for all our annotations and at the end we obtained a feature vector for all the annotated clips.

For computing the MFCCs we used the created mp3 files to compute the Magnitude Spectrum (MS) and used the obtained values to compute the MFCCs. We obtained various number of MFCC coefficients (25, 40, 100, 1000, 3000, 4000). We concatenated the obtained MFCC co-efficients to the generated video representation and used it as an instance to train a SVm classifier with linear and polynomial kernels. The results for this algorithm are shown in section 4.

## 3.2 Stacked convolutional ISA

The system consists of layers of PCA (for whitening) and ISA. ISA is an unsupervised learning algorithm, it is a two-layered network with a square nonlinearity in the first layer and square-root nonlinearity in the second layer as shown in 2. We used the system as provided by Quoc Le and Will Zou after changing the parameters a bit. We ran a 3-layer network initially on $(1/3)^{rd}$ of the total data and then ran a 2-layer network on the entire data. We used the default block sizes of 16x16x10, 20x20x14, 20x20x18 for the first, second and third layers respectively. We used a SVM + $\chi^2$ kernel to learn the model. The results are as presented in section 4.

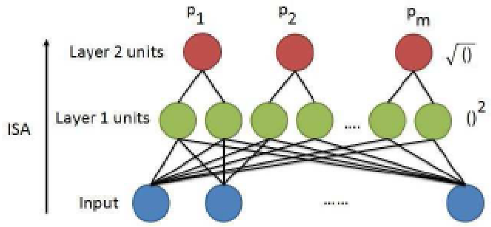
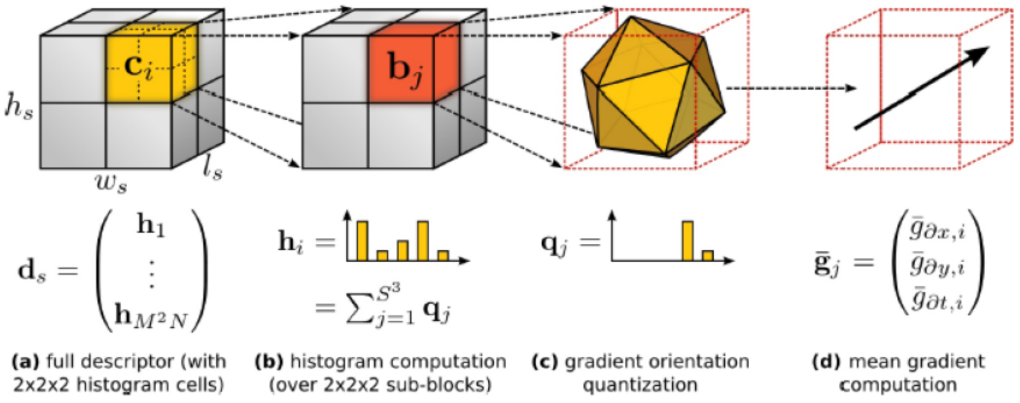


**Figure 2: ISA network**

To get a better understanding of unsupervised deep learning feature methods we implemented the Sparse Encoder on a set of images to get an idea of the kinds of features learnt. A sparse encoder is a neural network which learns the identity function. The main principle behind this architecture is to map the input representation to a high level representation which can be linearly separable after the first level. It helps

[4] http://sourceforge.net/projects/jaudio/
[5] http://lear.inrialpes.fr/people/klaeser/


$h_s$
$\mathbf{c}_i$
$w_s$
$l_s$
$\mathbf{b}_j$
$\mathbf{d}_s = \begin{pmatrix} \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_{M^2N} \end{pmatrix}$
$\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 1: HOG3D computation(from [1])**

**Table 3: Results on system 1 with changing audio MFCCs (on Interesting vs None)**

| Setup | Kernel | Accuracy |
|---|---|---|
| 45 co-effs | SVM + linear | 85.5% |
| 100 | SVM + linear | 88% |
| 500 | SVM + linear | 90.2% |

**Table 4: Results on V,A,V+A on system 1 (3000 coefficients)**

| Setup | Kernel | Accuracy |
|---|---|---|
| V | SVM + linear | 73.6% |
| A | SVM + linear | 52.2% |
| V+A | SVM + linear | 75% |

**Table 5: Results on system 2 (#layer in brackets)**

| Setup | mode | Kernel | Accuracy |
|---|---|---|---|
| Partial data (3) | Multi class | SVM+linear | 53.8% |
| Full data (3) | Multi class | SVM+linear | 58.4% |
| Full data (2) | Multi class | SVM+linear | 20% |
| Full data (2) | Multi class | SVM+$\chi^2$ | 23.05% |
| Full data (2) | Interesting vs None | SVM+linear | 34.33% |
| Full data (2) | Interesting vs None | SVM+$\chi^2$ | 39.56% |

in relieving the user from designing complex hand-crafted features every time a new sensor is incorporated.

## 4. RESULTS & ANALYSIS

The results for various experiments on system 1 are as shown in Table 2. As the quality of the video increases, the performance improved as more information is available for a given clip. The resolution of 3gp video samples was as low as 96x68 at times and was mostly at around 192x168. We used the interesting vs none to generate a summary of the downloaded full matches and the clips were not that accurate. This is mostly because of the number of none's marked, increasing that would definitely give a better model.

In Table 4, we see that as the number of audio co-efficients increases, the performance also increases. This is because the amount of information from audio is high and since audio is a good differentiator. We also performed experiments on how the system performed with video alone, audio alone and video+audio combined. The results of that experiment are in Table **??**. We see that V+A performs better as expected but only slightly. This could be attributed to the fact that the marked classes all have the commentator and crowd voices (cause all the marked classes were highlights/interesting events) and hence the difference was not that high. However, including none and then performing a none vs rest would definitely show the advantage of using audio.

S3ISA (2 layer n/w)[Params: 300 dimension 1st layer 200 dimension 2nd layer 3000 centroids] AVI - Multi-Class Chi-Sq - 20.03AVI - Interesting V/S None Chi-Sq - 35.56

The results from the experiments on system 2 (using ISA(3)) are as shown in Table 5. The training for the first case was done on $(1/3)^{rd}$ data and the results are quite encouraging to see that small amounts of data used for feature learning could match system 1. For goal versus rest, the accuracy matched that of system 1 at 86%. However for the other classes, the results are not as promising mainly because of the small number of annotations.

## 5. FUTURE WORK

We would like to include audio in the same system as well i.e. use audio as raw data in the same pipeline and test the accuracy of the entire system. Some of the possible ways to represent audio could be MFCC itself as input or Spectrograms as input to the first layer. Since the framework is generic and would learn the features from raw data, we would like to include text as well into the mix and see how that information could be used in improving the accuracy.

One of the reasons for the bad performance could be related to the parameters used, one task could be to tweak

Table 2: Results on system 1 (HOG3D and Dense)

| Setup | Kernel | Mode | Accuracy |
|---|---|---|---|
| AVI + audio (4000) | SVM + linear | Multi-class | 69.4% |
| AVI + audio (4000) | SVM + linear | Interesting vs None | 91% |
| AVI + audio (4000) | SVM (one class) | Interesting vs None | 93% (n=0.01, C=6.1e-05, $\gamma$=1) |
| 3gp + audio (4000) | SVM + linear | Interesting vs None | 83.5% (C=0.0039, $\gamma$=1.0) |

the params and use the optimal values for further analysis. We also ran into a lot of contraints with respect to memory and system disk space availability; with the current understanding we can minimize such failures and run the system for various param values.

## 6. ACKNOWLEDGEMENT

We would like to thank Quoc Le and Will Zou for their guidance and suggestions throughout the entire project.

## 7. REFERENCES

[1] M. M. A. Klaser and C. Schmid. A spatio-temporal descriptor based on 3d gradients. *BMVC*, 2008.

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *CVPR*, 2005.

[3] S. O. G. Hinton and Y. Teh. A fast learning algorithms for deep belief nets. *Neural Comp*, 2006.

[4] T. T. H. Bay, A. Ess and L. V. Gool. Surf: Speeded up robust features. *CVIU*, 2008.

[5] A. K. I. L. H. Wang, M. M. Ullah and C. Schmid. Evaluation of local spatio-temporal features for action recognition. *BMVC*, 2010.

[6] Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time- series. *The Handbook of Brain Theory and Neural Networks*, 1995.

[7] D. Lowe. Object recognition from local scale-invariant features. *ICCV*, 1999.

[8] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.

[9] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *PAMI*, pages 1615–1630, 2005.

[10] B. Olshausen and D. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 1996.

[11] D. P. Y. Bengio, P. Lamblin and H. Larochelle. Greedy layerwise training of deep networks. *NIPS*, 2007.