Real-time Scalable Video Stream Analysis with Object, Event and Anomaly Detection.

Joseph Honour - 130291538

Newcastle University

Author Note

Completed as part of my undergraduate BSc in Computer Science with Industrial Placement (G401), supervised by Matt Collison.

Abstract

Computer vision has been a large area of research in recent years, devising methodologies to understand and act on events seen within video streams. A major application of computer vision is to detect anomalies autonomously, and alert users to when they occur. Although industrial technologies exist that are able to do this to a basic standard, they often rely on expensive and exclusive hardware.

This paper proposes an extendable and scalable framework that is able to provide accurate anomaly detection, in real-time, without complex hardware requirements. The framework will show how the adoption of distributed computing and machine learning enable real-time anomaly detections, without requiring specialized hardware. My design approach is to allow extensibility at every opportunity so the framework can be adapted for a multitude of use cases, some of which I propose within this paper. Furthermore, the framework will allow horizontal scaling enabling it to handle large volumes of data, while keeping its real-time requirements intact. Finally, the framework will be hosted publicly allowing new avenues to be explored by the community, with avenues of exploration suggested at the end of this paper.

Declaration

“I declare that this dissertation represents my own work, except where otherwise stated.”

Acknowledgments

This is my acknowledgments.

Table of Contents

[Abstract 2](#_Toc505790110)

[Declaration 3](#_Toc505790111)

[Acknowledgments 4](#_Toc505790112)

[Introduction 7](#_Toc505790113)

[Motivation 7](#_Toc505790114)

[Aim 7](#_Toc505790115)

[Objectives 7](#_Toc505790116)

[Paper Structure 7](#_Toc505790117)

[Background and Literature Review 8](#_Toc505790118)

[Video Processing Methodologies and their Adoption 8](#_Toc505790119)

[Object Detection Techniques 9](#_Toc505790120)

[Object Tracking Techniques 10](#_Toc505790121)

[Event Detection Techniques 11](#_Toc505790122)

[Anomaly Detection with Machine Learning 11](#_Toc505790123)

[Anomaly Detection Models 11](#_Toc505790124)

[The Impact of Human Behavior 11](#_Toc505790125)

[Distributed Computing and the Cloud 11](#_Toc505790126)

[Cloud Providers 11](#_Toc505790127)

[Distributed Computing 11](#_Toc505790128)

[Distributed Messaging 11](#_Toc505790129)

[Existing Technologies and Approaches 11](#_Toc505790130)

[References 12](#_Toc505790131)

[Footnotes 14](#_Toc505790132)

[Tables 15](#_Toc505790133)

[Figures title: 16](#_Toc505790134)

# Introduction

## Motivation

This will be my motivation.

## Aim

Propose an extendable framework for client-side object detection, accompanied by server-side event and anomaly detection, making use of distributed computing and machine learning.

## Objectives

1. To research existing video processing techniques and available software packages in order to detect objects and events within a video stream.
2. To research machine learning techniques for detecting anomalies in time series data produced from objective one.
3. Develop testing scenarios that will allow the evaluation of machine learning models in their ability to detect anomalies in real-time.
4. Develop a framework that provides a minimum viable product of object, event and anomaly detection, while being scalable and extensible.
5. Using the test scenarios defined in objective three, evaluate the applications ability to detect anomalies and alert users in real-time.
6. Compare and contrast the performance and storage requirements of the proposed framework against existing CCTV technologies and approaches.

## Paper Structure

I will describe my paper structure here.

# Background and Literature Review

## Video Processing Methodologies and their Adoption

In order to develop an effective video processing framework, we must be able to understand what is within each frame of a video to a level that we can perform any desired actions, such as detecting anomalies. Architecture designs for this have been presented in the past (Figure 1), that show theoretically the different stages you must build upon to enable successful video processing.



Figure 1: Adaptation of video processing within a general framework for automated visual surveillance system (Ko, 2008).

The adoption of a framework following this structure allows for accurate information to be captured at different levels of processing, enabling complex behavior and identification of objects to be accomplished.

### Object Detection Techniques

Detecting objects in images relies on being able to interpret the combinations of pixels correctly to identify the particular object you are looking for. The most common method of doing this is using a Haar feature-based cascade classifier (P. Viola and Jones, 2001). This machine learning approach works by showing a classifier a multitude of images, with some containing the object you wish to detect. The classifier then attempts to apply features to the image that allow it to accurately detect the desired object. Once the classifier is trained, we extract the Haar features from the images, where each feature is a single value obtained by subtracting the sum of pixels under the white rectangle from the sum of pixels under the black rectangle (Figure 2).

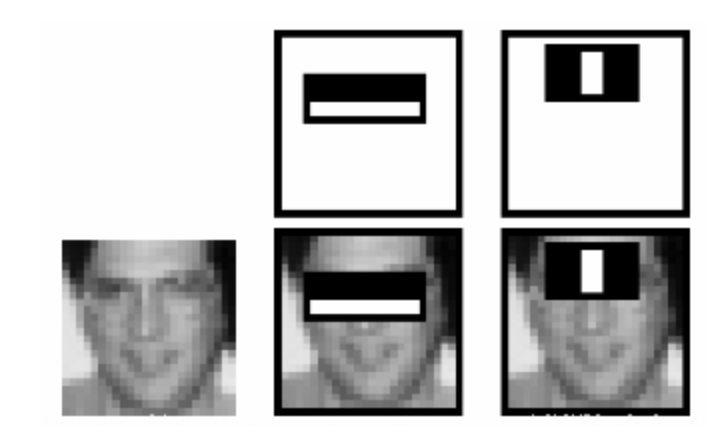
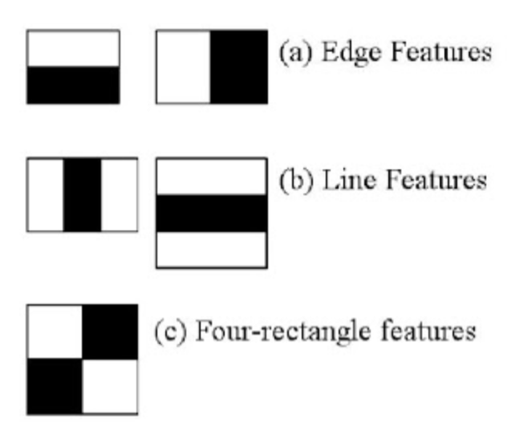


Figure 2: Left, an example of Haar cascade features. Right, the adoption of Haar cascade features in detecting a face (OpenCV, 2018).

From this point we are able to improve the performance of detections using an algorithm called Adaboost (P. a Viola and Jones, 2001), which allows ranking of features based on their error rate when attempting to successfully identify the object.

Further work has been proposed in this field using a deep learning based approach (Lecun, Bengio and Hinton, 2015). However, to make this approach feasible within real-time constraints specialized GPU (Graphics Processing Unit) hardware must be used, which would limit the extensibility of any proposed framework making use of these techniques.

### Object Tracking Techniques

Provided an objects location is correctly calculated, the next phase of a proposed framework is to give the object persistent identity between frames, which employs tracking techniques. Tracking works on an online basis, meaning as new sequential data becomes available to the model, it can adapt at runtime to provide improved predictions on future data. Usually, this incorporates identifying an objects appearance and motion pattern allowing the accurate predictions of an objects location as it moves. When providing object tracking capabilities a variety of algorithms have been developed and made available (Table 1).

|  |  |  |  |
| --- | --- | --- | --- |
| Approach | Multiple Instance Learning | Median Flow | Tracking-Learning-Detection |
| Overview | Tracks an object by treating its location as a set of positions (‘bags’) that each could contain the objects location based on its previous location. This gives the learning algorithm the responsibility of removing the ambiguity of the exact object location and predicting which instance in each bag is most correct (Baben and Belongie, 2009). | Tracks an object by selecting a variety of points within the object space and then computes the trajectory for this object, both forwards and backwards in time. This means the tracker is able to compare both trajectories and make accurate predictions of the objects final location (Kalal, Mikolajczyk and Matas, 2010). | Breaks the tracking problem up into: Tracking, learning and detection. The tracker then follows an object between frames with the detector correcting the tracker based on all observed appearances. The learning estimates the error of the detector and updates it to avoid these errors in future (Kalal, Mikolajczyk and Matas, 2012). |
| Pros | Works well when object partially occluded due to its bag representation. | Works well when an objects trajectory is predictable. | Works well under long periods of occlusion. |
| Cons | The tracker cannot handle full occlusion of objects well. | The tracker becomes less reliable under fast or unpredictable motion. | The tracker does not perform well under large full rotations and can create lots of false positives. |

Table 1: A comparison of three common approaches to object tracking.

### Event Detection Techniques

Detection techniques.

## Anomaly Detection with Machine Learning

This will be a talk on anomaly detection and machine learning.

### Anomaly Detection Models

Talk about the models specifically and research done into them.

### The Impact of Human Behavior

Talk about how humans may affect ability to detect anomalies.

## Distributed Computing and the Cloud

This will be a talk on distributed computing and the cloud.

### Cloud Providers

Talk about cloud providers and their benefit.

### Distributed Computing

Apache Storm, talk about the key technologies.

### Distributed Messaging

Apache Kafka, talk about the key technologies.

## Existing Technologies and Approaches

This will be a talk on existing technologies.

References

Baben, B. and Belongie, S. (2009) ‘Visual tracking with online Multiple Instance Learning’, *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 983–990. doi: 10.1109/CVPR.2009.5206737.

Kalal, Z., Mikolajczyk, K. and Matas, J. (2010) ‘Forward-backward error: Automatic detection of tracking failures’, in *Proceedings - International Conference on Pattern Recognition*, pp. 2756–2759. doi: 10.1109/ICPR.2010.675.

Kalal, Z., Mikolajczyk, K. and Matas, J. (2012) ‘Tracking-learning-detection’, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7), pp. 1409–1422. doi: 10.1109/TPAMI.2011.239.

Ko, T. (2008) ‘A survey on behavior analysis in video surveillance for homeland security applications’, *Applied Imagery Pattern Recognition Workshop, 2008. AIPR ’08. 37th IEEE*, pp. 1–8. doi: 10.1109/AIPR.2008.4906450.

Lecun, Y., Bengio, Y. and Hinton, G. (2015) ‘Deep learning’, *Nature*, pp. 436–444. doi: 10.1038/nature14539.

OpenCV (2018) *Facial Detection with Haar Cascades in OpenCV*. Available at: https://docs.opencv.org/3.3.0/d7/d8b/tutorial\_py\_face\_detection.html (Accessed: 30 January 2018).

Viola, P. a and Jones, M. J. (2001) ‘Fast and Robust Classification using Asymmetric AdaBoost and a Detector Cascade’, *Advances in Neural Information Processing System*, (December), pp. 1311–1318. Available at: https://papers.nips.cc/paper/2091-fast-and-robust-classification-using-asymmetric-adaboost-and-a-detector-cascade.pdf.

Viola, P. and Jones, M. (2001) ‘Robust real-time object detection’, *International Journal of Computer Vision*, 57(2), pp. 137–154. doi: http://dx.doi.org/10.1023/B:VISI.0000013087.49260.fb.

Footnotes

1[Add footnotes, if any, on their own page following references. For APA formatting requirements, it’s easy to just type your own footnote references and notes. To format a footnote reference, select the number and then, on the Home tab, in the Styles gallery, click Footnote Reference. The body of a footnote, such as this example, uses the Normal text style. (Note: If you delete this sample footnote, don’t forget to delete its in-text reference as well. That’s at the end of the sample Heading 2 paragraph on the first page of body content in this template.)]

Tables

Table 1

[Table Title]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column Head | Column Head | Column Head | Column Head | Column Head |
| Row Head | 123 | 123 | 123 | 123 |
| Row Head | 456 | 456 | 456 | 456 |
| Row Head | 789 | 789 | 789 | 789 |
| Row Head | 123 | 123 | 123 | 123 |
| Row Head | 456 | 456 | 456 | 456 |
| Row Head | 789 | 789 | 789 | 789 |

Note: [Place all tables for your paper in a tables section, following references (and, if applicable, footnotes). Start a new page for each table, include a table number and table title for each, as shown on this page. All explanatory text appears in a table note that follows the table, such as this one. Use the Table/Figure style, available on the Home tab, in the Styles gallery, to get the spacing between table and note. Tables in APA format can use single or 1.5 line spacing. Include a heading for every row and column, even if the content seems obvious. A default table style has been setup for this template that fits APA guidelines. To insert a table, on the Insert tab, click Table.]

Figures title:

Figure 1. [Include all figures in their own section, following references (and footnotes and tables, if applicable). Include a numbered caption for each figure. Use the Table/Figure style for easy spacing between figure and caption.]

For more information about all elements of APA formatting, please consult the APA Style Manual, 6th Edition.