Anomaly Detection for Video Surveillance Applications

Carmen E. Au, Sandra Skaff, and James J. Clark
McGill University, Montreal, Canada

Abstract

We investigate the problem of anomaly detection for video surveillance applications. In our approach, we use a compression-based similarity measure to determine similarity between images in a video sequence. Images that are sufficiently dissimilar are deemed anomalous and stored to be compared against subsequent images in the sequence. The goal of our research is two-fold; in addition to detecting anomalous images, the issue of heavy computational and storage resource demands is addressed.

1. Introduction

In this paper, we consider the problem of detecting anomalous or novel images in a sequence of images. The need for heightened security is unfortunately becoming more prevalent in today's world. While aptly-placed video cameras are intended to capture the images of possible offenses or offenders, the majority of the time, it is but a security guard who spends his or her time staring into a series of uneventful monotonous images. Since new events rarely occur, it is extremely difficult for a security guard to remain vigilant at all times. Thus, our work removes the onus of detecting anomalous situations from the security guard; and rather, places it on the video surveillance system. We developed a compression-based similarity measure technique, which compares the sizes of compressed images to detect for anomaly. While there exist other approaches to automated video surveillance [3, 4, 7], our compression-based technique inherently provides storage and computational resource reduction.

One way of detecting novel images is to compare each new image to all the previously-seen images. However, this method is not ideal; storing every image of the video sequence requires significant memory and far too many unnecessary comparisons. Thus, by storing only the novel images, it is only the images with useful information that are kept and used to be compared against future incoming images.

2. The Similarity Measure

A scene is considered anomalous when the maximum similarity between the scene and all previously viewed scenes is below a given threshold. We propose to use a similarity measure that quantifies the mutual information between data sets. By definition, compression programs aim to remove redundancies within a file. Li *et al* [5] proposed a similarity metric that involved compressing each data set independently, measuring the sizes of the compressed results and then concatenating the two data sets and compressing the combination data set. The idea is that the size of the compressed version of the concatenation of two similar files would be smaller than that of two dissimilar files. In terms of the normalized compression distance (NCD) developed by Li *et al*, our similarity function is given by:

$$p(I_1, I_2) = \frac{size(C[I_1]) + size(C[I_2]) - size(C[I_1 \oplus I_2])}{size(C[I_1]) + size(C[I_2])} \quad (1)$$

where C[] indicates the compression operation, and $\oplus$ indicates the concatenation of the data sets. Though this compression-based technique discards much information, because we are merely detecting anomalous images instead of attempting to detect *and* recognize the nature of the dissimilarity, we are not concerned with the discarded information. In fact, since our system need only be able to say that it has seen a similar image before, and does not need to say *what* is similar, we could even implement a particularly lossy compression technique which throws away much information that would otherwise be needed to recognize what is dissimilar. However, not all types of compression algorithms are suitable for implementing such a similarity measure. We require an algorithm that can detect and remove the redundancy in a concatenation of two images. Block-sorting lossless algorithms, such as those derived from the Burrows-Wheeler transform [1], for example the *bzip2* algorithm, are well-suited to our needs.. By default in the *bzip2* program, files are compressed in 900 KB

The 18th International Conference on Pattern Recognition (ICPR'06)
0-7695-2521-0/06 $20.00 © 2006 IEEE

block-sizes. In order for the metric not to be skewed by the size of the objects, the block-size needs to be at least as big as the size of the file to be compressed. In our demonstration system, all images are of equal size, and thus, the images must be at most 450 KB in size, since our similarity measure takes the size of the concatenation of the two files being compared as input. Given that the images obtained from our camera are roughly 102 KB, the standard *bzip2* compressor is an acceptable compressor for the NCD.

Similarity is determined using equation (1). Given two images, I_1 and I_2, both images are compressed individually, and the concatenation of the two images is compressed. If the resulting similarity measure, $\rho(I_1,I_2)$, is greater than or equal to a given threshold, then the two images are considered similar, otherwise, the two images are considered dissimilar.

3. Minimizing Computation and Storage

At first glance, it would appear that even by only saving the novel images, computational and storage resources still grow linearly with time. This proves to be impractical if the algorithm is used for video surveillance applications. However, we note that in many applications, the rate of occurrence of novel inputs will actually decrease over time. Initially, everything will appear novel; however, as time goes on, inputs will begin to repeat themselves, and fewer novel images will be detected. This decrease in novelty can be likened to a newly-hired security guard. On his or her first day at work, everything appears new. Little by little, the building 'regulars' become familiar to him or her, while the appearances of new people would rarely occur. Although the number of stored images is still increasing, and thus the overall computation is increasing, because the rate of novel inputs is decreasing with time, the increase in computation is sublinear.

The order in which the comparisons are made can also decrease the computational load. In our approach, we compare the current input to the images that are most likely to be similar to them. This is done by sorting the stored images by the amount of times they have been deemed similar to an input, from the most often to the least. The reasoning is that an input that will ultimately be classified as similar will most likely be considered to be similar to one of the images that has been seen most frequently in the past. For example, if the video camera were surveying a corridor, then an empty corridor with no people walking by would likely be the most frequently viewed image. Thus, it would be reasonable to rule out that image first, by comparing the new input to it first. Of course, inputs that will ultimately be determined to be novel will be compared against every stored image; and in that case, sorting is irrelevant. However, as mentioned, the number of novel images will decrease over time, and thus, these cases will be rare and the overall computational load is decreased.

4. Experiments on Indoor Images

Before we could begin testing the system, we needed to establish the threshold to which all similarity measures would be compared. A set of 100 images was manually labelled as similar or dissimilar. The threshold was set to a value of 0.055 so that the greatest number of dissimilar images could be obtained, without exceeding a false positive rate (similar images deemed as dissimilar) of 2%. We applied the compression-based anomaly detector to a sequence of intensity images that were obtained from a surveillance camera that was placed in a corridor at our research center. The images were taken over a period of two weeks. In Figure 1, we drew a box around one day of events. In this box, we can see that at the beginning of each day, with the morning rush of people arriving to work, the system detected many novel images; however, the number of novel images detected slowly tapered off during the course of the day (as the camera would simply be recapturing the images of the

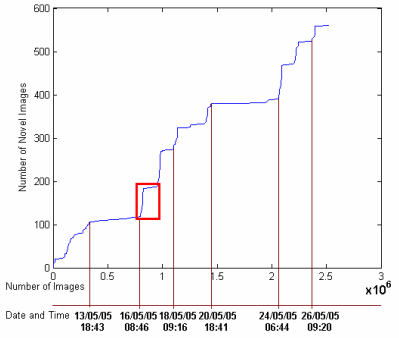

Figure 1. No. of novel images Vs No. of intensity images processed over 2 weeks.

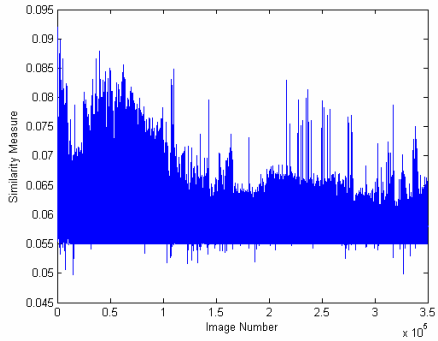

Figure 2. Highest similarity measure for each image.

Figure 3. Examples of similar (top) and dissimilar (bottom) image classification.

same people, but at different times). The following morning, a new influx of novel images was detected due to the subtle changes in the inputs; such as, the changes in clothing. During the weekend (20/05/05 - 24/05/05), there was less activity at the research center, and thus, very few novel images were detected; however, with the start of a new week, a series of new images was captured again.

Using equation (1), we calculated the similarity measure for each image that was captured by our camera, compared to each of the previously stored images. Figure 2 shows the plot of the highest similarity measure calculated for each image. Any image with a highest similarity measure below the threshold was considered novel. The plot shows the occasional sharp dip, which confirms that novelty is rare.

Our system processed the intensity images as quickly as possible; as soon as one image was processed, it would begin processing another one. Thus, the rate of image acquisition was highly dependent on the time it took to process each image. Initially, when processing the images, the rate of image acquisition was approximately 0.5 sec/image. However, as the number of novel images increased, more comparisons were made in order to determine novelty, and thus, the rate decreased. At 400 novel images, the rate of image acquisition could be as slow as 6 sec/image. However, because novelty decreases with time, we would find that very few images were actually processed at the worst-case rate. In Figure 3, the top two images were deemed similar by the algorithm, and the bottom one was deemed dissimilar from the other two. Although it is difficult to provide an intuitive description of the similarity measure, it can be gathered from the similar images that the figures in the hallways are at similar locations, and dressed in a similar fashion. However, in the bottom image, the location and grey level of the figure is different from the other two images.

4.1 Edge Detection

When applied to intensity images, the anomaly detection algorithm was sensitive to changes in illumination. We wanted the algorithm to consider two images that only differed by their illumination as similar. Thus, we used a Canny Edge Detector [2] to obtain edge images from the intensity images. Applying an edge detector to an image filters out a significant portion of the image, including information about illumination, all the while preserving the important structural properties within it.

In Figure 4, we show the number of novel images over 10^5 images. The dashed line indicates the number of novel images detected when the anomaly detection algorithm was applied to intensity images, while the solid line indicates the number of novel images detected with edge images. The first noticeable advantage to using edge images was that, while it took 26 hours to process all the intensity images, it took only 6 hours for the edge images. The significant disparity between the two processing times can largely be attributed to the edge images being 2.5 times faster to compress than the intensity images. As can be seen in Figure 5, the difference between the similarity measures for the similar images versus for the dissimilar images is greater and more clearly defined for the edge images than for the intensity images.

Unlike with the intensity images, the anomaly detection algorithm was not affected by the changes in illumination with the edge images. Moreover, edge

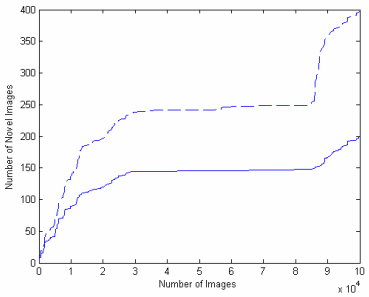

Figure 4. No. of novel images vs. No. of images for intensity images (dashed) and edge images (solid).

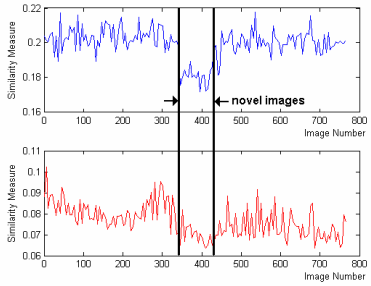

Figure 5. Similarity measure for edge images (top) and intensity images (bottom)

images also discarded information such as clothing changes, implying that while with the intensity images, for each location a person was in, the algorithm could potentially store as many novel images as this person had outfits; with the edge images, the algorithm would only store one. In fact, running the algorithm on 10^5 intensity images found twice the amount of novel images as with the edge images.

4.2 Person Detection

If the surveillance camera was moved slightly, then with the intensity and edge images, the algorithm would determine images that it had previously-seen as novel, because the whole image would be shifted over. However, just as a security guard would still be able to recognize a person even if the camera had been moved, we wanted to find a feature that would also render our algorithm location and scale invariant. Thus, we applied Nair and Clark's person detector [6] on a sequence of images and extracted a bounding box around each detected person. The idea was that once the bounding box was extracted and scaled, we could then use the anomaly detection algorithm to detect new people, regardless of their position. Equation (1) was applied to the cropped images to test for similarity between people.

Figure 6. Examples of applying a person detector to images.

Figure 6 shows three images that have been processed by the person detection algorithm. The person bounding boxes were cropped and scaled, and the similarity measure was applied on these subimages. It was found that the top two images were similar, despite the location of the person, whereas, the person in the bottom image was sufficiently different that it was deemed dissimilar.

Applying a person detector on the images requires that only the images where people have been detected be checked for novelty. Since a large proportion of the time, we would not expect a high volume of activity, this would greatly reduce processing time. Processing an image would also be sped up by the fact that compression would be faster with the smaller cropped images. Moreover, because there would be fewer novel images to store, and these images would be smaller, storage requirements are also reduced.

5. Conclusion

In this paper, we have introduced and defined the concept of compression-based anomaly detection, which detects when an image is different, at a given time, from all of the previously-seen images in a video surveillance application. It is a problem that seems trivial at first glance, but in practice, imposes serious demands on computational and storage resources. We attack these issues by using a compression-based technique for (dis)similarity measurement, by storing only the novel images, and by extracting various features from the image to be used in the algorithm. The compression technique is advantageous, in that we can store the compressed version of the images, which is an integral part in reducing the storage requirements.

Moreover, from the experiments performed on our demonstration system, we can conclude that, as expected, novel images are rare. And since in our approach, we store only the novel images, we have further reduced the storage requirements as well as the computational requirements.

6. References

[1] M. Burrows and D. J. Wheeler, "A block-sorting lossless data compression algorithm," *SRC Research Report*, vol. 124, Palo Alto, CA, May 1994.

[2] J. Canny, "Computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679-698, 1986.

[3] J. Fernyhough, A.G. Cohn, and D.C. Hogg, "Constructing qualitative event models automatically from video input," *Image and Vision Computing*, vol. 18, pp. 81-103, 2000.

[4] N. Johnson and D. Hogg, "Learning the distribution of object trajectories for event recognition," *Image and Vision Computing*, vol. 14, pp. 609-615, 1996.

[5] M. Li, X. Chen, X. Li, B. Ma, and P. M. B. Vitanyi, "The similarity metric," *IEEE Transactions on Information Theory*, vol. 50, pp. 3250-3264, 2004.

[6] V. Nair and J. J. Clark, "An unsupervised, online learning framework for moving object detection," 2004 IEEE Conference on Computer Vision for Pattern Recognition, Washington D.C., United States, vol. 2, pp. 317-324, 2004.

[7] C. Stauffer and W. E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 747-757, 2000.