Computer Vision Report

UP776193

Abstract—This report details the process and results of experimentations involving detecting human movements activities in video. Through the use of Motion History Images, Feature Detectors and Descriptor Extractors, a classifier is trained to recognise different actions. The trained classifier is then tested to measure its performance. This project will analyse the performance differences between different components used to fulfil this task.

1 Introduction

This project consists of experimenting with different components that are used to detect and identify human movement in video. Coupled with this report is a software solution used to conduct these experiments. This solution was developed from a basic software given in lab tutorials for the Computer Vision unit at the University of Portsmouth. The basic functionality was built on and improved in order to be able to include different detectors, extractors, and classifiers for experimentation. Following this, experiments were run on two distinct datasets to determine components performance compared to each other. This report documents the process of developing the software solution, the results gathered from running experiments and a discussion into the analysis of the results.

2 Details of the Approach

This section details the process of developing the software solution used to run the proposed experiments. The development process firstly involved implementing the basic functionality offered by the lab tutorials and then modifying them with advanced functionality to allow experimentation. This section also analyses the datasets used in the experiments and details the expected results of experimentation on these datasets.

2.1 Basic Functionalities

Following the lab tutorials, weeks 15-19, the basic functionalities that the system would need to operate were implemented. This code featured the fundamentals in generating a Motion History Image (MHI), calculating the bag-of-words from this image, then subsequently training and testing a classifier. The MHI is generated through calculating the difference between frames across the videos' duration. For efficiency and reducing time cost of processing each video, every five frames are taken from the video. The final result of the generated MHI should show the movement present throughout the video. In a perfect scenario, this should highlight the action being carried out by the actor in the video. No other details such as colours or objects displayed in the video are relevant to this experiment and are, as such, discarded.

There existed a lot of repeating code in the four tutorials, this was all condensed into single functions that could be called by later methods. Separating this code into functions made it easier to debug and kept the process consistent across the entire system. This had the added benefit of improving the easiness in which to alter the constants and components to further experiment. Following this, the code was then altered to allow components such feature detectors to be switched out depending on the experiment being conducted. This made the experimentation process more efficient. Within the basic functionality, the feature detection and extraction is handled by SIFT and SVM is used as the classifier.

To be able to handle a dataset, the system reads in a text file containing the file path for every video to be used for training/testing alongside the label that describes the action present in the video. For the datasets that will be used in the experiments, they will have to be organised in such a way that the system in its current form can understand. The text file list can easily be generated by a Python script, however. A different list of videos is used for training and testing to avoid the classifiers simply matching the labels it used in training.

Lastly, as a performance measure of the classifiers, this basic functionality includes generation of a confusion matrix. This functionality will need to be adjusted when a different classifier is used. The confusion matrix will plot in a table the predicted values compared to the true values. This should indicate what action the classifier is trying to predict in the video compared to the actual label of the action. By viewing the confusion matrix, some insight into the behaviour of the classifier will be able to be analysed.

2.2 Advanced Functionalities

With the basic functionality implemented, additional components were then able to be added. At least two of each component were intended to be implemented into the system, with more possibly being added if there was time to do so. This at the very least allowed for experimentation across all primary components in the system. As an alternative for SIFT, SURF was included as an option for the feature detector and extractor. SURF is intended to be an improvement over SIFT by its creators and should result in faster and more accurate detection [3]. Similarly, Brute-Force was added as an alternative to the Flann-based image matching already present in the basic functionality. Brute Force matching is more simplistic than Flann-Based matching and supposedly costs more in execution time when used

on a large dataset [4]. A second classifier, Normal Bayes, was then implemented as an alternative to SVM. With these additional components added to the solution, a larger set of experiment will be able to be performed efficiently.

2.3 Dataset

Two distinct datasets will be used in the experiments with different components. The first dataset is the Human Motion Database (HMDB) provided Serre Labs [1]. This dataset features videos sourced from movies. Each video is at most a few seconds long and focuses on a singular action. This dataset was reorganised into a format that can be processed by my solution. This involved creating a list of all files to train/test and situating all the video files into one folder. To avoid spending too much time training since multiple experiments will be carried out, I restricted the number of activities and as such, the number of labels to ten. Ten videos for each label were then used for testing, while the remaining videos used for training. The total number of training videos is 1702.

The second dataset being used is a smaller set of videos which each feature a single action which was developed for the use of SVM experiments [2]. These videos focus on the single action and were recorded to clearly demonstrate it, with the actor in the centre of the screen and with minimal background movement. I expect this dataset to result in a more accurate classifier since the lack of camera movement allows for a clearer MHI to be generated for each video. The videos present in this dataset are also longer than the ones used in HMDB. This may improve the MHIs being generated, since the action is repeated multiple times during each video. However, since this is a smaller dataset, the classifier may not have enough content to be well-trained. The total number of training videos is 538 and the number of test videos is 60. In this dataset also, the number of labels is six.

3 Results

This section will detail the experiments conducted on both datasets, alongside discussion and analysis of the results. Quantitative data will be given as much as possible to provide evidence of the experiments results and to back up given theories and analysis. On all results but the classifiers, the accuracy of the classifier and the time of execution will be recorded. While accuracy may be more important, the time cost of each component is also critical. This is especially important when using large scale datasets. For the classifiers, the confusion matrix will also be given. As mentioned in Section 2.1, this will give insight into the behaviours of the classifiers. The confusion matrices can be found as tables within this report.

3.1 SURF and SIFT Descriptor Detectors and Extractors

This experiment will determine the effects of using SURF or SIFT on the performance of the system. For the purposes of this experiment, the Flann-Based matcher and SVM classifier will be used.

SURF and SIFT are both descriptor detectors and extractors. These components are used to identify keypoints in the MHI generated for each video in the dataset. Following this, a keypoint descriptor is calculated from the area around each keypoint. A descriptor matcher then attempts to use these to match keypoints between two images. I expect SURF to perform the fastest between the two, since that was the intention behind its development [3].

The results show that SURF is the more accurate and faster detector and extractor in both datasets. In dataset 1, SURF was 16% accurate and in dataset 2, SURF was 51% accurate. For SIFT, it was 11% accurate and 48.3% accurate in dataset 2. While this margin of increased accuracy is small, it is still an increase. In dataset 1, SURF executed approximately 8 minutes faster than SIFT, while in dataset 2, SURF executed a minute faster. This shows the speed of execution depends somewhat on the size of the dataset.

3.2 Flann-Based and Brute-Force Matching

This experiment will test the performance of the Flann-Based Matcher and the Brute Force Matcher. In this experiment, SURF and SVM were used. These components are used to match keypoints between two separate images. While Brute Force matcher is fairly simple, Flann is more advanced, but both are intended to find the nearest neighbour to each keypoint [4].

The results of this experiment demonstrated that the Brute-Force Matcher was faster than Flann when using dataset 1 and 2 but only marginally. In dataset 1, the experiment completed 103 seconds faster when using the Brute Force matcher. Interestingly, only in dataset 1 was there a difference in accuracy. The experiment using Brute-Force matching was 13% accurate compared to the Flann experiment which was 16% accurate. In dataset 2, both experiments were 51.67% accurate. This may have been caused by the larger dataset making the accuracy measurements more precise, or there was a slight difference in the results of training the classifier. Either way, the results show that using the Brute Force Matcher is faster than Flann.

3.3 SVM and Normal Bayes Classifiers

SVM and Normal Bayes are both classifiers. Their role is to take an image and the features found by the feature detectors and extractors and attempts to label it. In the case of this experiment, the classifiers will be trained to recognise specific human actions and then label it accordingly. Here, the accuracy of the classifiers is the most significant result to review. Alongside the accuracy and time of execution, the confusion matrix generated from the predictions given by the classifier is presented. This should provide insight into how the classifier tends to predict actions in images. This component should have the most significant impact on the performance of the solution as it considerable in computing power costs and will directly affect the accuracy of the whole system. In all the following experiments, SURF and Flann was used alongside the selected classifier.

When comparing the results of dataset 1, the SVM classifier was both less accurate and slower to execute. Normal Bayes required 430 seconds to execute, compared to SVM, which completed in 889 seconds. The accuracy of SVM is

also questionable at only 16%. Though this is only 3% less accurate than Normal Bayes, the confusion matrix, (Table 1) shows that SVM tried to classify the vast majority of actions as walking. The only other actions SVM tried to predict are catching, punching and sitting up. It can be inferred that SVM's accuracy can be credited to it trying to label almost every action as walking. Comparing this to the Normal Bayes confusion matrix (Table 2), the range of predictions across the entire selection of labels is far more varied. Despite this variation, the accuracy is only marginally higher at 19%. This shows that there is a possibility that the classifier was "guessing" the action shown in the video.

Dataset 2 has different results, here the SVM classifier was more accurate at 51.6% compared to Normal Bayes which was 41.67% accurate. While SVM is more accurate, it is still slower than Normal Bayes having taken 115 seconds to execute compared to Normal Bayes 70 seconds. The confusion matrix (Table 3) for the SVM classifier shows a larger variance in predicted labels compared to the confusion matrix from dataset 1. The confusion matrix generated for the Normal Bayes classifier (Table 4) is similar to the matrix generated for dataset 1. This matrix shows a consistent spread of predictions across the range of labels.

TABLE 1
SVM, SURF, Flann, Dataset 1 Confusion Matrix

5	0	0	1	0	1	3	0	0	72
0	0	0	2	0	0	4	0	0	134
0	0	0	0	0	0	0	0	0	124
0	0	0	12	0	0	2	0	0	81
0	0	0	0	0	0	5	0	0	219
0	0	0	5	0	0	5	0	0	142
0	0	0	2	0	0	19	0	0	92
0	0	0	0	0	0	2	0	0	132
1	0	0	0	0	0	1	0	0	93
2	0	0	4	0	0	12	0	0	496

TABLE 2
Normal Bayes, SURF, Flann, Dataset 1 Confusion Matrix

38	3	2	0	8	10	7	9	6	2
19	17	20	13	10	10	26	8	4	3
2	3	66	26	14	7	6	8	2	2
0	4	20	45	9	6	14	5	2	0
12	14	44	41	47	6	27	12	19	19
11	12	50	36	6	16	14	3	5	3
13	9	2	21	1	2	42	2	0	1
15	8	29	18	13	7	12	13	8	4
9	8	14	15	7	5	3	3	13	3
46	44	102	63	69	29	57	45	25	37

4 Discussion and Conclusions

This section will discuss the results of the above experiments. It will detail the combination of detectors, extractors, matchers, and classifiers that gave the best results while also giving an explanation into possible reasons why. A brief conclusion into this experiment will also be detailed.

4.1 Best Combination

From the results, the best combination of components seems to SVM, SURF and Flann when using dataset 2 and Normal Bayes, SURF and Flann when using dataset 1. SURF and Flann are the common components in the two combinations. The results showed that these provided the more accurate classifier, though Flann can increase the execution time. This will ultimately have an impact on a larger dataset where the Brute-Force matcher may be the preferred option.

4.2 Dataset Analysis

When reviewing the results from experimenting between the datasets, the overall accuracy of the experiment was significantly lower when using dataset 1. This was expected as mentioned in Section 2.3 as this is more than likely due to the increase in background movement. This increase in exaggerated movement resulted in the MHI generated being less clear and obscuring possible keypoints. Currently, only around 10% of the total images are used for testing. Increasing the number of test images may also increase the accuracy of the classifier.

4.3 Results Analysis

Expectedly, the results differed when changing the classifier. However, I didn't expect the change in dataset to have such a profound change in results. This is perhaps evidence that the MHI generation is flawed and cannot take into account changes in scale as well as changes in background movement. This is also evidenced by the low accuracy yielded by the classifiers. Stabilising the videos so that any movement of the camera is less of an issue may help to alleviate some background movement throughout the video, but it will not help if the background itself is moving. This would also make the system considerably more complex and would require a lot of preparation work on the dataset. In order to improve the current system, it should be modified to be as accurate as possible with a particular set of components before different components are swapped in. This should ensure that the entire system is operating at peak efficiency and that any external factors that could affect the components has less of an impact.

4.4 Further Experiments

This experiment could be furthered with the addition of more components for more comparisons. This could include the use of ORB or Decision Trees, for example. This would give a more definitive result into which is the best combination of components, though I believe that this result would depend also on the selected dataset. A more advanced method of creating more accurate MHI images could also be investigated. This would solve the problems created by using dataset 1.

References

[1] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. *HMDB: A Large Video Database for Human Motion Recognition.* ICCV, 2011.

TABLE 3
SVM, SURF, Flann, Dataset 2 Confusion Matrix

51	21	9	0	11	1
22	48	10	2	6	3
17	9	59	7	8	2
0	0	3	39	10	28
0	1	3	17	46	16
2	0	2	17	0	60

TABLE 4
Normal Bayes, SURF, Flann, Dataset 2 Confusion Matrix

27	36	12	4	2	2
23	41	10	1	2	0
8	13	47	3	4	6
1	0	1	46	22	14
1	3	8	26	53	13
0	0	1	22	5	73

[2] C. Schuldt, I. Laptev and B. Caputo, "Recognizing human actions: a local SVM approach," Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004., 2004, pp. 32-36 Vol.3, doi: 10.1109/ICPR.2004.1334462.
[3] Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features." *European conference on computer vision*. Springer, Berlin, Heidelberg, 2006.
[4] "OpenCV: Feature Matching", *Docs.opencv.org*. [Online]. Available: https://www.docs.opencv.org/master/dc/dc3/tutorial_py_matcher.html.