Login details:

ABACUS:
username: vijay@abacus.iiit.ac.in
password:code@abacus
directory: /lustre/cvit/vijay/Himangi

ATOM:
username:himangi.s@10.2.16.16
password:cvit123
directory: /Pulsar3/himangi.s/Soccer

1. CreateDataSet
    - We wish to use the intermediate zoom level and segment the videos to obtain clips of that zoom level with relevant actions.
    - Watch a video and manually annotate the videos by noting down the action type, start timestamp and duration(in seconds).
    - Create a file in space separated format one per line: <type_of clip start timestamp clip_time_in_seconds> eg: /lustre/cvit/vijay/Himangi/DPMResults/Spain_Netherlands/part4_out_time.txt
    - At present, we have 9 categories :
        i. 0 -> walk
        ii. 1 -> kick
        iii. 2 -> run
        iv. 3 -> fall
        v. 4 -> dribble
        vi. 5 -> goalkeeper
        vii. 6 -> hand_kick
        viii. 7 -> throw(on a foul)
        ix. 8 -> stand
    - Currently the output folder is : /lustre/cvit/vijay/Himangi/DPMResults/Actions
    - Make sure to change the vidno variable in line 12 to one plus the last created video number to avoid overwiting the dataset.
    - The script is: /lustre/cvit/vijay/Himangi/CreateDataSet/demo.m. We also have a README here.
    - Call the script with the full path of the video and the timestamps file corresponding to it. Eg: demo('/lustre/cvit/vijay/Himangi/DPMResults/Spain_Netherlands/part4_out.avi','/lustre/cvit/vijay/Himangi/DPMResults/Spain_Netherlands/part4_out_time.txt')
    - The dataset is created as a set of videos categorized into folders based on the label in the text files.

[Note: The Actions folder created above was transferred to the ATOM and all further steps were performed there. This is because we did not have the correct versions and paths of ffmpeg and opencv as required by the dense_trajectories library in ABACUS.]

2. Extract the dense trajectory features for all the videos.
    - We use the library: http://lear.inrialpes.fr/people/wang/dense_trajectories
    - The script is /Pulsar3/himangi.s/Soccer/dense_trajectory_release_v1.2/demo.sh.
    - Change START and END variable names to the current set of video number boundaries.
    - After extracting the dense trajectory features, we have a file called featuresN (N is

the video number) in the same path as the original video folder. Eg: /Pulsar3/himangi.s/Soccer/Actions/fall/features108

- PS: This runs only on node2 or node3 or node4. I wasnt able to qsub(submit remotely) the script, but it ran fine in a live qrsh session on the node by “sh demo.sh”.

3. Now we use these features to create a bag of words model by kmeans clustering.
    - Since there are a huge number of features per video, we are selecting just about 3750 features randomly from each of the extracted features.
    - These features from all the video features are then saved in “feas.mat”.
    - We cluster them using the kmeans clustering with 20000 centers and thus have 20000 words in the bag of words model.
    - This vocabulary thus built is saved in “vocab.mat”.
    - This should be clear from the script: /Pulsar3/himangi.s/Soccer/dense_trajectory_release_v1.2/build_vocab.m
    - To Run it : change the start_vidno, end_vidno and numFeasPerImg = 1050000/num_videos in the script. Then call build_vocab from matlab command line.
4. Next, we use the bag of words model to transform the feature space before training the SVM:
    - load the “vocab.mat” created in the previous step.
    - We are 'sort of' finding the nearest neighbor of each of the video features among the words in “vocab.mat” and get a histogram of the words of size numWords as the transformed feature vector.
    - The label is extracted from the folder name and we store the label along with the transformed features.
    - This is saved in “FinalFeas.mat”.
    - The script is : /Pulsar3/himangi.s/Soccer/dense_trajectory_release_v1.2/buildHistFeas.m
    - To Run it : change the start_vidno, end_vidno in the script. Then call build_vocab from matlab command line.
5. Then we train an SVM based on the features and labels saved in step 3.
    - The SVM library used here is the liblinear library: http://www.csie.ntu.edu.tw/~cjlin/liblinear/
    - The script used : /Pulsar3/himangi.s/Soccer/dense_trajectory_release_v1.2/train_svm.m
    - The model is saved in “svm_model.mat”.
6. Testing the SVM.
    - First segment the video to the zoom level using CreateDataSet as in step2 to get a test dataset as ActionsTest.
    - Transfer that to ATOM: /Pulsar3/himangi.s/Soccer/ActionsTest
    - Now extract dense trajectories for the test set using : /Pulsar3/himangi.s/Soccer/dense_trajectory_release_v1.2/demoTest.sh. You will have to set the START and END variables according to the video test dataset. (Only difference from the demo.sh in step 2 is the directoryname :p)
    - Now, transform the feature vectors of the test space using the bag of words model of step 3 using: /Pulsar3/himangi.s/Soccer/dense_trajectory_release_v1.2/buildHistFeasTest.m. The features are now saved as FinalFeasTest.mat
    - Then use the SVM model to predict the labels for the test samples and find the accuracy of the model using:

/Pulsar3/himangi.s/Soccer/dense_trajectory_release_v1.2/test_svm.m

Things tried and that can still be used

[All ON ABACUS]

1. Preprocess the video for human detection using DPM(Deformable Parts model)
   - Paper: http://cs.brown.edu/~pff/papers/lsvm-pami.pdf (Although we use it as a blackbox)
   - Code: https://github.com/rbgirshick/voc-dpm
   - Create a dataset by dividing the soccer videos with some common team into parts of 10 minutes each. Those parts with significant players were selected for human detection as they are more relevant. The DPM algorithm was run on 16 10-minute chunks. It took about a day and a half for it to finish for threshold value of -1.
   - The script to run : /lustre/cvit/vijay/Himangi/voc-dpm/demo.m
   - Call the script with the videoname and the output folder. Eg: demo('/lustre/cvit/vijay/DATA/SOCCER/Spain_Netherlands/part1.mp4', '/lustre/cvit/vijay/Himangi/DPMResults/Spain_Netherlands')
   - A file name <videoname>.txt with the frame_number , bounding box specifications(top left, height, width) and confidence values is generated in the outputfolder. Eg: /lustre/cvit/vijay/Himangi/DPMResults/Spain_Netherlands/part1.txt
   - Now visualize the DPM output as a video by annotating the video using the file generated on running demo. The script is: /lustre/cvit/vijay/Himangi/voc-dpm/writeToFile.m. Call writeToFile with the input_videoname, the file generated by demo.m and an outputvideo_name. Eg: writeToFile('/lustre/cvit/vijay/DATA/SOCCER/Spain_Netherlands/part1.mp4','/lustre/cvit/vijay/Himangi/DPMResults/Spain_Netherlands/part1.txt', '/lustre/cvit/vijay/Himangi/DPMResults/Spain_Netherlands/part1_out.avi')
   - We can view the video outputvideo_name to see the DPM results.
2. Annotate the video manually to get bounding boxes of the actions.
   - The idea was to set bounding boxes of the actions universally for an entire clip.
   - The DPM results gave bounding boxes of the humans, but they were not uniform across the clip.
   - We get dense trajectory features each associated with a specific coordinate and one way to increase the training efficiency and hence the testing accuracy would be to use the dense trajectories from only thouse coordinate regions where the action was being performed.
   - Idea : Select 10 random frames from the clip. Manually annotate the frame by marking 2 points(top left and bottom right). Then set the bounding box for the frame as the rectangle enclosing all the 10 boxes.
   - Script providing the interface:

/lustre/cvit/vijay/Himangi/DPMResults/Actions/find_x_y.m

These two were attempts not fully incorporated but could be enhanced and included. A better way out is to use a tracking algorithm to get human tracks and then manually label tracks with Actions.