Moving and Static Object Recognition using VGG

Report_PA4

Ahmed, Varghese, Satyanarayana, Samant
Electrical & Computer Engineering
Colorado State University

***Abstract*— The paper presents our implementation of PA4 which demonstrates the recognition of static and moving objects in a single frame by using convolutional neural networks. In PA2, we extracted the best feature points from a frame and in PA3, we not only extracted various foreground pixels using background subtraction methods but also tracked those pixels throughout the video using MOSSE tracker. Thus, by combining PA2 and PA3, we extract the best static feature and various moving objects and give it as an input to VGG net so that we can recognize the respective features.**

I. INTRODUCTION

As Fig 1. suggests, the video is given as an input to two different programs. Both the programs extract the video frame-by-frame to do the following processing:-

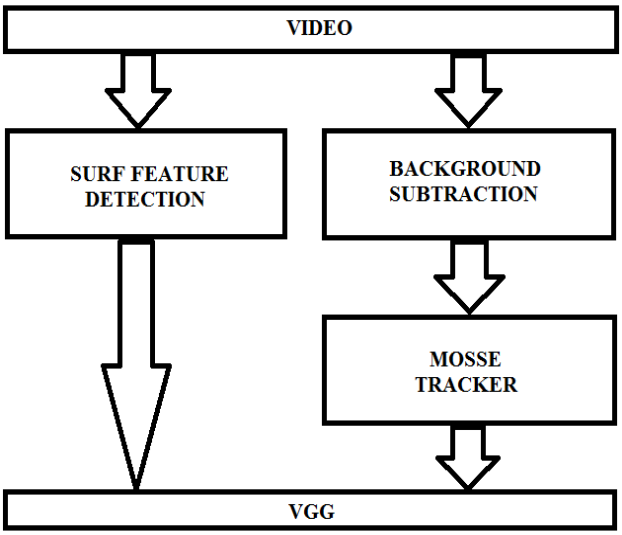

***FIG: 1** Block Diagram*

We used the SURF feature detector from OpenCV library. We first decide the Hessian threshold according to the video using trial and error. It helps us to detect the most significant key points in a frame. We must change it according to the video as different videos may have a different number of interesting features. For example, a video may have a lot of interesting static features to capture thus, to get the best features we must increase the Hessian threshold and vice-versa. The inclusion of Hessian threshold reduces the processing time while looking for the best feature among many others. Then, we just extract the best feature point among many based on the response and size of the features. We also append these features frame-by-frame until the end of the video because our objective is to obtain the five best static features in the video.

In background subtraction method, background subtractor from OpenCV was used. It includes the actual subtraction between t and t+1 frame where t is the present frame. As observed in the previous assignment, ViBE was the best method for background subtraction but it was slow as compared to background subtractor method and as we were combining the system with MOSSE and VGG, the system was not able to process the video in real time. And, as in MoG, the ghost was very significant, we preferred using background subtractor as it was fast and relatively accurate.

Depending on the number of the moving objects in the frame, we had to use MOSSE tracker in the upcoming frames.

The tracked objects are given as input to the VGG and hence, recognized.

VGG asks input as two images at once and gives the probability of the match within the 1000 classes. Thus, we get two probabilities for each frame, one for static and other for moving. As we had to find the five best static and moving objects, we append the frame, contour and matching probability for each frame until the end of the video and sort it in descending order with probability as the key to obtain the best five objects in static and moving domain.

II. METHODS

A. SURF Implementation

SURF is included in the OpenCV package. The Hessian threshold is set and acts as a parameter to control the number of key points selected in a frame. The key points are obtained by finding the maximums from the images resulting from the difference of Gaussians followed by Laplacian of Gaussian approximation. The Hessian matrix is the basis for the detection of the feature points because of its good performance in computation time and accuracy. The Hessian determinant determines location and scale, and applying non-maximum suppression and interpolating the maxima of the determinant we can obtain the key points. The descriptor extraction method can be given in two steps; orientation assignment and descriptor extraction.

B. Background Subtraction

In the background segmentation method, each frame is converted to grayscale and a Gaussian blur is applied to it. Here, the current frame is subtracted from the background model and a threshold value is applied to obtain a foreground model. This method of background and foreground segmentation is relatively easier to implement compared to the Mixture of Gaussians and ViBe methods described below. A visual representation of the basic background segmentation method is given in Fig 2.

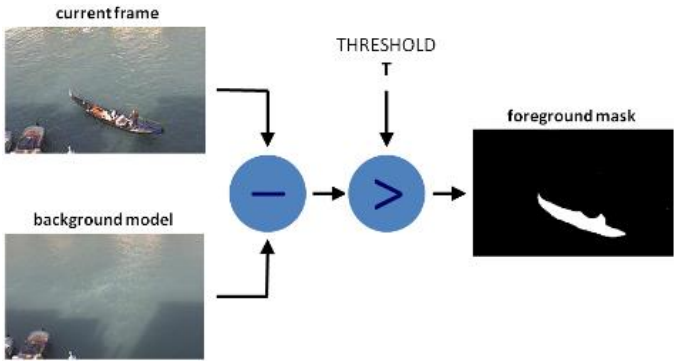

Fig 2: Basic Background Segmentation Observation

C. Mosse Tracking

Once the frames in the video have been processed to classify the background and foreground, we now need to keep track of the objects moving in and out of the frames. The tracking algorithm needs to be robust to variations in lighting, scale and non-rigid deformations. To track the objects, we have used an implementation of MOSSE filter (Bolme, Beveridge, Draper, and Liu).

In this method, we see that the multiple moving objects within the frames are rotated and translated in random ways and a filter is generated from these images. This filter is compared to the moving objects in the coming frame and if the filter gives a good match with the object, it is tracked in the ongoing frames.

D. Convolutional Neural Networks

	Validation Set of Places365		Test Set of Places365	
	Top-1 acc.	Top-5 acc.	Top-1 acc.	Top-5 acc.
Places365-AlexNet	53.17%	82.89%	53.31%	82.75%
Places365-GoogLeNet	53.63%	83.88%	53.59%	84.01%
Places365-VGG	**55.24%**	84.91%	**55.19%**	85.01%
Places365-ResNet	54.74%	**85.08%**	54.65%	**85.07%**

Fig 3- Performance Table

Before 2012, we had object recognition methods which needed powerful GPUs and a large memory and it took a lot of time to train. But once AlexNet was developed, it brought a new era of Convolutional Neural Networks. AlexNet was a neural net which was just 8 layers deep and gave us accuracy close to 70% which was much better than the previous methods. Moreover, it had significantly fewer weights to train, thus reducing the time incorporated to train the weights. Over the years, with the development of GPUs and storage, ConvNet has significantly improved to obtain results better than the previous versions by increasing the depth of the layers.

In 2016, we saw the development of ResNet which was 150 layer deep and is the current state-of-art method for object recognition and works best for ImageNet dataset. But it involves high cost and requires a huge amount of resources thus, cannot be used for educational purposes and hence, restricted to commercial purposes.

But, we had one more method last year which was second best to ResNet which was able to classify the images with excellent accuracy with just 16 or 19 deep layer network. It's called VGGNet.

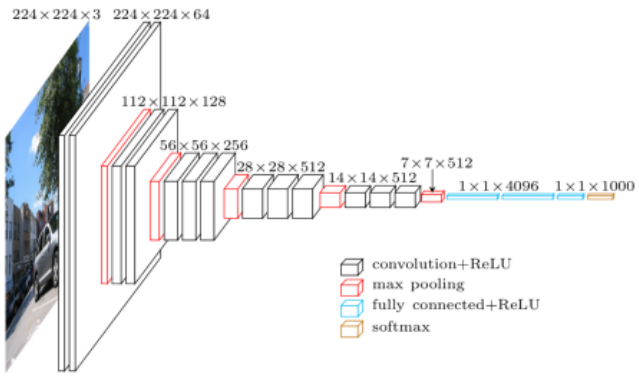

Fig 3. VGG16 net structure

During training, the input to our ConvNets is a fixed-size 224 × 224 RGB image. The only preprocessing is subtracting the mean RGB value, computed on the training set, from each pixel. The image is passed through a stack of convolutional (conv.) layers, where we use filters with a very small receptive field: 3 × 3. In one of the configurations, they also utilize 1 × 1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity). The convolution stride is fixed to 1 pixel; the spatial padding of conv. layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1 pixel for 3 × 3 conv. layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv. layers (not all the conv. layers are followed by max-pooling). Max-pooling is performed over a 2 × 2 pixel window, with stride 2.
A stack of convolutional layers (which has a different depth in different architectures) is followed by three Fully-Connected (FC) layers: the first two have 4096 channels each, the third performs 1000way ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-

max layer. The configuration of the fully connected layers is the same on all networks.
All hidden layers are equipped with the rectification non-linearity.

E. Haar Cascade

A classifier (namely a *cascade of boosted classifiers working with haar-like features*) is trained with a few hundred sample views of a particular object (i.e., a face or a car), called positive examples, that are scaled to the same size (say, 20x20), and negative examples - arbitrary images of the same size.

After a classifier is trained, it can be applied to a region of interest (of the same size as used during the training) in an input image. The classifier outputs a "1" if the region is likely to show the object (i.e., face/car), and "0" otherwise. To search for the object in the whole image one can move the search window across the image and check every location using the classifier. The classifier is designed so that it can be easily "resized" in order to be able to find the objects of interest at different sizes, which is more efficient than resizing the image itself. So, to find an object of an unknown size in the image the scan procedure should be done several times at different scales.

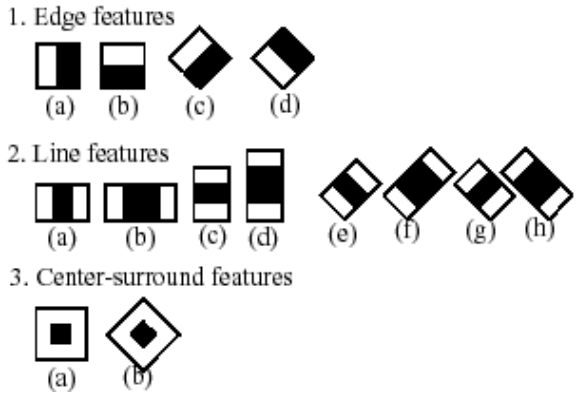

Fig 4 – HAAR-like features

A Haar-like feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums. This difference is then used to categorize subsections of an image.

III. EVALUATION

A. Base Implementation

During PA2 and PA3, we spent a lot of time evaluating the different methods, noting down their advantages and disadvantages and tested these methods against time, accuracy, precision, robustness etc. Using our observations from these assignments, we were able to decide the best possible methods we can use for PA4.

We used two methods to obtain static key features in an image, SIFT and SURF. SIFT was very accurate and used to produce fewer key points as compared to SURF, but the key points were small in size and as previously mentioned we distinguished the best key points using not just the response but also the size so that VGG will be able to recognize the feature. But we observed that when we used that condition, we didn't get any significant key points and hence, we moved to SURF so that we can obtain some key points. Now, it is a trade-off for accuracy and there's one floating parameter. We had to adjust Hessian threshold in accordance to different videos to adjust the window size.

For background subtraction, we used background subtractor from OpenCV, which was not as accurate as ViBE but it was fast and the ghost was not prominent in the image. ViBE being accurate, was slow and it was really very difficult to combine it with MOSSE and VGG and work it in real-time. Hence, background subtractor was a clear option.

MOSSE tracker was the best choice while tracking. We tried implementing KALMAN filter but were unable to implement it with the robustness and accuracy that MOSSE provided us.

We also used HAAR-Cascade method as a substitute to robust MOSSE tracker. We used it because the implementation was relatively easy. The system was able to track the moving objects really very well but the only problem was that we had to manually train cascade filters for each object, for example-car, humans etc. It might work in a practical setting where we know the objects we are tracking but it will fail miserably when it's introduced to new objects.

VGG was the only CNN we used because of the unavailability of time and resources to implement the other heavy CNNs.

B. Observation

We implemented the code on many videos, walking video, videos related to dogs, example video given in class and many others. When we implemented the code on the walking video, human beings were recognized as snakes. It worked perfectly while detecting the pick-up truck in example video. Even the IP address video worked well. We also tried it on another car video where it detected different forms of the car with 80% accuracy. But while detecting the static images, unless we get the whole object into one contour it becomes difficult to recognize objects. The sky was detected as lakefront, road demarcation as flight wing and building as a whole dockyard. Here, are few examples related to the code-:

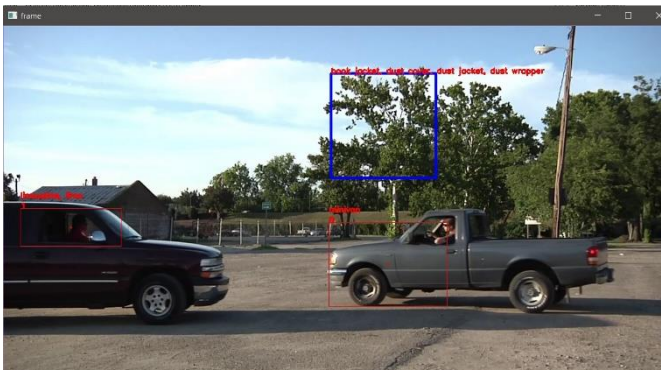

Fig 5 – Recognition using MOSSE & VGG on example video

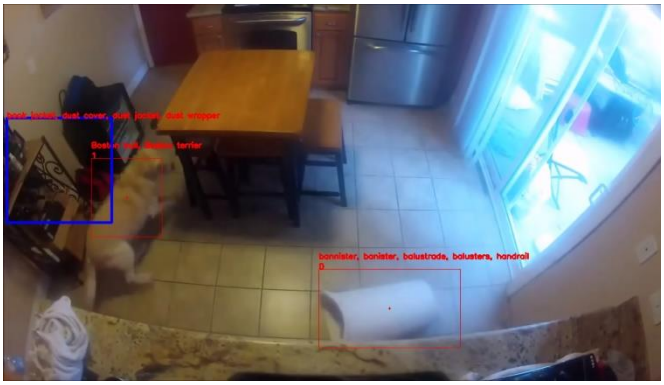

Fig 5 – Recognition using MOSSE & VGG on dog video

As previously mentioned, we implemented code using haar-cascade as well. Haar-cascade performed better as compared to the one using MOSSE. The example is as follows-:

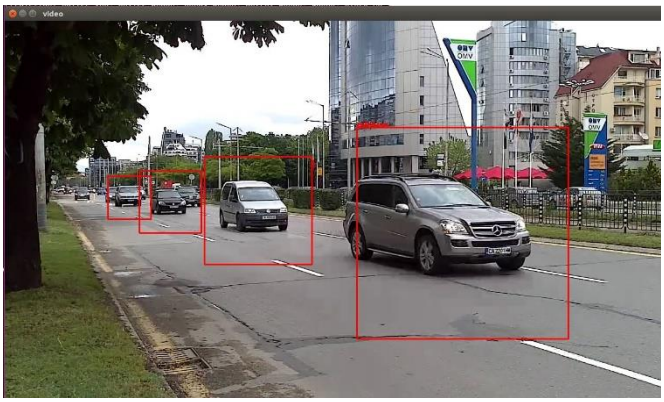

Fig 5 – Recognition using haar-cascade & VGG on cars video

C. *Unbiased Review*

In order to obtain unbiased reviews about the program, we talked to people outside CS department and asked for their views on tracking using both MOSSE and Haar-cascade. More than 70% of the people were in support of Haar-cascade. We also asked them about the general performance of the code. People were not satisfied with the performance of the code because of the combined errors from SURF, background subtraction, MOSSE, and VGG as well.

IV. Conclusion

We successfully implemented the recognition system where we are detecting both static and moving objects. While testing on various videos, we observed that we did a good job with moving objects where we obtained considerable accuracy but with still objects, we were not able to obtain a high precision. But that was also because VGG was trained just for 1000 classes and it was not able to recognize innumerous other classes which a human eye can recognize. Thus, we see that there's a scope for huge improvement in the field of computer vision as we have only been able to achieve high accuracy for 1000 different classes and we have a million others which we cannot recognize at the moment.

References

1] Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features." *European conference on computer vision*. Springer Berlin Heidelberg, 2s006.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] Lowe, David G. "Distinctive image features from scale-invariant keypoints." *International journal of computer vision* 60.2 (2004): 91-110.

[3] Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features." *European conference on computer vision*. Springer Berlin Heidelberg, 2006.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[4] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[5] K. Elissa, "Title of paper if known," unpublished.

[6] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[7] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[8] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.