A Real-time Analytics Pipeline for Scalable Smart Video Surveillance.

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.

Word Count: 14,779

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. With the wide deployment of CCTV and IP cameras in the past years, the opportunity for smart video analysis is available. However, smart video surveillance remains a feature largely reserved for new hardware systems. To enable the benefits of smart video analysis over currently deployed hardware we need to overcome the process of applying smart video analytics against existing surveillance equipment. With increasingly sized surveillance operations being deployed, smart surveillance operations must be able to scale to thousands of input streams. The most suitable approach to this is through the use of Cloud computing, enabling dynamically sized hardware to be deployed based on the resources required by the system.

This dissertation proposes an extendable and scalable pipeline that is able to provide an end-to-end analytics solution for smart video surveillance. The analytics pipeline is built on core technologies that are reliable, scalable and open source. Leveraging; OpenCV, Apache Kafka, Apache Flink, Apache Spark, Neo4J and Terraform, the pipeline provides object detection and tracking, scalability, activity detection, event classification, data representation and system deployment respectively.

The design approach taken allows for extensibility at every opportunity, so the pipeline can be adapted for a multitude of use cases, with an implemented use case based on the Abbey Road crossing in London shown within this dissertation. Finally, the pipeline will be hosted publicly allowing new applications to be explored by the community, with avenues of exploration suggested at the end of this dissertation.

Declaration

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

Acknowledgments

I would like to thank Matt Collison for his constant support and time. He has continually gone above and beyond expectations to drive this dissertation project forward. I would further like to thank BJSS for investing in me during my placement year, with special thanks to Matt Smith, Ricky Barefield and Steve Rothwell.

Finally I would like to acknowledge my Dad, Dave Honour, who has always been my role model and inspiration.

Table of Contents

1. Introduction

1.1 Motivation

In the United Kingdom, it has been estimated that over 1.85 million surveillance cameras are currently in operation, with each person being caught on camera an average of 68 times per day [1]. Fueling the mass deployment of surveillance equipment is its capability to deter criminal activity [2] coupled with its enablement of event causality [3]. As a result of mass deployment, CCTV is estimated to have prevented hundreds of thousands of crimes per year [4], providing evidence for thousands of criminal investigations, including being available in 96% of homicide investigations [5]. However, with approximately 5,000 years of footage produced every day it has also been estimated that CCTV is only used in 0.5% of crimes that are recorded [2], [6], and while physical CCTV cameras are an active deterrent to criminal activities, there remains a limited approach to responding to the data produced.

Video surveillance is traditionally monitored in one of two ways:

1. Passively, where it is only investigated in response to an event.
2. Actively, where it is viewed live by a single, or team of, people.

Active monitoring and video investigation are time consuming processes, which enable a multi-million pound industry to form around them [7]. As the chance of an event of interest occurring within a video stream decreases, users tend to adopt a more passive approach, which has led to developments that bridge the gap between passive and active monitoring. Sensor driven active monitoring has produced a more efficient approach to video analysis, where movement sensors or external stimuli trigger a prompt for active monitoring of footage [8]. Within an industrial setting, dedicated hardware has been integrated with sensor-driven active

monitoring with increasingly more sophisticated computer vision being used to trigger active monitoring [9].

With the development of Smart Cities, the capacity of smart video stream analysis needs to scale to data volumes never encountered before within this domain. Surat, India is the fourth fastest growing city in the world, with a current population of 5.5 million. Monitoring the city, is over 6,000 CCTV cameras, streaming live video to a single command and control center [10]. For Surat to fully utilize the data generated from its CCTV infrastructure, it needs to be able to autonomously process live video footage, among other sensor readings, to gain real-time insights that can aid in its utilization of public services.

Large scale applications bring with them new considerations; minimizing data transfer and distributing work to enable greater processing throughput, while providing analysis results in a quick enough time to be valuable. To support future capacities seen within Smart Cities, and existing video surveillance deployments, a base analytics pipeline is proposed and implemented. It provides common functionality for video processing by default, while allowing extensions to be developed, enabling its adoption in the widest variety of use cases. Furthermore, the adoption of distributed technologies provides a base to meet the demands of even the largest surveillance networks.

The pipeline can then undergo a twofold evaluation. Firstly, a use case evaluation can be conducted to show the legitimacy of the pipeline to meet bespoke surveillance operations, while documenting the development experience. Secondly, a stakeholder with extensive experience within video surveillance operations can give feedback in order to show the scope in which the pipeline could impact current surveillance operations, along with the benefits its deployment would bring.

1.2 Aim

Develop an extendable and scalable analytics pipeline for video stream analysis with object, activity and event classification.

1.3 Objectives

1. To investigate existing academic literature alongside current video processing techniques and available software packages, in order to support the creation of a scalable analytics pipeline.
2. Present and develop an analytics pipeline that provides a minimum viable product of object, activity, and event detection, while being scalable and extensible.
3. Develop a use case that will allow the evaluation of the real-time video processing pipeline.
4. Using the use case defined, evaluate the pipelines ability to support existing video processing techniques, while meeting bespoke user requirements.
5. Evaluate the pipelines applicability to current projects being conducted by the United Kingdom Police Force, using a directed interview to a viable stakeholder.

2. Background and Literature Review

Computer vision applications are often deployed with a variety of desired outcomes, including facial recognition [11], [12], optical character recognition (OCR) [13], medical image analysis [14], automatic number plate recognition (AMPR) [15] and traffic analysis [16]. Within computer vision domains, analysis is usually implemented with a pipelined architecture, with a focus on pattern recognition.

2.1 Video Processing Methodologies and their Adoption

The exploration of theoretical stages within generic video processing are well documented. Ko et al [17] provides an industrial case setting, with the goal of providing surveillance information to government agencies. This can be used as a basis for common functionality required of smart surveillance systems. Within this setting, functionality can be seen to include background modelling, object segmentation, object classification and object tracking, with the outcome of person identification alongside behavior and activity analysis (Figure 1). This provides an avenue of interrogation within the realm of smart surveillance systems. An alternative approach is to use black box models for pattern recognition, such as deep learning techniques. These models are able to integrate multiple phases of the pipeline described with regards to providing an accurate classification model [18]. Typically, modern video analysis pipelines adopt a combined approach that use black box models for object detection, object tracking, behavior analysis and event classification but allows for limited interrogation of the intermediate steps.

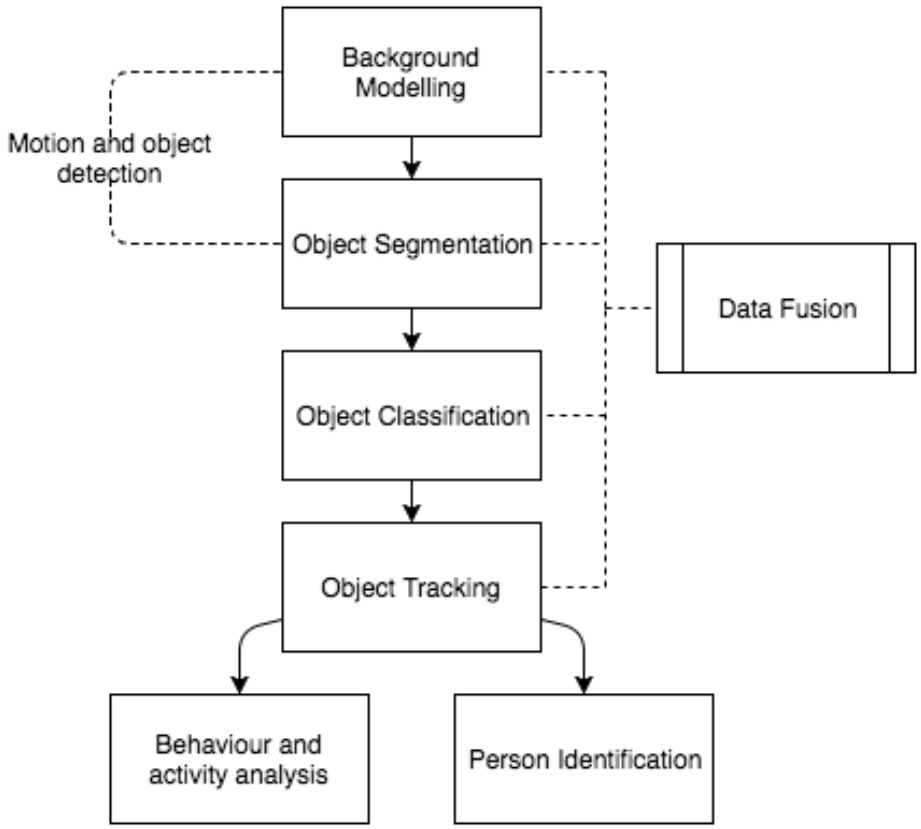

Figure 1: Adaptation of video processing within a general analytics pipeline for automated visual surveillance system [17]. This shows common stages involved in providing computer vision applications in the context of automated surveillance systems.

2.1.1 Object Detection is the base of all computer vision applications, providing the ability to accurately identify objects within a frame. Work within this field has relied upon being able to interpret the combinations of pixels correctly to identify an object. The most common method of doing this is using a Haar feature-based cascade classifier [19]. This machine learning approach works by showing a classifier a multitude of images, with some containing the object you wish to detect. Then, by applying features to the image that allow accurate pattern recognition of the object, we can train the classifier to identify the unique signature of the desired

object when it is present. From here, given a trained classifier, we extract the features from the images that identify the object, 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). The composition of the identifying features of an object enable the final model to identify if, and where, an object is present to a degree of accuracy.

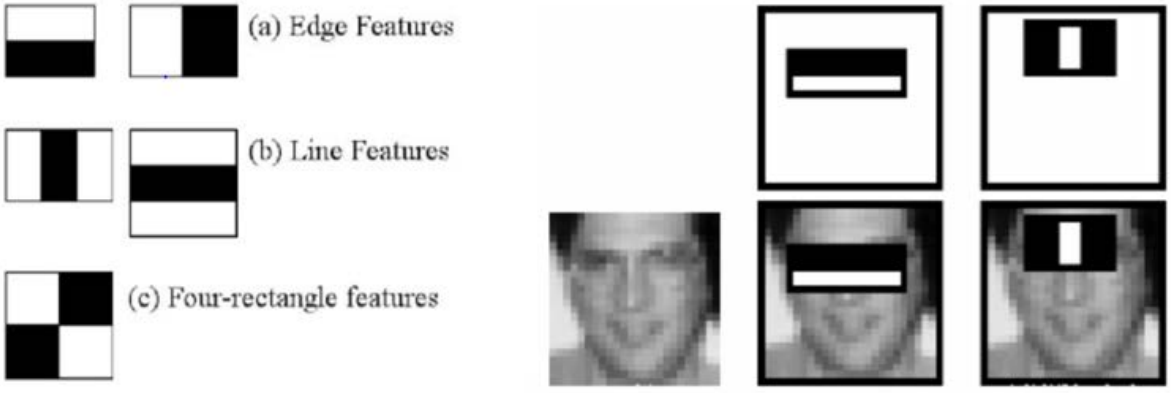

Figure 2: Left, an example of Haar cascade features. Right, the adoption of Haar cascade features in detecting a face [20].

Further to this, we are able to improve the performance of detections using an algorithm called Adaboost [21], which allows ranking of object detection features based on their error rate when attempting to successfully identify the object.

In more recent years, with the improvement of Graphics Processing Units (GPU) hardware, advanced detection techniques have been created through Deep Learning models [18]. Accompanying this, object detection techniques often utilize motion and background separation methods, enabling them to provide insight into events and activities occurring [22]. Utilizing the foundation of accurate object detection, objects can be given identity between frames.

2.1.2 Object Tracking Techniques enable the observation of object movement vectors, along with the monitoring of object interaction patterns. Work within the object tracking domain focuses on the building of predictive models with online data, meaning they adapt at runtime to provide improved predictions. Algorithms within this domain often model an objects location as a set of positions that each could contain the objects location based on its previous known location [23], [24], [25], with the most common tracking techniques described in Table 1. Given successful employment of object tracking, we can now observe the behaviors of individual objects through a video stream.

Approach	Overview	Pros	Cons
Multiple Instance Learning	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 [23].	Works well when object partially occluded due to its bag representation.	The tracker cannot handle full occlusion of objects well.
Median Flow	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 [24].	Works well when an objects trajectory is predictable.	The tracker becomes less reliable under fast or unpredictable motion.
Tracking-Learning-Detection	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 element estimates the error of the detector and updates it to avoid these errors in future [25].	Works well under long periods of occlusion.	The tracker does not perform well under large full rotations and can frequently create false positives.

Table 1: A comparison of three common approaches to object tracking. This aims to show the diversity in object tracking, while illuminating the tradeoffs of each approach.

2.1.3 Behavior and Activity Analysis allow the assignment of context to object movements, enabling more advanced processing techniques downstream in the video processing pipeline, such as event classification. Within computer surveillance it is often desirable to identify endangering or suspicious activities [17]. Active work within this field is varied, with successful modelling of behaviors being produced using Markov Models [26], [27], along with the more computationally intense techniques architected with Neural Networks [17]. Markov Model techniques have identified behaviors to precision rates above the 90^{th} percentile [28], however they struggle with noise in the data which can cause accuracy to drop. Other popular methods of behavior analysis include [17]:

- Dynamic Time Warping: a technique for comparing the similarity between two sequences of events. This allows the calculation of the probability that a shown behavior corresponds to a previously known behavior pattern.
- Finite State Machines: models' behavior as a finite set of states, with transitions between related states. This allows for analysis of real-time video, watching for movements that trigger a tracked object to progress to a different state. A comparison between the transition path through the state machine and known behavior traits, enables the identifying of witnessed behaviors.

The understanding of activities within a video stream provides a deep insight into what is transpiring, allowing the modelling of the video context through time.

2.1.4 Event Classification builds upon activity analysis, giving a method for identifying unusual data points within the context of the video stream. Work in this field focuses on building a model to represent the current state of the data, and then compares new data points to the

model, calculating how far each point deviates from the existing data set. Multiple models are able to provide this, frequently built around clustering techniques [29].

Further to this, the adaptation of One Class Support Vector Machines have provided anomaly detection over short-term observed behaviors within video streams [28]. They are able to detect whether a short-term series of points is outside of normal observed behaviors, offering an abnormal threshold capability that can distinguish which data points require more in-depth anomaly analysis.

Recently, with the wide adoption of Neural Networks, many methods have emerged that are built around this core architecture providing high accuracy detections, at the cost of requiring high performance hardware [30]. Based around learning the distribution of features within the data, the models can detect when a data point falls outside of the normal distribution and classify it as anomalous. A Restricted Boltzmann Machine is a Neural Network model for the learning of distributions and, when multiple are stacked together, they build a Deep Belief Network [30]. However, Deep Belief Networks bring with them a large computational overhead and therefore may not be appropriate for a generic video processing pipeline.

2.2 Distributed Computing and the Cloud

Video stream data is rich and complex and therefore raises significant challenges in data handling and processing when attempting to retain real-time requirements. In order to provide actionable intelligence analysis not only has to be functionally correct but must also be produced within a time frame that allows significant actions to be taken.

2.2.1 Parallel Computing allows the concurrent processing of data streams, enabling simultaneous analysis over large quantities of data. This is often measured by system throughput; a measure of the number of items able to pass through a system concurrently. Within the realm of

video processing, there is a need to perform large amounts of calculations on individual frames, in order to provide services ranging from initial object detection to tracking and analysis. As more video streams are added to a system for analysis, there is a need for these expensive operations to scale, therefore we must be able perform these operations in parallel over a distributed set of machines. A multitude of openly available framework exist that can be leveraged to vastly increase the throughput of a proposed analytics pipeline (Table 2).

Framework	Functional Details
Apache Storm	Apache Storm runs on a distributed cluster of machines, where a topology is defined representing the series of processing stages to apply to each event. A topology is comprised of bolts, with a bolt representing an individual processing stage on an event. This architecture gives Apache Storm power to process billions of events across a distributed set of machines in a single day [31].
Apache Flink	Apache Flink works by expressing data processing as a series of tasks to apply to each event as a directed graph. The sending of messages between stages in the dataflow pipeline make use of buffers, with the backpressure from these buffers being used to control the throughput of the pipeline. This allows Apache Flink to provide throughputs of up to 80 million events per second [32].
Apache Spark	Apache Spark was originally designed for distributed batch processing but has since been extended for stream processing. It processes events in small micro-batches in order to run them in real-time. It represents data as a resilient distributed dataset that can have operations applied to it [33].

Table 2: An overview of popular available frameworks for distributed computing, showing a brief insight into the architecture of each framework.

As these technologies offer similar services, performance comparisons between them exist. Under high workloads, it has been found that Apache Flink out performs other systems with regards to latency, however Apache Spark is able to provide the largest processing throughput [34]. Nevertheless, these systems are in their infancy of development and therefore a full conclusion about the final performance of each system are hard to make.

2.2.2 Distributed Messaging provides the ability to decouple the different stages of processing, allowing for the addition of new features without effecting existing stages of a processing pipeline. As well as decoupling, video processing analysis relies on creating an immutable log of objects, behaviors and anomalies detected within a given video stream. This immutable log enables the understanding of event causality; which is a core component of video processing analytics. Further to this, a distributed messaging layer should be able to scale to multiple video stream inputs without effecting real-time performance requirements. Providing these capabilities is Apache Kafka [35].

Apache Kafka is a distributed streaming platform, which lets applications publish and subscribe to topics of messages. It stores the messages published in a fault tolerant way, partitioning information across multiple machines within a configured cluster. To communicate with Apache Kafka a client performs a high-performance, language agnostic, TCP protocol, meaning any system can speak with it. Further to its core functionality, Apache Kafka is also extremely fast, outperforming traditional messaging systems when handling the producing and consuming of messages between decoupled systems (Figure 5).

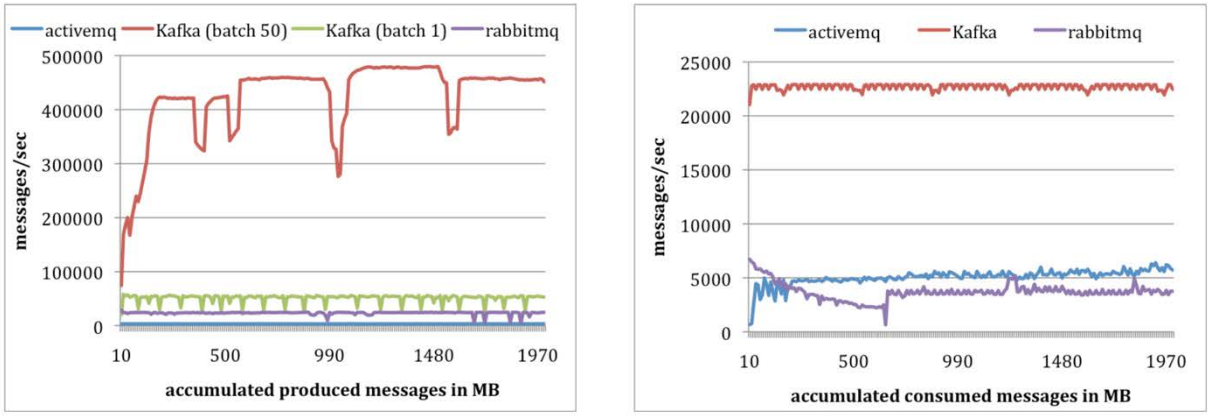

Figure 5: A comparison of Apache Kafka's producer performance (Left), and its consumer performance (Right) against competitor messaging systems [36].

2.2.3 Cloud Computing enables a user to rent the computational power they require, scaling up when demand is high, and reducing when systems become idle. This elasticity is a highly desirable feature for an analytics pipeline that needs to cater to a large variety of projects and requirements. This enables the distributed processing systems discussed to have resources allocated to them at runtime, when demand is high on an individual stage of the pipeline more machines can be created to meet non-functional requirements. Further to this, machines can be destroyed if they are not required, reducing running costs of the system.

The benefits of Cloud computing do not come without risks however; with the virtualization of hardware exact performance metrics may not be known until runtime. As a client of a Cloud provider, you are also at the mercy of their availability; if their service becomes unavailable, you are incapable of fixing it yourself. Further considerations to make can include, data confidentiality, software licensing costs and runtime renting costs [37].

2.3 Existing Technologies and Approaches

Computer vision is often most successful in batch analysis with extensive post processing, however smart video surveillance requires real-time event classification. This relies upon the computer assuming the active monitoring role rather than to prompt a human to analyses the footage, which is beyond the scope of most existing applications. Table 3 provides a summary of existing applications that supply a variety of analytics over video streams. However, these applications are often designed for an individual use case or developed under closed licensing and therefore cannot be built upon.

Existing Work	Functionality Details
Nest [9]	Alerts users in real-time, via an app, to event and motion detection events seen on camera. The service provides event and anomaly detection, alongside real-time alerts, triggered by the presence of anomalous behavior. However, the application requires specialized hardware and is not extendable for development.
A Video Analysis Framework for Surveillance System [38]	Gives a novel framework approach to online video analysis using .NET 2.0. Provides object and event detection, with extensibility to insert new functionality within the application. It is limited by its lack of distributed computing; therefore, it will not be able to scale to all user requirements. Further to this, the product is developed with .NET, requiring a Microsoft workstation to run alongside purchased licenses.
DiVA [39]	Gives a distributed video processing framework that uses a database as a message source, allowing components to communicate agnostic of technologies adopted. Provides object detection with the goal of detecting object abandonment and removal. Extensibility is provided through the communication of modules/algorithms occurring at a database level. However, it uses a singular database to communicate, rather than a distributed messaging system. It also requires fixed hardware meaning it currently does not make use of Cloud Computing.

Table 3: A list of current technologies in the market that aim to provide smart CCTV.

2.4 Existing Data Pipelines

Although not in the domain of computer vision, scalable analytics pipelines have been used for numerous other applications, including network anomaly detection, log aggregation and analysis. Table 4 provides a summary of data analytics pipelines that our proposed computer vision pipeline will be able to leverage in order to provide real-time analytics. This existing work allows great insight into tool adoption for the stages of our video analytics pipeline and proves the technologies ability to work at scale within a variety of domains.

Existing Work	Functionality Details
Real-Time Network Anomaly Detection System Using Machine Learning [40]	The paper proposes a framework based around an Apache Kafka streaming architecture. It makes use of Apache Storm for event processing with HDFS as its distributed data store. It is therefore able to detect real-time anomalies in network traffic across the entire University of Missouri Kansas City network.
Twitter Heron: Stream Processing at Scale [41]	The paper provides an insight into stream processing at Twitter. They show how they previously leveraged Apache Storm and have now presented a new tool, Heron, for event processing. The paper shows the scale at which a stream processing-based architecture is able to perform operations on data, and further shows the adoption of Apache Kafka as a message broker.
Evaluating New Approaches of Big Data Analytics Frameworks [42]	This paper looks at the overall field of big data analytics, and particularly mentions and compares two common processing frameworks; Apache Spark and Apache Flink. The paper shows their ability to perform specific operations over millions of data points and presents conclusions between the two frameworks as to which perform better at certain tasks.

Table 4: A list of existing technologies within the real-time analytics landscape, showing the approaches taken to processing data in real-time at scale. This aids in finding an appropriate technology toolchain capable of applying data analytics within the video processing domain.

3. Proposed Analytics Pipeline

Building on the work previously seen within the realm of video processing, coupled with the work seen in other domains where scalable real-time analytics have successfully been implemented (Table 4), this dissertation proposes an analytics pipeline architecture and implementation for real-time scalable video analysis.

3.1 System Architecture

Proposed is a scalable analytics pipeline that supports common video processing techniques by default, while being open for extension to enable domain specific modifications. The pipeline processes the raw video footage as close as possible to the video source, annotating objects of interest. Annotations are then sent to the distributed messaging layer, enabling further information to be extrapolated, while reducing the amount of data transferred for processing. Once data is within the messaging layer, downstream processing stages attempt to classify events occurring within the video feed and identify events that deviate from the norm (Figure 6).

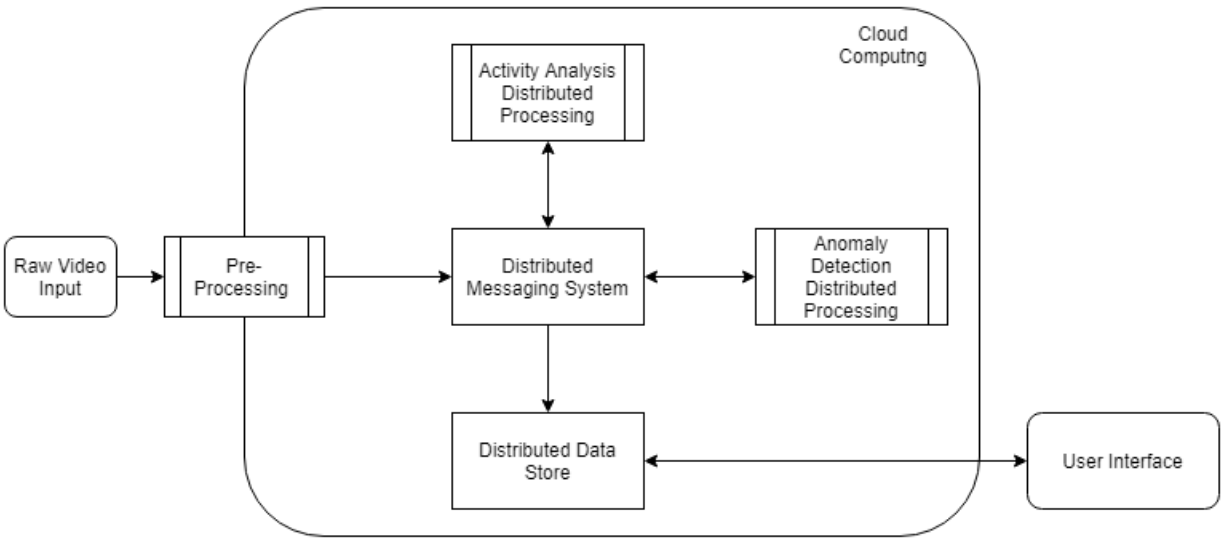

Figure 6: The proposed video processing architecture, showing how communication between systems is achieved through the distributed messaging system.

The pipeline adopts a streaming architecture to provide real-time analytics on data generated from the raw video input (Figure 7). Streaming based technologies allow for operations to be applied to individual events as they occur, rather than periodically querying data that is generated. This provides the benefit of real-time feedback as video footage is transmitted. However, this does not limit traditional batch processing, as the data is still streamed to a data store as it's produced by the analytics services. Thus, offline querying that may be too time consuming or expensive to be performed in real-time, is still supported within the pipeline. In order to achieve this the pipeline design is modular, with modules subscribing to data streams within the distributed messaging system, allowing for compartmentalized operations to be performed agnostic of other systems acting on the same stream.

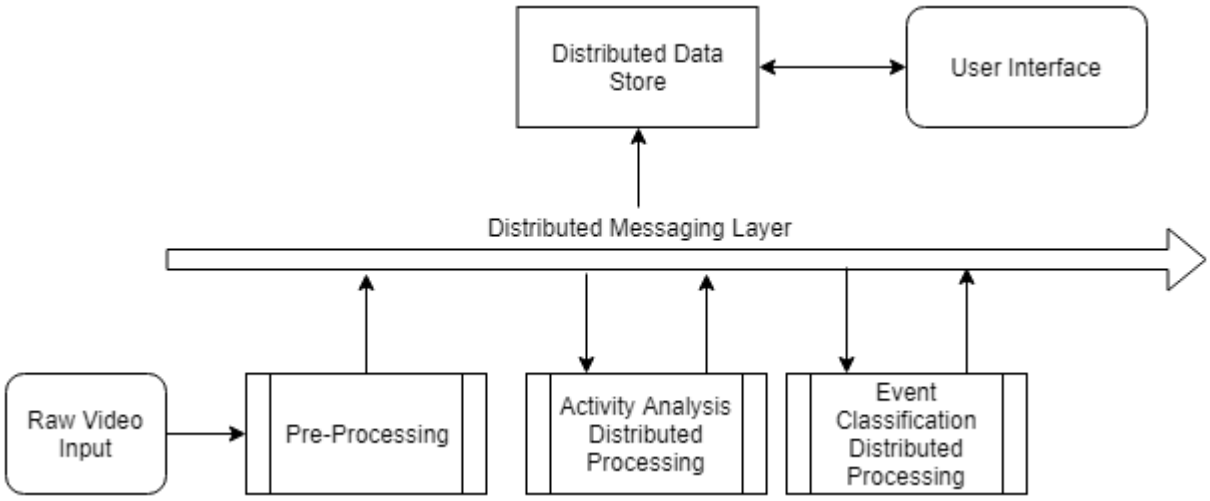

Figure 7: The proposed video processing architecture, showing the flow of data through the pipeline. Raw video input enters the preprocessing stage, which extracts all meta-information required for downstream processes to operate. All downstream processes then listen for information through the distributed messaging layer.

3.1.1 Pre-Processing of the raw video input allows for fine-grained control of the volume of data entering the downstream processing stages of the pipeline. This is the first stage of the pipeline and the interfacing component between existing camera hardware and the deployed analytics pipeline. The module is responsible for annotating the video, extracting all necessary information from each video frame, which can then be entered into the analytics pipeline through the messaging system. Using a technique known as Edge Computing [43], filtering and detection is applied on the raw video feed to control the amount of data entering the downstream processing stages. This allows for the full utilization of the networks capacity through the reduction of redundant data transfer between systems. The messaging system therefore does not need to support the streaming of live video, but instead accepts the meta-information captured from pre-processing, enabling downstream analytics while massively reducing the load on the messaging layer. Thus, this increases the throughput of the pipeline as less unnecessary data arrives at the messaging system for processing. This further allows the quality of data to be maintained within downstream processing services, as degraded or faulty readings can be determined before they have a chance to reach the core analytics stages of the pipeline.

3.1.2 Activity Analysis is then performed on the annotated data published to the distributed messaging system by the video pre-processing service. This is done over a distributed processing framework, allowing for parallel identification of activities. Making use of pattern matching techniques, the service consumes the stream of video annotations produced by the pre-processing stage and compares the processed events with its internal model of activity signatures. When an activity is identified it can then be published to the messaging service for downstream services to analyze.

3.1.3 Event Analysis then engages with the information produced from upstream stages to identify clusters of behavior, looking for events that deviate from the norm. This is supported through the use of an unsupervised cluster detection model by default. This enables a user to use event classification techniques without understanding or building complex reasoning models. However, as the service is extendable and modular it can further support complex reasoning models but does not supply them, as they cannot be made generic for all possible use cases. If an event of interest occurs the service has the capacity to send notifications to the appropriate user, allowing for proactive investigation to occur.

3.1.4 Data Storage occurs as the data is streamed between services. The service consumes all messages being sent over the distributed messaging layer and persists them to a database. Performed in real-time, the service aims to provide data interfacing capabilities, allowing further services to query the data flowing through the entire pipeline at all stages, and semantically reason about it. This enables offline analysis of the data by users, along with querying and exploration of data produced by the pipeline, agnostic of data producing services.

3.1.5 Sub System Communication is completed through a distributed messaging layer, decoupling systems and maintaining a flexible and extensible application. Communication occurs through a common Application Program Interface (API) between each service and the messaging layer. The decoupled nature of each sub system means they can be deployed independently, allowing the most appropriate tool to be used for each area of processing, with individual resource allocations for different parts of the system. This enables users to deploy infrastructure on a per-service basis, giving a fine-grained level of control to avoid over, or under, allocation of resources to a task. A messaging layer adds complexity to the analytics pipeline, as systems have the added overhead of indirectly communicating with each other,

which is a noticeable cost when adopting this architecture. Nevertheless, the ability to have technology agnostic communication between systems outweighs the complexity overhead associated with messaging services.

3.1.6 Cloud Based Architecture provides an ability to rent hardware rather than buying it upfront, giving a flexible way of managing infrastructure depending on specific performance requirements. Due to individual processing services requiring deployment to a distributed set of machines, enabling throughput and latency targets to be met, a Cloud based architecture has been chosen. This allows for scaling to occur through the dynamic allocation of machines to a service, in order to meet all non-functional requirements. Though, the deployment can be made to a local environment, allowing considerations to be made as to the most appropriate production infrastructure on a per use case basis [37].

3.1.7 Improvements to previously seen work [38] [39] have been made through the use of Cloud Computing infrastructure, enabling clusters of machines to distribute work for a single task, rather than just distributing work between different stages of the processing pipeline. The analytics pipeline identifies the success of existing work; taking a modular approach to design, communicating through a shared messaging layer, allowing extensibility to meet individual requirements. This approach hopes to overcome the challenges of scaling, while maintaining the success of modular design seen in previous work, enabling large scale computer vision applications to become possible.

3.1.8 Limitations can be found with a distributed based approach; the network latency of the infrastructure heavily affects distributed processing, as each node in a cluster must communicate with its counterparts to organize and distribute work. This can drastically reduce the performance of the proposed pipeline if deployed onto a degraded network environment.

Mitigating this, the analytics pipelines deployment to Cloud infrastructure allows for dynamic network configuration to meet individual requirements, coupled with intense processing happening at the edge of the Cloud. This reduces the overall network load throughout the pipeline allowing the underlying networking hardware to be fully optimized.

Further to this, Cloud infrastructure can become expensive as network usage is charged to the user along with the rented computing power of the machines. To combat this, the pipeline is not linked to a single Cloud provider, or to the Cloud at all, giving the freedom of choice to the adopting user.

3.2 System Implementation

The system architecture is implemented as shown in Figure 8, constructing the data processing pipeline in Figure 9. The sub systems are written independently and do not share dependencies, allowing them to be modified and extended without effecting the development of other services. Accompanying this, support projects have been created allowing the automated deployment of all required distributed processing frameworks to Amazon Web Services (AWS).

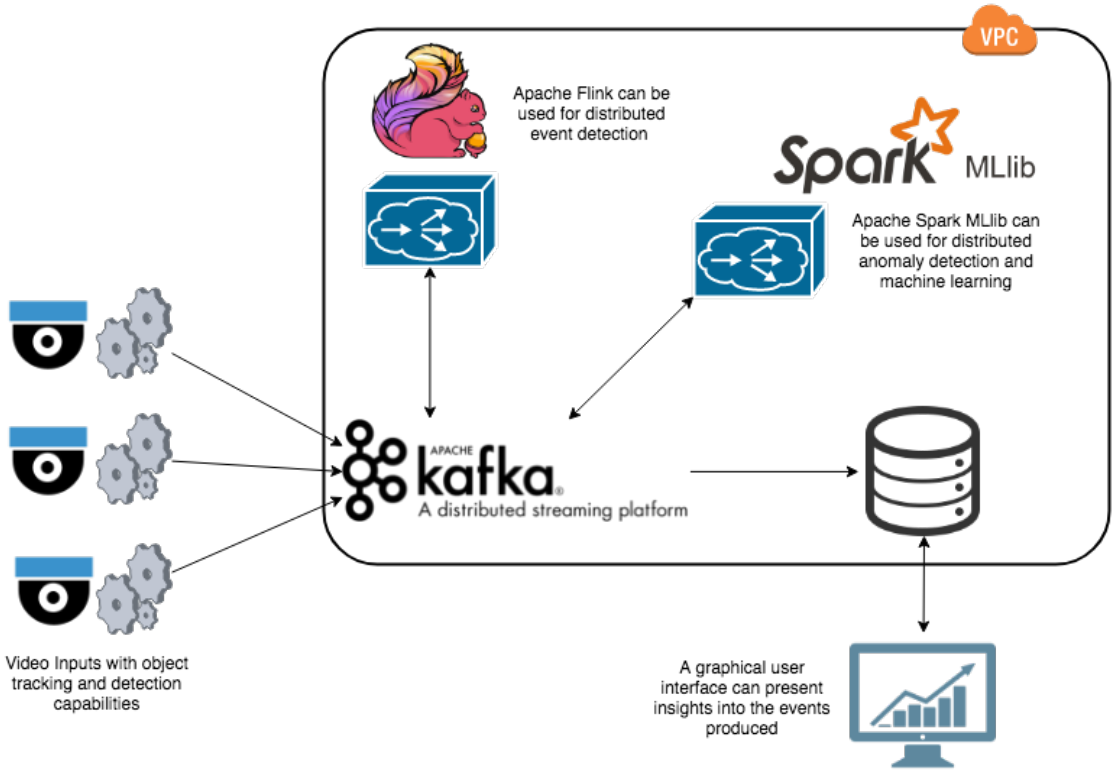

Figure 8: The implemented video processing pipeline, distributed computing technologies (Apache Kafka, Apache Flink, Apache Spark) are adopted to provide high throughput, low latency, processing within a Cloud environment.

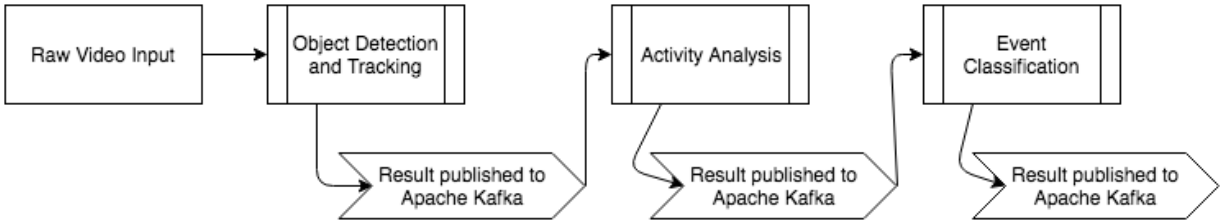

Figure 9: The data flow through the application, showing how each processing stage makes use of Apache Kafka to read input and produce outputs.

3.2.1 Sub-System Communication is provided using Apache Kafka. Apache Kafka [35] is chosen due to its ability to scale to support millions of messages per second, along with its low latency [36]. This enables projects of any size to adopt the analytics pipeline and allows the pipeline room to scale as demand increases (Figure 10, Figure 11). This data displays how Apache Kafka supports hundreds of thousands of messages per second, even when configured to sacrifice asynchronous performance for message delivery guarantees. This enables data integrity assurances to be met while continuing to meet real-time requirements of video processing systems. Further to this, Apache Kafka can act as an efficient buffer between systems, allowing for asynchronous communication, reducing time spent waiting for message responses. Buffering also allows slower downstream processes to not impede the processing progress of upstream processes, which are able to operate at a faster rate. This enables the implementation of computationally expensive processing stages without degrading the throughput capacity of the pipeline, as Apache Kafka handles back pressure from any processing stage. Apache Kafka is implemented through its deployment as a standalone system within the pipeline, making use of Terraform cloud deployment tools discussed within 3.2.8. Once deployed, the individual processing stages are able to publish and subscribe to their appropriate message queues, enabling the input and output streaming of data. With Apache Kafka at the heart of the video processing pipeline, the individual video processing components are able to communicate with efficient buffering.

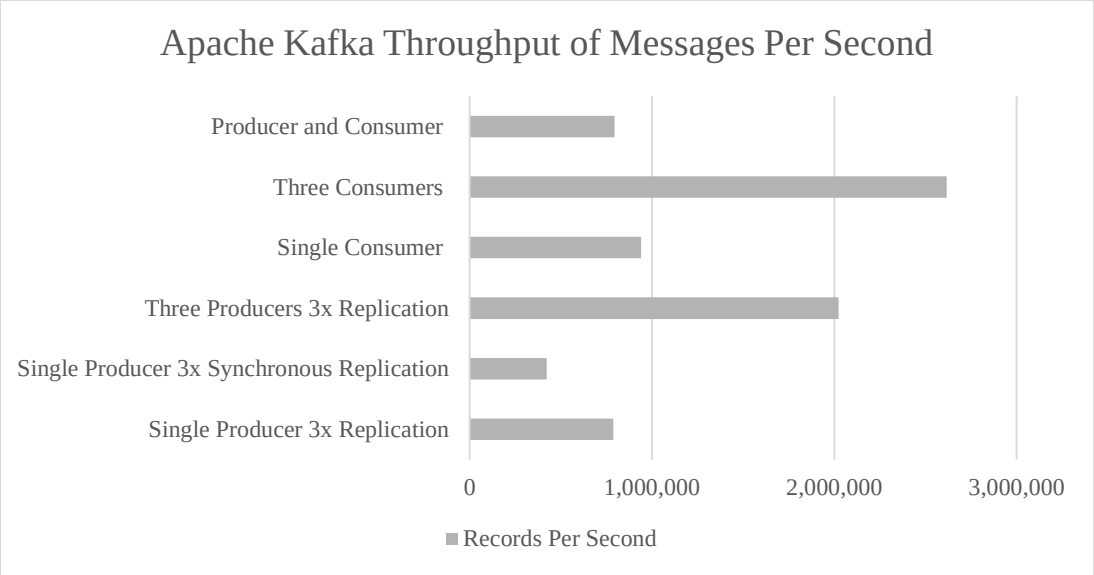

Figure 10: Apache Kafka throughput in messages per second [44].

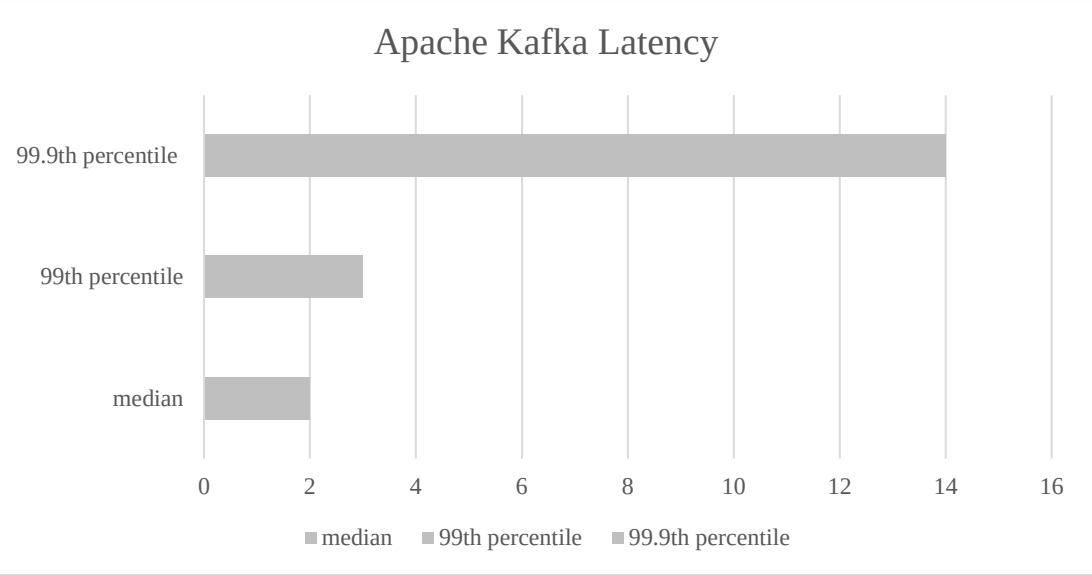

Figure 11: Apache Kafka latency in milliseconds [44].

3.2.2 Video Pre-Processing is implemented as a Python package offering configurable video processing techniques for object detection and tracking. Its responsibility is to process the raw video input stream and convert it to video annotations that are consumable by downstream

processing services. The module is built around a processing pipeline interface, allowing the user to choose which processing techniques are applied to each frame of the video input, and where the results of the processing stages are sent. Video inputs can be accepted from file and from a directly connected USB camera (Figure 12). Further to this, person and car detection is provided using the OpenCV library, with tracking of all detected objects enabled through a Kernalized Correlation Filters model. This dynamic configuration allows for the package to be extended and configured easily to individual domains. The output of the pre-processing stage is sent to Apache Kafka by default, with the annotated video shown on screen. This aids in the visualization of generated data through the viewing of the produced annotations being sent for downstream processing.

```
PIPELINE = \
    PIPELINE.with_frame_processing_stage(ResizeFrameProcessor(width=500, height=500))\
            .with_video_processing_stage(Tracker([
                    PersonDetector(),
                    CarDetector(
                        car_cascade_src="video_processing/detection_models/car_cascade.xml")]))\
            .with_video_output_stage(LocalDisplayVideoOutput())\
            .with_processing_stopping_criteria(QuitButtonPressedStoppingCriteria())

if ARGUMENTS.kafkaurl is not None and ARGUMENTS.kafkatopic is not None:
    PIPELINE = PIPELINE.with_message_sender(ApacheKafkaMessageSender(
        server_address=ARGUMENTS.kafkaurl.split(","), topic=ARGUMENTS.kafkatopic))

start_application(PIPELINE.build())
```

Figure 12: An example configuration of the video pre-processing package. This displays how the use of a builder pattern can be adopted to enable easy building of custom processing stages.

3.2.3 Activity Analysis is delivered as an Apache Flink task, which can be executed on any deployed Apache Flink cluster. The implemented task is able to apply its processing

operations over all the machines in the cluster, allowing it to operate on multiple input records concurrently. The task offers distributed activity detections for standing, walking and running people, along with the detection of parked and moving cars. Reading the data produced from the pre-processing service via Apache Kafka, the service adopts Apache Flink's advanced pattern matching technology to look for simultaneous movement events typical of known activity behaviors. The pattern matching library observes a stream and is able to provide data windowing functionality, in which it can detect a defined pattern of events. The service is tuned through configuration files read in at task runtime, allowing for activity identification to be tuned to provide precise results for particular camera installations and viewing angles. Furthermore, the service is made extendable through the Flink pattern matching API, and new patterns can be developed to allow the identification of new behaviors.

3.2.4 Event Classification adopts the core libraries within the Apache Spark ML library to provide unsupervised event classification. The service reads data produced from the activity analysis service and then uses an unsupervised K-Means clustering model to assign it a cluster of origin and calculate how far the instance deviates from the cluster's center. This model is adaptive, as the data is streamed through the system it is able to adjust cluster locations to provide accurate classifications even if data patterns change. The service is tunable through command line parameters, enabling the configuration of the number of clusters the model should attempt to identify. Once it calculates the events cluster and deviation, it publishes the information to Apache Kafka. This is implemented as an Apache Spark job and can therefore be submitted to any deployed Apache Spark architecture, allowing it to make use of distributed processing techniques. Apache Spark provides data streaming in the form of micro batch

processing, adding an artificial latency to processing, which should be considered if further downstream processes require the output of this service in a low latency manor.

3.2.5 Event Notifications are enabled through a task running on Apache Flink, reading data produced from the event classification service via Apache Kafka. Notifications can then be sent for events assigned to a specific cluster, or for events that deviate from their assigned cluster by a set amount. These rules can be configured at runtime through configuration files, allowing for dynamic business domain rules to be created and changed as more insight into data is achieved. If an event is deemed to require a notification, the current system supports the sending of an email which contains the events unique ID, along with its assigned cluster and deviation score (Figure 18). This enables the alerting of invested parties with the necessary information required for further investigation of an event, giving them the tools to act on events of interest.

3.2.6 Data Storage is achieved through an Apache Flink task that consumes messages from all Apache Kafka topics, writing them to a database. As each service publishes to its output topic, the data storage task consumes the messages along with the other downstream processes. Neo4J is the chosen database by default, offering clustered storage with fast access, scaling to large datasets. The storage service then, for each message, is able to create a node within the Neo4J database instance. Once a node has been created, the data storage service creates relationships between nodes to show the flow of data relations (Figure 13). This allows for easy data interpretation and semantical reasoning, as the data is modelled through its relationships.

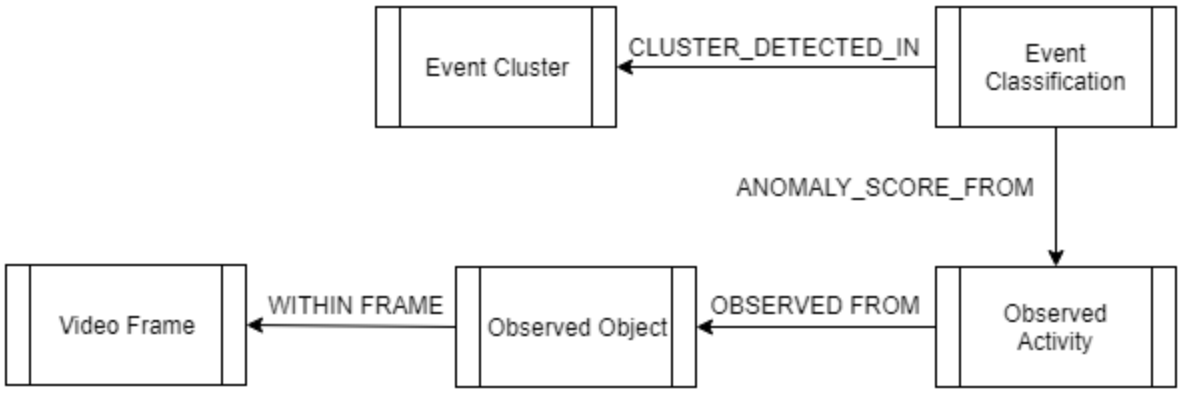

Figure 13: The relationships between entities created within Neo4J. An object can be seen within many video frames, while an activity can be observed over many objects. Each activity has an event classification, which is assigned to an event cluster.

3.2.7 Data Interfacing is provided through the Neo4J web interface. This interface provides easy tools for data querying and exploration, accompanied by a visualization tool (Figure 14). This was chosen due to its ease of use, and the fact Neo4J is already being used as the data storage provider. The interface is made available through a website, offering functionality to export query results and capture images of the data represented as a graph. This allows for the data to be quickly interpreted by non-technical users, as well as supporting complex graph querying algorithms, such as shortest path finding, which may allow further business insight into the relationships forming between the stored nodes.

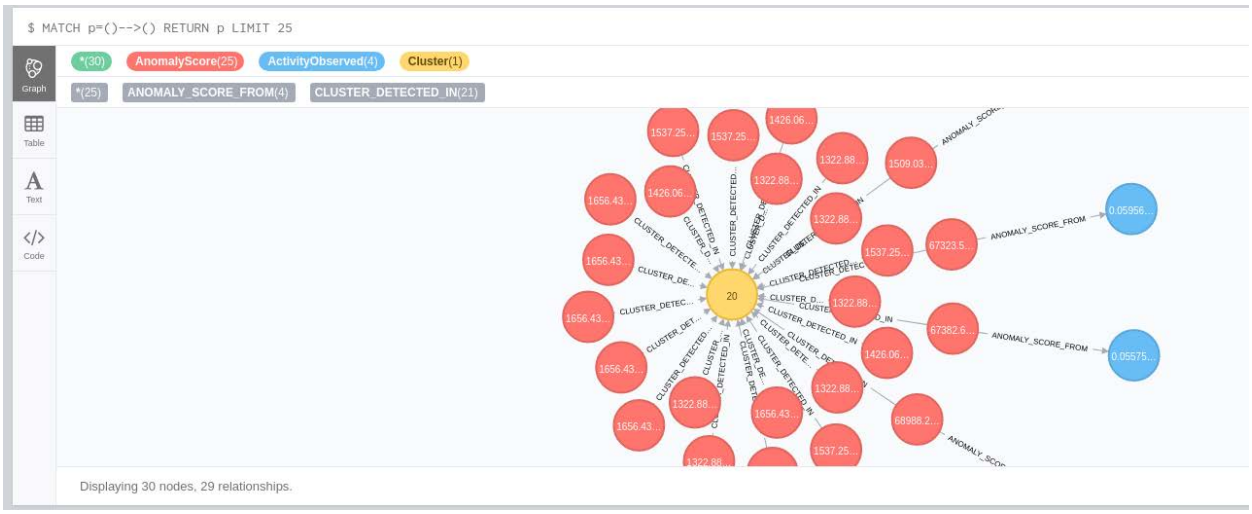

Figure 14: An example query result in the Neo4J dashboard. This shows the clear data visualization capacity of Neo4J. The returned graph is also interactive, so you can move and expand data from your original query, giving you power of exploration.

3.2.8 Infrastructure Deployments are made simple for the configured pipeline through Terraform scripting. Terraform is an ‘Infrastructure as Code’ language, meaning you can define your required deployment of machines as code, executing the configuration to create the desired infrastructure. The base pipeline has scripts to define Apache Kafka, Apache Flink, Apache Spark and Neo4J deployments. These scripts create minimum viable clusters for all distributed technologies (Table 5), supporting scaling through modification. The Terraform scripts deploy all machines to AWS currently, as this is one of the most popular cloud providers and therefore has been chosen as the standard for the out of the box behavior of the analytics pipeline. These scripts have been developed separately to any single required system within the pipeline, and therefore a client who is already running any of the infrastructure is not forced to use this scripting technology. This enables existing clients to adopt the parts of the base pipeline that are required, without forcing the adoption of the entire pipeline. Decoupling in this manor aims to

achieve a modular adoption process, where a client can configure and deploy services independently, without requiring the pipeline to be completely redeployed if only a single module was updated.

Framework	Deployed Infrastructure
Apache Kafka	3 EC2 t2.micro Ubuntu instances - 2 EC2 instances run Kafka brokers - 1 EC2 instance runs Zookeeper
Apache Flink	3 EC2 t2.micro Ubuntu instances - 2 EC2 instances run a Flink Task Manager - 1 EC2 instance runs a Flink Job Manager
Apache Spark	3 EC2 t2.micro Ubuntu instances - 2 EC2 instances run Spark worker nodes - 1 EC2 instance runs a Spark master node
Neo4J	1 EC2 t2.medium Ubuntu instance - The instance runs the Neo4J database

Table 5: A table to show the deployed AWS infrastructure for each required service that is delivered by the analytics pipeline out of the box.

3.2.9 Testing was primarily focused around the integration of systems through Apache Kafka. As the messaging layer controls the inputs and outputs of all systems, it controls the most critical aspect of the analytics pipeline; the data. In order to test this functionality, a series of test cases were performed on the deployed system (Table 6). This enabled confidence that data flowed through the system successfully and that components were feeding predictable results to downstream processing stages. In future, more advanced integration testing can be developed in order to produce a regression test pack that gives confidence in the pipeline as it is adapted.

Test Case	Test Aim	Result
Deploy the production infrastructure using Terraform scripts.	Prove the production infrastructure is resourced correctly, and that tasks can be submitted for running.	The production infrastructure was successfully deployed and each service, submitted to their respective runtime, started and ran successfully.
Feed test video annotations into Apache Kafka, confirming they are processed by each service in order.	Prove the data flow through the application services is correct; reading and sending information to correct Kafka topics.	The data was processed in the correct order, with each service successfully communicating to the correct Kafka topics.
End-to-End test of data processing over a short pre-configured video clip, from initial meta-information capture to database storage.	Prove the systems are able to interact with each other over realistic data streams and that the data is stored for querying successfully.	The components of the pipeline successfully interacted with each other, producing tangible data. A bug was detected in the data storage service due to a race condition between records being stored from different topics. This was rectified, and the test was re-ran showing the elimination of the bug.

Table 6: Main test cases conducted over the pipeline during development. These aim to give confidence in the pipelines ability to be deployed and provide accessible data streams to configured services.

System Implementation can be found at: https://github.com/Guardian-Development/Finch-VideoProcessingFramework

4. Use Case Implementation

The analysis pipeline implemented as part of this dissertation consists of a combination of core video processing functionality, enabling extension and flexibility to meet a broad range of client use cases. Presented below is a practical use case showing the deployment procedure of the analysis pipeline, enabling the documentation of the development experience and the collection of runtime metrics, in order to gauge the pipelines performance.

4.1 Aim

In order to determine the success of the implemented analytics pipeline, the development experience when adopting the pipeline must be evaluated. In order to enable the pipeline to fulfill its goal of enabling future development projects in the realm of smart surveillance systems, the evaluation of the pipelines extendibility and journey to production must be completed. This aims to show that the pipeline can provide an approachable starting point for future development projects, while reducing the development time spent on creating a real world smart surveillance system.

4.2 Scenario

Figure 15: The Abbey Road video feed. This video footage contains vehicles (Left) and pedestrians (Right), providing a good basis to evaluate the pipelines applicability to current CCTV scenarios.

Abbey Road in London, pictured on the front of the famous Beetles album ‘Abbey Road’, is a busy road crossing and a popular destination for tourists and locals (Figure 15). Due to this, it is under live surveillance at all times, with high volumes of pedestrian and vehicle traffic. This presents an opportunity to detect and track cars and people within the live video stream, infer activities of standing, walking, running, parked and driving, producing real-time insights into events occurring at the crossing. As this is an IP camera deployed to a street in a city, it also aims to show that the services are able to extract and work with video footage created by existing surveillance systems.

4.3 Implementation Procedure

To configure the base analytics pipeline, each service requires tuning in order to work for the chosen business domain.

4.3.1 Enabling Video Pre-Processing is completed by defining an appropriate processing pipeline within the provided python package. For our small use case, the pipeline is configured to take a single video feed input, produced by the deployed surveillance camera. The pipeline standardizes the size of each frame within the video feed, adjusting it to a 400px/400px aspect ratio to improve runtime performance, before applying person and car tracking, sending the tracking information to Apache Kafka for downstream analysis. Along with this, the annotated video is configured to be displayed to screen, allowing easy visualization of the data being sent to downstream processes (Figure 16).

This development configuration was made through the use of the supplied Python classes, editing the main function within the program in order to add the appropriate processing steps to the video processing pipeline. From this single development change, the program was then run through a command line interface within the production infrastructure, with key

connection details being passed in at runtime. This enabled the dynamic redeployment of the system; if the connection details to downstream services changed, restarting the program with the updated connection strings was all that was necessary for the service to modify where it sends its results. Thus, once the pipeline was configured to contain the required processing steps, no further development changes were required, reducing the amount of time spent developing and redeploying the system.

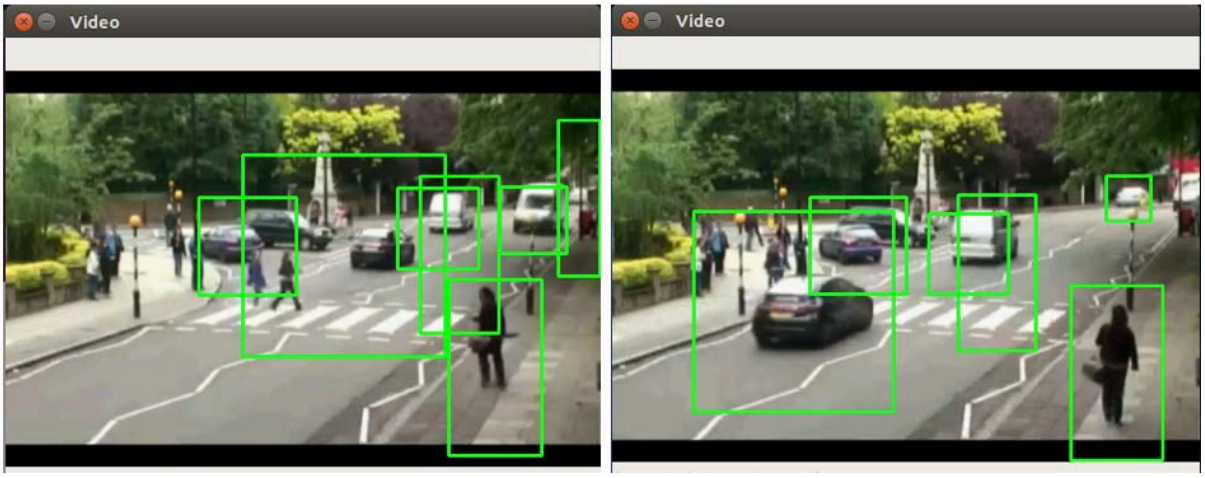

Figure 16: The video output of the deployed video processing pipeline identifying people and cars within the “Abbey Road” real-time video stream.

4.3.2 Enabling Activity Analysis requires little modification to the base pipeline, with the pipeline already supporting standing, walking and running, along with parked, and driving activities. From a development perspective, the activity analysis service therefore did not require any physical code changes, but instead required the updating of configuration files in order to run the activity models to work with the given video feed annotations. As the configuration files do not require compilation, and their location is passed into the service at runtime, the development procedure was streamlined during the deployment of this service. In order to tune the models,

there was a need to collect video annotations typical of the deployed video processing service. With these annotations, the next step was to investigate, through the running of the activity analysis service, which parameter combinations produced the most reliable activity detections (Figure 17). Once this was complete, these configuration files were saved and, when deploying the activity analysis service to production, passed at runtime to be used by the final deployed service.

```
# Standing Detector Properties
detectors.standing.item.type = person
detectors.standing.window.size = 50
detectors.standing.window.slide = 20
detectors.standing.displacement.min = 0.0
detectors.standing.displacement.max = 1.0
detectors.standing.repetition.triggers = 10

# Walking Detector Properties
detectors.walking.item.type = person
detectors.walking.window.size = 50
detectors.walking.window.slide = 10
detectors.walking.displacement.min = 1.0
detectors.walking.displacement.max = 5.0
detectors.walking.repetition.triggers = 10

# Running Detector Properties
detectors.running.item.type = person
detectors.running.window.size = 50
detectors.running.window.slide = 20
detectors.running.displacement.min = 3.0
detectors.running.displacement.max = 10.0
detectors.running.repetition.triggers = 15

# Parked Detector Properties
detectors.parked.item.type = car
detectors.parked.window.size = 50
detectors.parked.window.slide = 10
detectors.parked.displacement.min = 0.0
detectors.parked.displacement.max = 0.5
detectors.parked.repetition.triggers = 30

# Driving Detector Properties
detectors.driving.item.type = car
detectors.driving.window.size = 50
detectors.driving.window.slide = 20
detectors.driving.displacement.min = 1.0
detectors.driving.displacement.max = 10.0
detectors.driving.repetition.triggers = 10
```

Figure 17: The final configuration file used within the activity analysis service. This displays the parameters that are available to the developer in order to tune the different activity models.

4.3.3 Enabling Event Analysis is completed through the provided Apache Spark task. The task requires no development changes for this use case, as the provided unsupervised event analysis model is sufficient for this deployment. When configuring this service, the number of clusters the unsupervised K-Means algorithm will use to classify events had to be selected, which is done through a configuration file. To investigate the number of clusters that should be identified the system was ran within a local environment, generating classifications based on typical data produced from the activity analysis service. This data was then investigated in order to determine the number of activity clusters that existed, resulting in the selection of 5 for this specific use case.

Consuming the anomalous scores, a separate Apache Flink task was deployed, configured to send emails when an anomalous result was witnessed (Figure 18). The trigger for this service to send an email can be caused by either marking a cluster as anomalous, sending an email for every record assigned to that cluster, or when a record deviates by a given threshold from its assigned cluster. These parameters are tuned through a configuration file, providing the capability to select rules influenced by the data produced from the event classification service without requiring a development change. As this is a small use case with a limited runtime it is difficult to create meaningful rules, therefore all items within cluster 1 were set to be anomalous, allowing the evaluation of data interrogation when an anomaly is witnessed. The email sent when an anomaly is observed contains information to identify and investigate the anomalous record, with a link to the Neo4J cluster where data can be visualized and interrogated.

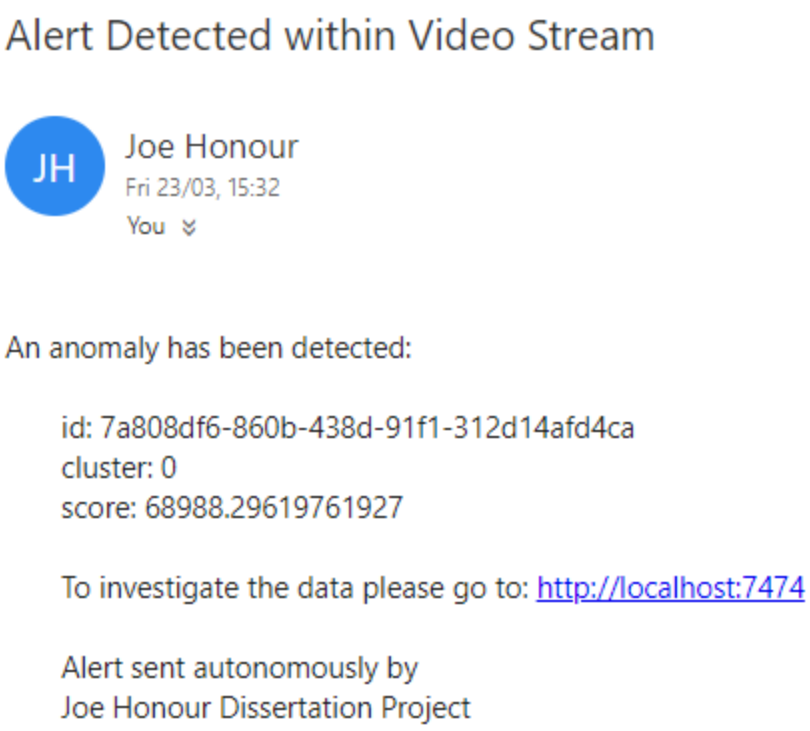

Figure 18: The email sent to a user when an anomaly is detected. This included the unique ID of the object involved in the anomaly, the event classification cluster it was assigned along with its deviation score.

4.3.4 Data Persistence was provided through the deployment of the data storage task, which takes all Apache Kafka streams and stores them as nodes within Neo4J. The task was then configured to create relationships between the nodes (Figure 13) to allow for reasoning with regards to event causality. This also provided the data interfacing tool through the Neo4J dashboard and gave visualization and querying capabilities over the produced data.

4.4 Deployment Procedure

To deploy the infrastructure required to run the analytics pipeline, the pipeline makes available a series of Terraform scripts. This enables the smooth, reproducible, deployment and management of the production infrastructure.

4.4.1 Apache Kafka Deployment is required before any other stage, as it is the single communication broker between all services. In order to deploy Apache Kafka, the built in Terraform scripts were executed, outputting the dynamic connection details required for all services to talk to Apache Kafka.

4.4.2 Neo4J Deployment enables the storage of all data flowing through the Apache Kafka cluster. The Neo4J database was deployed onto a single machine and its connection credentials were made available to the data storage service, which will be deployed once the Apache Flink infrastructure is running.

4.4.3 Video Pre-Processing Deployment is responsible for parsing real-time video, with the adoption of Edge Computing techniques, this stage needs to be deployed as close as possible to the camera source. This service was therefore deployed to AWS region Ireland, as this is the closest region with the hardware requirements necessary to run the service. The service was then started, processing the raw video stream, and sending the annotated object location and tracking data to the deployed Apache Kafka cluster.

4.4.4 Apache Flink Deployment is responsible for producing a cluster of machines capable of running all required Apache Flink tasks. The required Apache Flink tasks include activity analysis, event alerting and data persistence. The Apache Flink cluster was deployed through the execution of the default scripts. Once the cluster was deployed, each task was started

in order, being passed the Apache Kafka credentials of the deployed Apache Kafka cluster, allowing them to communicate through the distributed messaging layer.

4.4.5 Apache Spark Deployment is used to run the event classification service. The Apache Spark cluster was deployed using the default Terraform scripts, with the event classification job being submitted to the cluster for execution on successful deployment.

With the modular deployment process completed, the video analytics pipeline was running successfully over the raw video footage.

4.5 Findings

Metrics were collected from a thirteen-megabyte (four minute) section of the video analyzed by the deployed pipeline. The objective of these metrics is to provide insight into the data flow and storage size required to analyze video footage from a single camera feed. This aims to accompany the overall evaluation of the pipeline, by providing the pipelines footprint on the architecture it is deployed upon.

Our first metric is the amount of storage used throughout each stage of the pipeline in order to provide analysis over the segment of video footage (Figure 19, Figure 20).

Video footage used can be found at: https://www.youtube.com/watch?v=V3oZI1G3H5M&t=1s Accessed: 30/04/2018

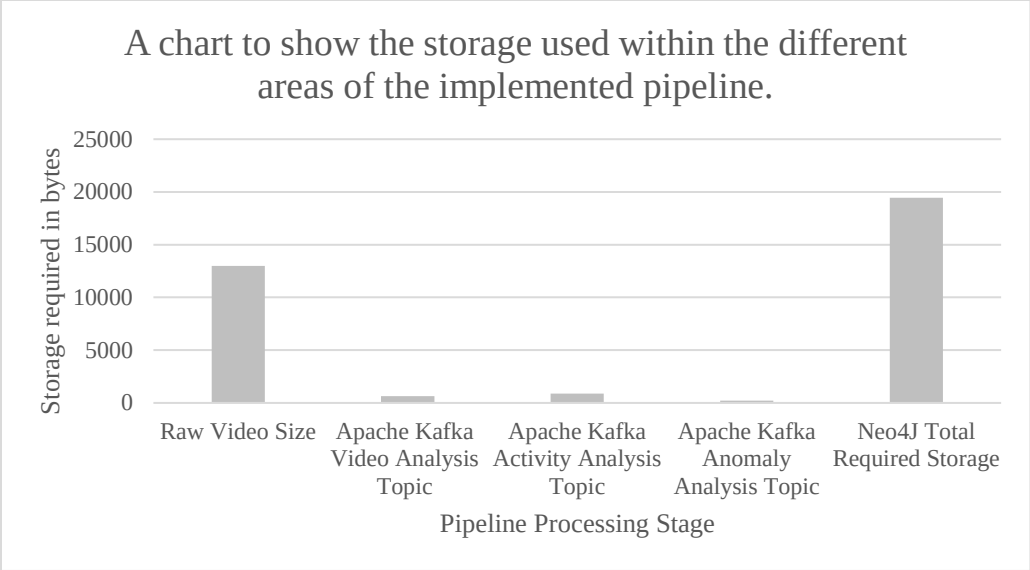

Figure 19: A chart to show the storage used within the different areas of the implemented pipeline over the four minute section of video processed. It displays clearly how the real-time processing stages (video, activity, and anomaly analysis) drastically minimize the amount of data stored, against the raw video footage and the database store.

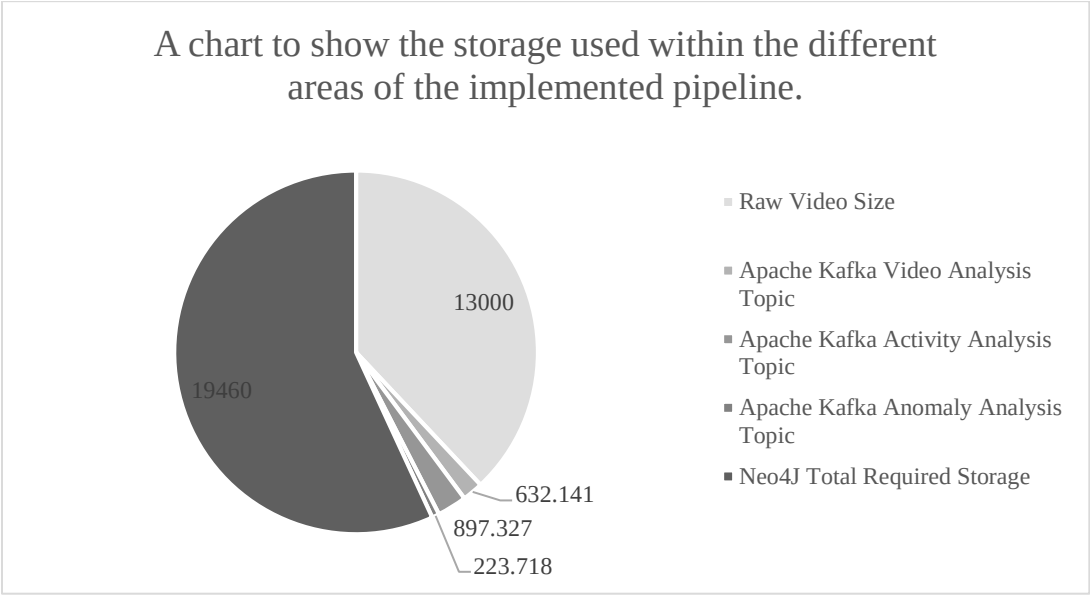

Figure 20: A chart to show the storage used within the different areas of the implemented pipeline over the four minute section of video processed. This aims to show the proportion of data stored at each section of the pipeline in comparison to other stages.

The second metric collected was the number of messages sent by the system at each stage of the pipeline, to show the strain each system placed on the pipeline as individual entities (Figure 21, Figure 22).

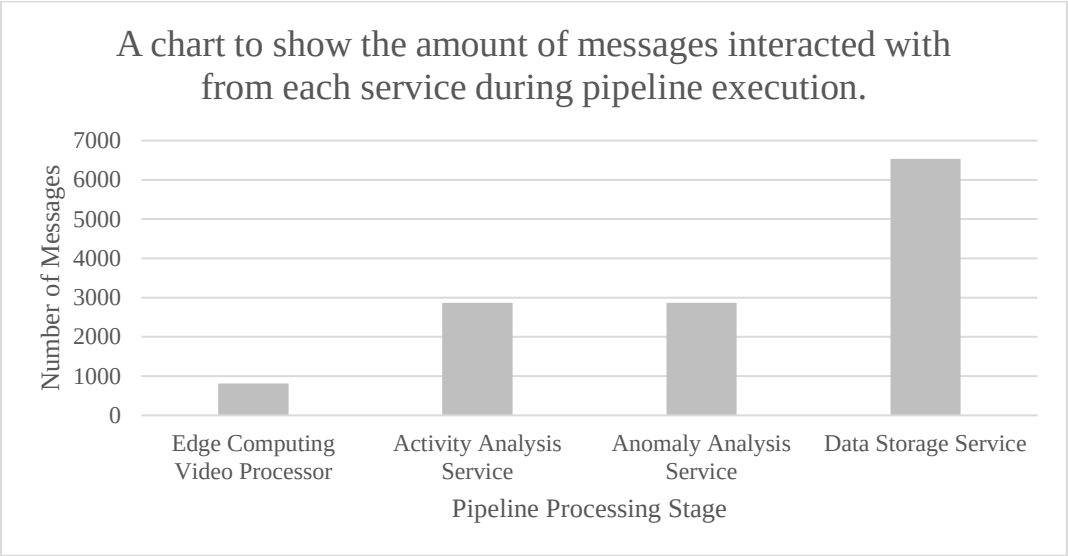

Figure 21: A chart to show the amount of messages interacted with from each service during pipeline execution. This aims to show the number of messages required to provide the functionality found in each service within the pipeline.

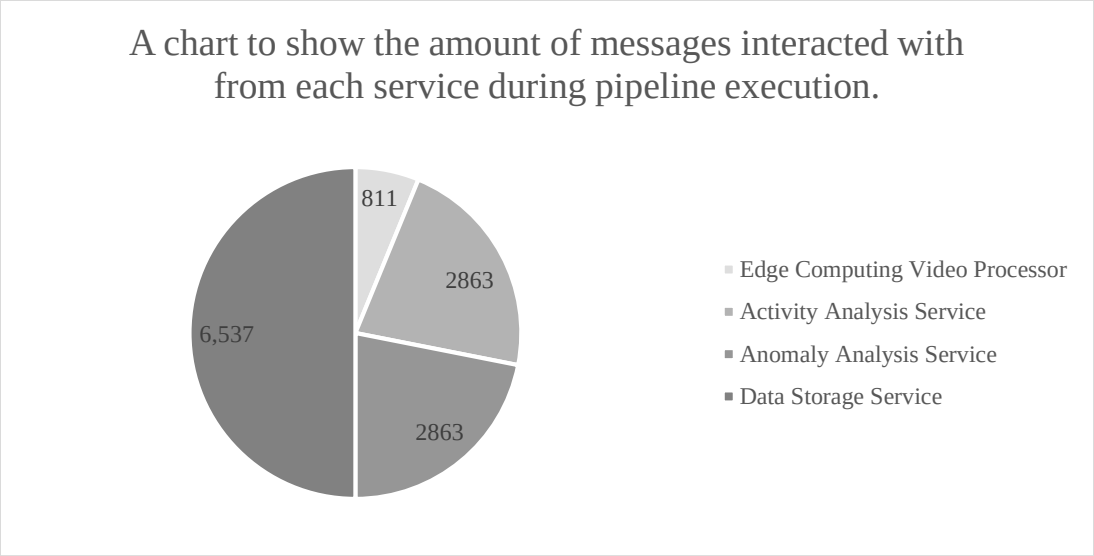

Figure 22: A chart to show the amount of records interacted with from each service during pipeline execution. This aims to show the proportion of records required by each service, in comparison to other services within the pipeline.

4.6 Evaluation

With the aim of showcasing the pipeline as a realistic base for video surveillance operations, the evaluation of the pipelines ability to present useful information is critical. Furthermore, the evaluation of the deployment procedure and performance of the pipeline are required. This aids in determining if the pipeline can physically be used with current surveillance systems and that the pipeline is capable of scaling to meet large user requirements.

4.6.1 Development Experience is a major part of the pipelines capability to be adopted when providing smart surveillance systems. As shown through the use case development procedure, minimal modifications are required to the existing pipeline to provide core functionality. However, when tuning models, a notion of typical data fed into the processing stages is required, which may become problematic if this cannot be obtained without a

production installation of the system. In contrast to this, the deployment procedure for the required infrastructure is simple and abstracted away from the developer. This enables a fast turnaround time when testing and deploying services. Coupled with the fast deployment of infrastructure, services are also designed to take in parameters at runtime through configuration files and command line arguments. This allows tasks to be deployed once, then ran with a variety of different configuration arguments, allowing a quick exploration of parameters in order to determine an optimum configuration for the specific use case.

4.6.2 Data Interrogation within the deployed pipeline is made available through a web browser interface, supplied by Neo4J. A common use case from the deployment of the pipeline, is to be able to visualize the causality of a particular event. This is the process in which the user is able to explore the downstream events contributing to a produced upstream event, such as the classification of a point to a particular cluster. This information can then be visualized and interrogated for further understanding.

To support this, the data is stored in an accessible state through the use of Neo4J, allowing for overall visualizations of the running pipeline to be produced, as shown in Figure 23. Further to the meta-data collection of witnessed events, the activities that have been identified can also be visualized within 3-Dimensional space, allowing for interactive exploration of data formations over time (Figure 24).

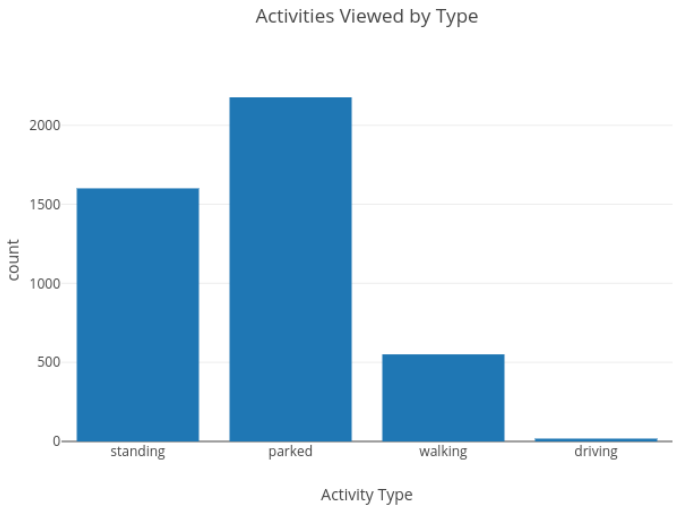

Figure 23: The implemented video processing analytics pipeline detecting activities within the given video stream.

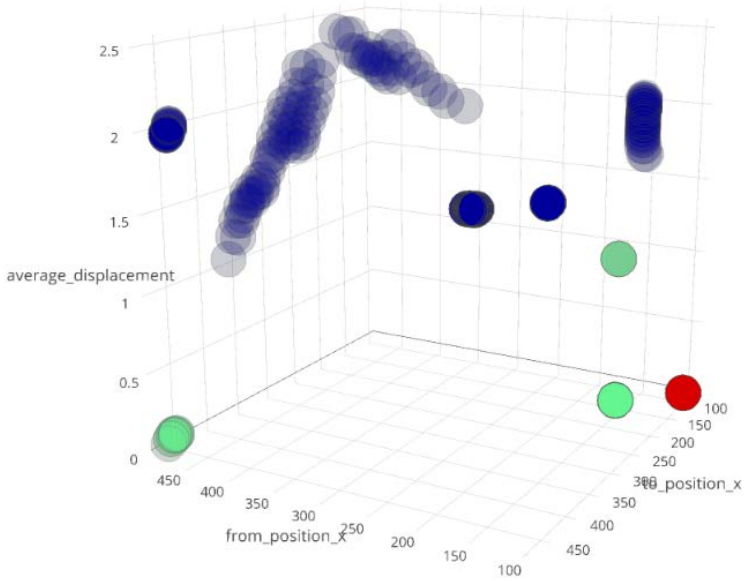

Figure 24: A plot to show the average displacement of events by the event type, in their observed x position. This shows the clustering of event types in the video stream, and where they are usually found. Enabling the isolation of an event for investigation from within the displayed graph.

From this, we can then evaluate the pipelines availability to track down a specific data point, to simulate a user receiving an event notification that they need to investigate. This aids in displaying the pipelines ability to accurately present data to a user, allowing them to make meaningful decisions based on the analysis produced. As you can see from Figure 25 we can visualize an event within its data space, in order to gain context as to where the event is located with respect to other data processed by the pipeline. From this point, we can take the unique ID displayed and login to the running Neo4J instance for further investigation.

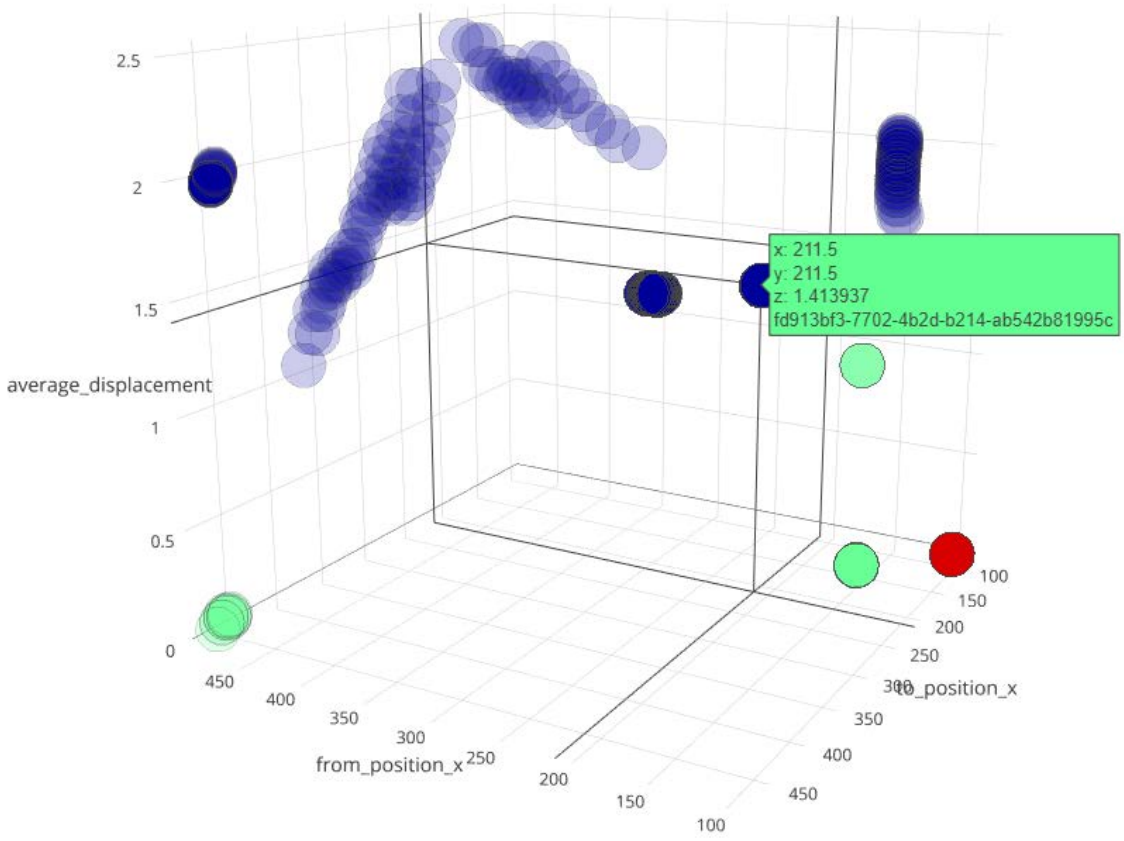

Figure 25: Isolating an event for investigation from within the displayed graph.

Within Neo4J queries can be executed over the dataset to find a particular node that meets a criterion, which are written in Cypher Query Language [45]. On execution of the query,

Neo4J returns an interactive graph (Figure 26) showing the corresponding data node. We can then make use of the graph database functionality and expand the nodes relationships to nodes around it. This enables the user to clearly see the anomaly score and cluster associated with this node, while also being able to explore other nodes connected to any of the nodes in question. Enabling the user to explore causality of events through the use or relational modelling within the graph database, we have met the aim of not only proving the sub systems are capable of interacting, but that they produce tangible information that is in a format the user can visualize and explore to gain insight.

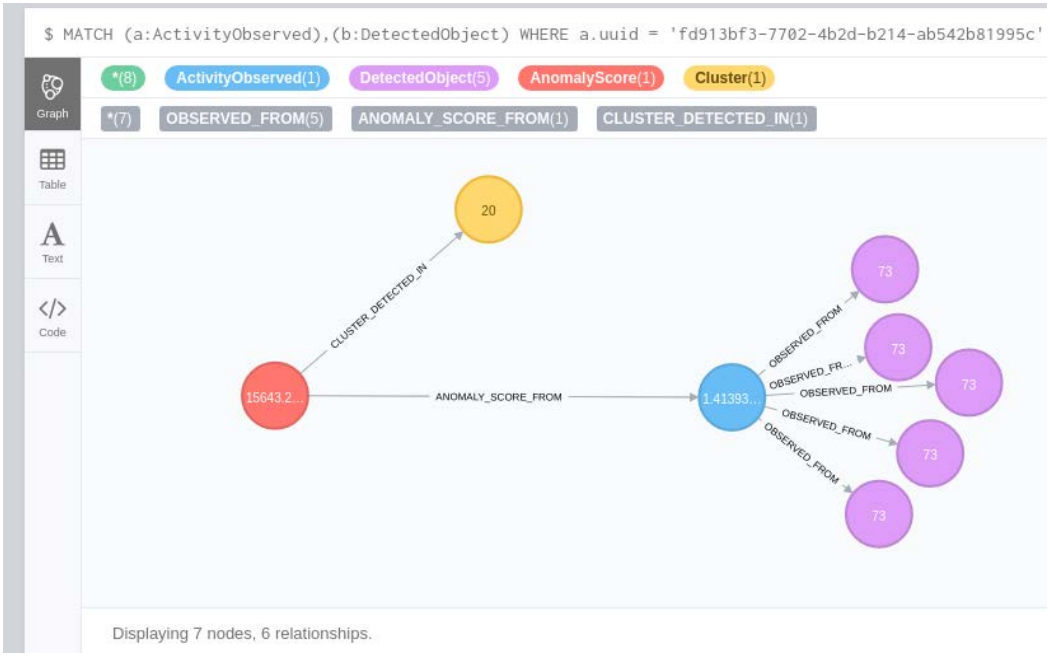

Figure 26: Finding the event for investigation in the data store (blue) now allows us to observe the anomaly score given to the event (red), along with the frames in which it was observed (pink). Along with this, we can see the cluster that the anomaly belongs to (yellow), and if wanted, can expand the cluster to see other associated events.

4.6.3 Ease of Deployment was achieved through the use of Terraform scripting. This required only a small manual input to configure some initial SSH connectivity settings within the AWS Management Console, before being able to deploy the complete infrastructure in stages. Deployment was smooth due to its separation from the individual running of tasks. Deploying separate infrastructure machine clusters meant that any problems were compartmentalized and easy to track down, without having to attempt redeployment of the entire system. This also required no knowledge of the underlying machine allocations to execute, as the scripts output connection information on successful completion and therefore a user is able to perform the submission of tasks and management of infrastructure from each distributed systems user interface. Concluding, this enables a user to quickly deploy infrastructure at scale, while still allowing for extension and management of machines through the provided scripts if a more advanced setup is required.

4.6.4 Pipeline Performance was measured by collecting the records produced within Apache Kafka along with the data storage amounts used by each service. This allows an accurate measure of what effect each analytics system has on the overall requirements of the deployed analytics pipeline.

With an initial video storage size of thirteen megabytes, the effects of edge computing can be observed on the overall data flow of the system (Figure 19). The video processing service successfully analyses the video footage, converting it into object locations and tracking information, resulting in an output storage requirement of six hundred and thirty bytes. This is a drastic reduction in storage capacity against the storage of the original video footage. Accompanying this, it is observed that the downstream analysis stages are still able to produce reliable insights into the video stream based only on the data produced by the video processing

service, showing that the meaningful data held within the video stream is successfully extracted to an appropriate level to provide in-depth analytics. Thus, we can conclude that Edge computing provides a drastic improvement over network and storage requirements needed of a deployed analytics pipeline.

Building on this, we can further observe that the downstream processing stages have very little overhead on the system in comparison to the edge of the processing pipeline (Figure 20). This displays how the filtering and aggregation of video annotations, over the streaming of the raw video footage, drastically reduces the amount of further processing that has to occur in downstream analytics stages of the pipeline. However, we can see that the physical data storage of Neo4J is larger than the original video footage file. This is mainly due to the commit log of Neo4J being stored in order to replay or restore the system to a previous stage, which is a noticeable overhead (Appendix, Item 2).

When looking at the stress of the system as a single unit, the key area to observe is the messaging system, as this is the integration point of all systems. Thus, a lag in this area will cause further downstream services to be limited in the speed in which they can process records. It is observed that the activity and anomaly services produce the most strain on the messaging layer (Figure 21, Figure 22), as they extrapolate multiple analysis records based on the object tracking information produced from the video analysis service. We can further observe that the analysis produced from the activity and anomaly service is of a magnitude of double the video analysis data source. Furthermore, we observe that there exists a one to one relationship between the messages produced by the event detection service and the event classification service. From this we can infer that, when deploying the Apache Kafka cluster, the provisioning of more resources to the messaging topics associated with the event detection and classification services

would be necessary. This would require partitioning the effected topics across more of the brokers, spreading the load placed on any single broker, in order to handle a larger amount of traffic.

4.6.5 Inferring Scalability to meet Smart City requirements is an important factor when considering the success of the pipeline. Through the close analysis of the message and storage requirements of the analytics pipeline in this use case, we can infer what deployment specifications would be required to meet smart city requirements. However, we will not infer the exact processing capabilities the system will need to contain, as this will be too highly dependent on the analysis techniques being performed on the video streams, and the models being applied to the analysis results.

Using the data collected from the thirteen-megabyte, four-minute, video sample used in the pipeline performance measurements, we can extrapolate our measurements to provide an average data production amount of a single camera in twenty-four hours (Table 7). We can conclude that with a message production rate of 27.24 messages per second, Apache Kafka is well within its theoretical limits to support the integration of services within the analytics pipeline over a single camera feed.

Kafka Data Stored (KB)	Database Storage (KB)	Kafka Messages Produced	Message Production Per Minute	Message Production Per Second
631,146.96	7,005,600	2,353,320	1,634.25	27.24

Table 7: A table to show the data produced from the analytics pipeline, processing a single CCTV camera, over a 24-hour period.

However, we can now expand on this data to view the suitability of the proposed architecture under Smart City requirements. As Surat, India moves towards becoming a Smart

City, it currently has deployed 6,000 CCTV cameras streaming to its command and control center. If we assume each camera produces, on average, the same amount of data as our use case we can extrapolate the theoretical data production the proposed pipeline would generate in order to process and provide analytics across the 6,000 camera streams (Table 8). We are able to conclude, at a rate of 163,425 messages per second, Apache Kafka would still be able to theoretically support the system in a production environment [44].

Kafka Data Stored (KB)	Database Storage (KB)	Kafka Messages Produced	Message Production Per Minute	Message Production Per Second
3,786,881,760	42,033,600,000	14,119,920,000	9,805,500	163,425

Table 8: A table to show the theoretical production of data by the analytics pipeline, processing 6000 cameras, over a 24-hour period.

5. Stakeholder Evaluation

As the proposed analytics pipeline is expected to enable projects in the realm of video surveillance, a stakeholder with experience within this domain was presented the pipeline and asked for feedback, with the goal of evaluating its scope and impact within the United Kingdom Police Force (Appendix, 8.2). This evaluation can then identify the benefits the pipeline could bring to the development of new surveillance systems.

5.1 Stakeholder Background and Experience

When weighting the relevance of the feedback given, the stakeholder's background and experience needs to be assessed. In order to do this, the stakeholder provided a statement with an overview of their relevant experience.

> "I currently work for Cheshire Police, having recently retired after nearly three decades with West Mercia Police. During this time, I had roles including but not limited to; Police Constable, Operational Intelligence Officer, Target Development Officer, Pro-active Crime Officer and Detective Constable. Prior to joining the Police, I worked within the Armed Forces, spending a number of years within the Anti-terrorism unit, making use of surveillance equipment including the use of CCTV."

This allows the stakeholder to be in a unique position of having extensive knowledge of Police and military operations, while obtaining firsthand experience of the use of CCTV systems currently in operation. Furthermore, the diversity of roles and positions held by the stakeholder within the Police Force allow a wide variety of application areas to be considered for the

proposed analytics pipeline. Thus, giving a full assessment of the pipelines impact and usefulness within current Police operations.

This information was shared by the shareholder on the strict agreement it would not be distributed outside of Newcastle University without expressed written consent.

5.2 Evaluating Applicable System Use Cases

As the pipelines goal is to support the development of smart surveillance projects, providing a grounding for domain specific use cases, the stakeholder offered a variety of applicable deployments of the system that would aid the UK Police Force. From these use cases, we can evaluate the pipelines ability to support the business case and estimate what level of development modification would be needed to meet the specific requirements. This aims to provide a metric for the pipelines appeal to real world surveillance operations, and the impact the pipeline could have on the UK Police force, while showcasing the benefit of adopting a common pipeline to video processing over the development of bespoke applications.

5.2.1 Policing Clustered Crimes was proposed by the stakeholder as an initial use case for the analytics pipeline. The stakeholder described the problem Police currently face as a series of crimes within a given area that are related, such as a high rate of vehicle thefts within a neighborhood. To combat this, the Police often deploy static cameras to the area, enabling them to capture criminal activities in order to prosecute the perpetrators. Further to this, if vehicle break-ins are reported, the Police frequently employ what are known as rat-traps; a series of dummy vehicles equipped with CCTV cameras that can record any attempted break-ins. These CCTV feeds are frequently monitored however it is often impractical to monitor all cameras at all times. Therefore, the CCTV feeds are typically used for evidence to track down a perpetrator rather than to catch the perpetrator in the act.

The stakeholder evaluated the use of the analytics pipeline, stating how a system able to detect specific criminal behavior patterns, such as the theft of a car, coupled with real-time alerts would allow the Police to act with a much faster response time. This aids in using CCTV to drive a more pro-active approach to policing, rather than its primary use as a traceability tool.

In evaluating the stakeholder's requirement for the detection of clustered crimes, we can deem that the pipeline would be able to provide this service with extension. The extent of the work required for the pipeline to meet this objective would be primarily compartmentalized within the activity analysis service, with modelling of behavior patterns leading to criminal activities being required. This relies on the assumption that the video processing service captures the necessary information about vehicles and cars. Further work could be extrapolated if it's deemed a more in-depth analysis is needed of the person's location, for instance the positioning of a person's limbs.

This application of the analytics pipeline would provide benefit to the Police Force through the reduced man hours spent watching live CCTV, alongside an improved reaction time to the detection of criminal activities provided by real-time alerts. Furthermore, if real-time alerts were sent directly to active Police Officers, the system could aid in the automation of the Police response to witnessed criminal activities.

5.2.2 Police Officer Location is important when deterring crimes, as correct placement of officers allows for a quicker response time to criminal activities. The stakeholder presented the case that criminals often repeat offenses in remote areas, where Police response times are likely to be predictable and slow. This allowed criminals to successfully elude capture and repeat the same crimes multiple times, aligning break-ins to when businesses have recovered from the original robbery. The stakeholder then demonstrated how the use of the analytics pipeline to

record the pattern of behavior leading to the original robbery could be used to identify the pattern reoccurring in the future, which could influence Police positioning, giving faster response times and increasing the likelihood of apprehending the criminals.

In order to provide this behavior, the pipeline would have to be modified solely within the event classification and notification services. The modification would consist of comparing data relationships found within the Graph database to what is currently transpiring, sending an alert when previous patterns align with current observed patterns, within a configured degree of accuracy. This would allow the benefit of notifying Police in real-time to suspicious behavior patterns, known to be typical of criminal activity. However, this approach does rely on criminals having already committed a similar crime, and therefore does not prevent all criminal activities. Although the stakeholder states that “criminals are people of habit”, the model would have to filter normal activities from those which lead to criminal behavior, which would further rely on a human input, removing some of the benefit of an automated system.

5.2.3 Traffic Monitoring allows the capture of vehicles on motorways and other major roads within the United Kingdom, with the Police attempting to stop vehicles that are driving erratically. The stakeholder presented the use of the analytics pipeline to track and identify vehicles that are not conforming to normal driving patterns, for instance if a driver is falling asleep or speeding excessively. This could then inform Traffic Officers, who could begin pursuit of the dangerous vehicle.

The pipeline would be able to perform this currently, with runtime modifications to the provided car activity detection models. This deployment would have a large impact on the Traffic Officers, guiding their placement and reaction time to dangerous drivers, actively saving lives. This is a domain in which the pipeline could be deployed with minimal modification, due

to it already containing models for the detection of moving cars along with the unsupervised modelling of abnormal behavior, showing cars deviating from the typical behavior of vehicles.

5.2.4 Person of Interest Identification would allow Police forces to track and capture criminals without active monitoring of CCTV systems. The stakeholder stated how the systems deployment to large metropolitan areas, in its current state, would allow for the detection of strange activities which, on detection, could send facial images of people involved in the activity for identification. This would avoid the constant running of facial detection software, while enabling the identification, and subsequent tracking, of individuals who exhibit behavior deemed criminal. The stakeholder then stated that only on the facial identification of a known criminal would an alert be sent to a control room, allowing for suspicious behavior not involving known criminals to be discarded.

The current pipeline would require extensive modification to be able to meet the requirements of this scenario, with the addition of sending facial images to server-side processes that could then perform facial recognition against an existing database of criminal faces. Regardless of the extensive modification, the pipeline would still be able to support this kind of large modification due to its modular design and distributed messaging layer; the decoupled architecture implemented enables new services to communicate with current services, without the modification of existing services. Thus, we can still conclude the pipelines flexibility provides a large benefit to Police operations that may require very intricate use cases.

5.2.5 Time Critical Crimes are often the most serious, such as kidnappings, where the speed in which the Police can process all information and act directly effects the likelihood of a victim’s survival. Within this domain, the stakeholder presented the value the analytics pipeline could have in processing video feeds pertinent to a time critical investigation. With the pipeline

used to process all available video footage, it could flag anomalous behaviors witnessed, allowing the Police to immediately playback these incidents, vastly reducing the amount of time spent watching CCTV feeds before a viable lead is detected. This allows the Police to act significantly faster and, if no leads are detected, does not waste a large amount of man power which could otherwise be spent in other avenues of investigation.

The pipeline would be able to meet the use case described with a small modification to the video processing service, allowing it to process video at a rate faster than the video was recorded. The pipeline deployment scripts can be utilized to deploy extensive infrastructure for each service, allowing for the largest throughput to be achieved in order to reduce the amount of time spent processing the video footage. This use case specifically shows the power of the pipelines modularity, allowing for its pre-existing services to be utilized for use cases outside of the original design. This deployment of the pipeline could allow for Police to act faster during kidnappings, improving the likelihood of victims being returned to their families.

5.3 Evaluating System Impact

Given the presented use cases from the stakeholder, it is apparent that the pipeline could either directly be deployed to or, with modification, enable an extensive set of features that could aid the UK Police Force.

When evaluating the primary benefits of the pipeline it is apparent that the modular design brings the flexibility required to meet bespoke use cases, while sharing common functionality between applications reducing the development time and cost in deploying the pipeline to production. The stakeholder, when referring to the ability to deploy individual services within the pipeline, stated that the pipeline “would give great flexibility in where we could use the system”. Furthermore, the stakeholder presented that a large problem the Police

Force faces through its use of CCTV is data collection and storage, with video often having to be downloaded in real-time. As the pipeline aggregates all data it alleviates the problem of data collection, extracting the necessary information from the video feed and making it immediately accessible to the Police. However, it does not have a solution to the storage of the raw video footage itself, as this will still be required for evidence within a court case, and this is a constraining point on the current system.

Therefore, concluding on the stakeholder evaluation, the proposed analytics pipeline design and implementation would provide significant benefit to the Police Force. Looking to the future, the stakeholder suggested that the system would currently be suitable for large metropolitan areas. However, further work would be needed, with regards to producing more specific models for the identification of particular criminal activities, if the pipeline was to be deployed to more rural areas. Further to this, the stakeholder stated that he was unaware of any existing solutions offering similar services to the Police Force, and that the software's free license and open hosting make it an extremely desirable avenue for the Police to pursue.

This evaluation, coupled with the use case demonstrating the approachability and physical development of the analytics pipeline, aims to show how the pipeline could support a large variety of surveillance operations. Bringing the benefit of modular design and reusability, the pipeline is adaptable to all stated use cases with, on average, a minimal amount of modification, reducing the cost and time to deployment of smart surveillance systems.

6. Discussion and Conclusion

6.1 Summary

Within this dissertation and the supporting work, a proposed real-time video processing pipeline has been architected and implemented. This implementation was proved successful through the use of a directed use case. The use case actively displayed the pipelines achievement in adopting Edge computing to reduce data transfer, along with its capability to perform end-to-end analytics over a video stream. Accompanying this, the pipelines offering of generic event classification enables it to be applicable for nearly all use cases with minimal modification, displayed through the use case implementation. Furthermore, when the pipeline was presented to a viable stakeholder within the UK Police Force, an extremely positive response was given showing merit in the pipeline to improve current surveillance operations.

6.2 Discussion

With the implemented analytics pipeline we aim to achieve scalability and ease of deployment in order to enable smart video surveillance. Through the use case presented, we have shown the adoption process in using the analytics pipeline for a bespoke application. Although the underlying technologies are distributed and have been shown to scale to millions of messages per second, the use case presented is limited and not designed to be a true load test of the pipeline. This leaves the burden of proof, with regards to scalability, on the technologies adopted and not to their specific use within this pipeline. Therefore, use cases should be devised to further prove the scalability of services in this specific domain.

The pipeline has been seen to improve on existing work through its technology agnostic architecture, achieved through its distributed messaging layer provided by Apache Kafka. Also, the use of Edge Computing to extract the meta-data required by downstream processes, rather

than streaming the raw video to the messaging layer, allows for a high throughput and optimization of the analytics pipeline. This is something unseen in previous work and allows our pipeline to have a reduction in the size of data transferred within it, enabling the optimization of deployed infrastructure. Building upon this, this dissertation shows that the pipeline implemented has the capability to meet the requirements of a smart city's surveillance operations; something unseen in previous work within this domain.

Developments in deep learning have allowed for sophisticated object detection and tracking. Accompanying this, deep learning can also provide the combination of many steps within the pipeline, for instance the detection of different object types and observed motions [46]. This may cause the performance limitations of the pipeline to be in the initial capture and processing of the raw video stream, and not in its downstream analytics. Therefore, the overhead of distributing work over many machines may not be necessary if the work being performed individually is expensive. Further to this, deep learning is often only made available through machines with GPU based hardware. Current Cloud providers offer this hardware as a service, however it is extremely expensive compared to CPU based compute power. The currently implemented pipeline deploys modestly powered machines, without GPU functionality, and therefore the pipeline may become too expensive to run on a Cloud environment if a deep learning-based model is adopted. However, with the use of Edge Computing, GPU hardware could be deployed physically within the surveillance camera network, and the results of processing the video could then be fed downstream to a Cloud based analytics pipeline. This may hinder the adoption process of the analytics pipeline though, as specific hardware requirements now have to be deployed to surveillance networks.

During the evaluation of the pipelines impact within the UK Police Force, many operational areas were identified that would be improved through real-time feedback of identified events. The pipeline was deemed to require minimal modification, in the most part, to functionally meet these requirements. However, a change to the current Policing approach would be needed; Police Officers will physically have to work differently if they are to use the information produced by the pipeline. This may require a large amount of time, with frontline training required, limiting the speed in which the pipeline can become operational and start producing information that is acted upon.

6.3 Objectives Met

In order to deem the success of the work completed a concluding point is made against each of the initial objectives, enabling a true evaluation of the final projects success.

6.3.1 Objective One: To investigate existing academic literature alongside current video processing techniques and available software packages, in order to support the creation of a scalable analytics pipeline.

During the project we have successfully identified existing technologies, evaluating their capability to provide modular design and extensibility, while observing their limitations with regards to scalability and development opportunity. Further to this, we can conclude on their influence over the architected analytics pipeline defined within this dissertation. The modularity previously seen is carried forward to the final analytics pipeline described, with the pipeline providing a clean separation of technologies and functionality within its processing stages. Additionally, the pipeline improves over previously seen work through its adoption of distributed messaging and processing, enabling users to meet high throughput and low latency requirements, which may not have been met using existing approaches. This was accomplished through the

study of real-time analytics pipelines that were successfully deployed to other domains, such as log and message aggregation.

6.3.2 Objective Two: Present and develop an analytics pipeline that provides a minimum viable product of object, activity and event detection, while being scalable and extensible.

Delivered as part of this work is an analytics pipeline that ships with object detection and tracking techniques that are able to convert raw video into video annotations that can be consumed by downstream processes. Downstream processes are independent of each other and consist of activity detection including models for; people standing, people walking, people running, cars parked and cars driving. Event detection is unsupervised and provided in the form a K-Means clustering model. All services are extendable, as shown through the evaluated use case, displaying how modifications and tuning can be made to modules within the pipeline to meet bespoke use cases. The analytics pipeline is built upon distributed technologies; Apache Kafka, Apache Flink, Apache Spark, allowing for large scale applications to be supported. This combination of features allows me to conclude that this objective was fully met.

6.3.3 Objective Three: Develop a use case that will allow the evaluation of the real-time video processing pipeline.

The use case adopted enabled the analytics pipeline to be tested in a bespoke setting, typical of a real-world surveillance deployment. It required modification and adjustment to deployed services, showing the flexibility of the pipeline. Further to this, as it was using video produced from currently deployed surveillance equipment, it forms a basis for inferring the pipelines capability to integrate with currently deployed surveillance systems.

6.3.4 Objective Four: Using the use case defined, evaluate the pipelines ability to support existing video processing techniques, while meeting bespoke user requirements.

The use case instrumented within this dissertation showed the development experience when adopting and interacting with the proposed pipeline, indicating its approachability for development and ease of use in bespoke circumstances. Furthermore, it proved that the pipeline could be deployed with ease, even with bespoke application requirements. The demonstration of the underlying technologies interoperability and their practical application allows me to conclude this objective has been met, however further work needs to be undertaken to prove the pipelines operation at scale, which is expanded upon in 6.4 future development.

6.3.5 Objective Five: Evaluate the pipelines applicability to current projects being conducted by the United Kingdom Police Force, using a directed interview to a viable stakeholder.

Once the pipeline was developed, it was described and presented to a viable stakeholder with significant experience both in the UK Police Force and the Armed Forces. The stakeholder showed a variety of use cases that the pipeline could be applied to, offering significant benefit to current Police operations. This showed how the pipeline could be currently used, and extended for use, displaying the benefits of the pipelines modular design and flexibility to meet the broadest selection of required use cases within the domain of video surveillance. The pipeline was shown to bring benefits including; the accurate placement of Traffic Officers, the capability to detect criminal activity patterns enabling Police Officers to respond in a faster time, and the batch processing of video so as to highlight points of interest to be investigated within time critical cases.

6.4 Future Development

The presented analytics pipeline offers some core functionality with its delivery enabling the pipelines approach to video processing to be assessed. However, the exploration of the

pipelines ability to support advanced video processing techniques has yet to be achieved. A true test of the pipeline will be its adoption alongside a specific research area. A major area of future work is therefore interfacing the pipeline with complex processing models, such as deep learning based schemas. This would enable proof that the pipeline is able to support features seen within current research areas, including those that contain custom hardware requirements typically not associated with distributed computing.

Further to this, future work can be conducted to show the cost effectiveness of the pipeline in large scale deployment scenarios. As Cloud providers allow users to rent hardware at a fixed price, it would be interesting to investigate the price required to support large scale use cases, and contrast this to the purchase of the equivalent hardware. This would allow for users to see cost savings associated with on premise deployments of infrastructure, in contrast to Cloud deployments.

As the pipeline produces a large amount of statistical data, it would also be of use to see metrics and statistics on the dataset in real-time. The ability to see event location densities and volumes may provide useful insight into camera positioning and movement patterns. From these analytics, a custom dashboard could be created allowing real-time visualizations of the data being produced within the analytics pipeline. This would allow for a more unique user experience and allow a greater interface into the data produced.

A further piece of work could also be developed to allow for the storage of video when an event worthy of a notification is produced. If an event causes an alarm, this notification could not only be sent to the client, but also to the video capturing service. The capturing service would then, instead of discarding the video, send that short piece of footage to a storage service. This would allow an end user to not only visualize the series of events that caused an alert to be sent,

but also keep the video evidence of the event occurring for future reference or refinement of models within the analytics pipeline.

Taking on possible applications from the stakeholder within the UK Police Force, future work towards the integration of new modules including facial recognition and Automatic Number Place Recognition (ANPR) would be extremely beneficial to the pipelines applicability to current surveillance operations. Furthermore, creating event classification models focused on the learning and detection of specific criminal activities would allow the pipelines deployment to Police operations outside of metropolitan areas.

7. References

[1] T. Reeve, “How many cameras in the UK? Only 1.85 million, claims ACPO lead on CCTV,” *CCTV Image magazine*, 2011. [Online]. Available: http://www.securitynewsdesk.com/2011/03/01/how-many-cctv-cameras-in-the-uk/.

[2] B. C. Welsh and D. P. Farrington, “Public area CCTV and crime prevention: An updated systematic review and meta-analysis,” *Justice Quarterly*, vol. 26, no. 4. pp. 716–745, 2009.

[3] J. M. Caplan, L. W. Kennedy, and G. Petrossian, “Police-monitored CCTV cameras in Newark, NJ: A quasi-experimental test of crime deterrence,” *J. Exp. Criminol.*, vol. 7, no. 3, pp. 255–274, 2011.

[4] S. J. McLean, R. E. Worden, and M. S. Kim, “Here’s Looking at You: An Evaluation of Public CCTV Cameras and Their Effects on Crime and Disorder,” *Crim. Justice Rev.*, vol. 38, no. 3, pp. 303–334, 2013.

[5] S. Yard, “CCTV in Homicide Investigations,” 2010. [Online]. Available: https://goo.gl/oS5Tgn. [Accessed: 15-Nov-2017].

[6] E. L. Piza, J. M. Caplan, and L. W. Kennedy, “Analyzing the Influence of Micro-Level Factors on CCTV Camera Effect,” *J. Quant. Criminol.*, vol. 30, no. 2, pp. 237–264, 2014.

[7] S. Germain, “A prosperous ‘business’: The success of CCTV through the eyes of international literature,” *Surveill. Soc.*, vol. 11, no. 1–2, pp. 134–147, 2013.

[8] SkyBell, “SkyBell HD,” 2018. [Online]. Available: http://www.skybell.com/product/skybell-video-doorbell-hd/. [Accessed: 04-May-2018].

[9] Nest, “Nest,” 2017. [Online]. Available: https://nest.com/uk/cameras/nest-cam-indoor/overview/. [Accessed: 24-Nov-2017].

[10] Nimish Sawant, “Smart City projects need CCTV video surveillance standards to ensure future stability of data, say experts,” 2017. .

[11] T. Ahonen, A. Hadid, and M. Pietikäinen, “Face description with local binary patterns: Application to face recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 2037–2041, 2006.

[12] Y. Gao, J. Ma, and A. L. Yuille, “Semi-Supervised Sparse Representation Based Classification for Face Recognition with Insufficient Labeled Samples,” *IEEE Trans. Image Process.*, vol. 26, no. 5, pp. 2545–2560, 2017.

[13] D. Berchmans and S. S. Kumar, “Optical character recognition: An overview and an insight,” in *2014 International Conference on Control, Instrumentation, Communication and Computational Technologies, ICCICCT 2014*, 2014, pp. 1361–1365.

[14] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. W. M. van der Laak, B. van Ginneken, and C. I. Sánchez, “A survey on deep learning in medical image analysis,” *Medical Image Analysis*, vol. 42. pp. 60–88, 2017.

[15] A. S. Jain and J. M. Kundargi, “Automatic Number Plate Recognition Using Artificial Neural Network,” *Int. Res. J. Eng. Technol.*, pp. 1072–1078, 2015.

[16] M. Magrini, D. Moroni, G. Palazzese, G. Pieri, G. Leone, and O. Salvetti, “Computer Vision on Embedded Sensors for Traffic Flow Monitoring,” in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2015, vol. 2015–Octob, pp. 161–166.

[17] T. Ko, “A survey on behavior analysis in video surveillance for homeland security applications,” *Appl. Imag. Pattern Recognit. Work. 2008. AIPR ’08. 37th IEEE*, pp. 1–8, 2008.

[18] Y. Lecun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553. pp. 436–444, 2015.

[19] P. Viola and M. Jones, “Robust real-time object detection,” *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, 2001.

[20] OpenCV, “Facial Detection with Haar Cascades in OpenCV,” 2018. [Online]. Available: https://docs.opencv.org/3.3.0/d7/d8b/tutorial_py_face_detection.html. [Accessed: 30-Jan-2018].

[21] P. a Viola and M. J. Jones, “Fast and Robust Classification using Asymmetric AdaBoost and a Detector Cascade,” *Adv. Neural Inf. Process. Syst.*, no. December, pp. 1311–1318, 2001.

[22] Y. Zhu, N. M. Nayak, and a K. Roy-Chowdhury, “Context-Aware Activity Recognition and Anomaly Detection in Video,” *Sel. Top. Signal Process. IEEE J.*, vol. 7, no. 1, pp. 91–101, 2013.

[23] B. Baben and S. Belongie, “Visual tracking with online Multiple Instance Learning,” *2009 IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 983–990, 2009.

[24] Z. Kalal, K. Mikolajczyk, and J. Matas, “Forward-backward error: Automatic detection of tracking failures,” in *Proceedings - International Conference on Pattern Recognition*, 2010, pp. 2756–2759.

[25] Z. Kalal, K. Mikolajczyk, and J. Matas, “Tracking-learning-detection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1409–1422, 2012.

[26] J. Kröckel and F. Bodendorf, “Intelligent Processing of Video Streams for Visual Customer Behavior Analysis,” *ICONS 2012, Seventh Int. Conf. Syst.*, no. c, pp. 163–168, 2012.

[27] T. T. Z. T. T. Zin, P. T. P. Tin, T. Toriu, and H. Hama, “A Markov Random Walk Model for Loitering People Detection,” *Intell. Inf. Hiding Multimed. Signal Process. (IIH-MSP), 2010 Sixth Int. Conf.*, 2010.

[28] P. Antonakaki, D. Kosmopoulos, and S. J. Perantonis, “Detecting abnormal human behaviour using multiple cameras,” *Signal Processing*, vol. 89, no. 9, pp. 1723–1738, 2009.

[29] M. Goldstein and S. Uchida, “A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data,” *PLoS One*, vol. 11, no. 4, 2016.

[30] M. Längkvist, L. Karlsson, and A. Loutfi, “A review of unsupervised feature learning and deep learning for time-series modeling,” *Pattern Recognit. Lett.*, vol. 42, no. 1, pp. 11–24, 2014.

[31] A. Toshniwal, J. Donham, N. Bhagat, S. Mittal, D. Ryaboy, S. Taneja, A. Shukla, K. Ramasamy, J. M. Patel, S. Kulkarni, J. Jackson, K. Gade, and M. Fu, “Storm@twitter,” in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data - SIGMOD ’14*, 2014, pp. 147–156.

[32] P. Carbone, S. Ewen, S. Haridi, A. Katsifodimos, V. Markl, and K. Tzoumas, “Apache Flink: Unified Stream and Batch Processing in a Single Engine,” *Data Eng.*, vol. 36, pp. 28–38, 2015.

[33] M. Zaharia, M. J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, I. Stoica, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, and S. Venkataraman, “Apache Spark: a unified engine for big data processing,” *Commun. ACM*, vol. 59, no. 11, pp. 56–65, 2016.

[34] S. Chintapalli, D. Dagit, B. Evans, R. Farivar, T. Graves, M. Holderbaugh, Z. Liu, K.

Nusbaum, K. Patil, B. J. Peng, and P. Poulosky, “Benchmarking streaming computation engines: Storm, flink and spark streaming,” in *Proceedings - 2016 IEEE 30th International Parallel and Distributed Processing Symposium, IPDPS 2016*, 2016, pp. 1789–1792.

[35] Apache, “Apache Kafka,” 2018. [Online]. Available: https://kafka.apache.org/. [Accessed: 04-May-2018].

[36] J. Kreps, N. Narkhede, and J. Rao, “Kafka: a Distributed Messaging System for Log Processing,” 2011.

[37] M. Armbrust, A. Fox, R. Griffith, A. Joseph, and RH, “Above the clouds: A Berkeley view of cloud computing,” *Univ. California, Berkeley, Tech. Rep. UCB* , pp. 07–013, 2009.

[38] N. Suvonvorn, “A video analysis framework for surveillance system,” *2008 IEEE 10th Work. Multimed. Signal Process.*, pp. 867–871, 2008.

[39] J. C. SanMiguel, J. Bescós, J. M. Martínez, and Á. García, “DiVA: A Distributed Video Analysis Framework Applied to Video-Surveillance Systems,” in *International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, 2008, pp. 207–210.

[40] S. Zhao, M. Chandrashekar, Y. Lee, and D. Medhi, “Real-time network anomaly detection system using machine learning,” in *2015 11th International Conference on the Design of Reliable Communication Networks (DRCN)*, 2015, pp. 267–270.

[41] S. Kulkarni, N. Bhagat, M. Fu, V. Kedigehalli, C. Kellogg, S. Mittal, J. M. Patel, K. Ramasamy, and S. Taneja, “Twitter Heron: Stream Processing at Scale,” *Proc. 2015 ACM SIGMOD Int. Conf. Manag. Data - SIGMOD ’15*, pp. 239–250, 2015.

[42] N. Spangenberg, M. Roth, and B. Franczyk, “Evaluating new approaches of big data analytics frameworks,” in *Lecture Notes in Business Information Processing*, 2015, vol.

208, pp. 28–37.

[43] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge Computing: Vision and Challenges,” *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, 2016.

[44] J. Kreps, “Apache Kafka Performance Benchmark,” 2014. [Online]. Available: https://engineering.linkedin.com/kafka/benchmarking-apache-kafka-2-million-writes-second-three-cheap-machines.

[45] Neo4J, “Cypher Query Language Introduction,” 2018. [Online]. Available: https://neo4j.com/developer/cypher-query-language/.

[46] D. Xu, Y. Yan, E. Ricci, and N. Sebe, “Detecting anomalous events in videos by learning deep representations of appearance and motion,” *Comput. Vis. Image Underst.*, vol. 156, pp. 117–127, 2017.

8. Appendix

8.1 Recorded Data

Service Pipeline	Data Size (KB)
Raw Video Size	13000
Apache Kafka Video Analysis Topic	632.141
Apache Kafka Activity Analysis Topic	897.327
Apache Kafka Anomaly Analysis Topic	223.718
Neo4J Total Required Storage	19460

Item 1: The raw data capture of the data produced by the analytics pipeline when processing a thirteen mega-byte video, conducted as part of the use case implementation.

Array Store	8.01 KiB
Logical Log	16.96 MiB
Node Store	199.97 KiB
Property Store	1.42 MiB
Relationship Store	797.00 KiB
String Store	8.01 KiB
Total Store Size	19.46 MiB

Item 2: The itemized storage statistics of the Neo4J database after processing the thirteen mega-byte video snippet, conducted as part of the use case implementation.

8.2 Interview Transcript

8.2.1 Personal Introductory Statement

To start, let me give an overview of my work history and experience enabling me to speak with some authority with respect to the suitability and adaptability of a proposed video processing pipeline. I currently work for Cheshire Police, having recently retired after nearly three decades with West Mercia Police. During this time, I had roles including but not limited to;

Police Constable, Operational Intelligence Officer, Target Development Officer, Pro-active Crime Officer and last Detective Constable. Prior to joining the Police, I worked within the Armed Forces, spending a number of years within the Anti-terrorism unit making use of surveillance equipment including the use of CCTV. During my career, I have always been an exponent of the use of technology to assist in the investigation and detection of crime through evidence gathering and production. I have witnessed first-hand the introduction of mobile phones, computers, CCTV and other digital mediums, to assist agencies with intelligence gathering along with the counterpart avoidance methods adopted by criminals to stay undetected.

8.2.2 Transcript

During this transcript Interviewer refers to the person conducting the interview and Person refers to the interviewee.

[Interviewer]: My dissertation is based around a Real-time analytics pipeline for Smart video surveillance. The core idea is to provide a base video pipeline that can work alongside current CCTV systems, providing object detection and tracking, activity analysis and anomaly detection. This can be people walking or running, parked or driving cars etc. This would work alongside what is currently deployed and therefore we don’t attempt to store the video footage itself. This will basically let you take video footage at the camera source, convert it to a series of object locations which is then sent to server, where we can perform more in depth analysis. This can then allow anomaly analysis.

[Person]: So this could be used to identify people walking erratically or not in a normal pattern, for instance when you have a suspect who is casing a joint, they will walk up and down without any real direction?

[Interviewer]: Exactly, so we currently perform that type of analysis through unsupervised learning. Which means it will look at all the parameters we collect, for instance where someone is walking and how quickly they are moving through the camera space, and place these points within a map, looking at densities of activities. From here, we can see if someone is walking out of the norm by checking how far their activities deviate from those known clusters of behavior. So for instance people normally walk in the X, Y, of the video stream but this person is walking completely separately to that. From here, we can send alerts via email, saying we have detected this person at this location which deviates from the normal seen behavior by this much.

[Person]: This would be good for military installations, such as the Houses of the Parliament, or other big structures which might be deemed as a terrorist target.

[Interviewer]: Yes, this would be a suitable application of the pipeline. I was also considering the CCTV camera networks of shopping centers or other CCTV in the public domain.

[Person]: Ah, ok. This would be good for places highly likely to be burgled.

[Interviewer]: Yes, then the email notification, or other notification services that could be developed could then alert someone to take a close look at a particular CCTV camera.

[Person]: Yeah, with people congregating to fight outside nightclubs, that sort of stuff, this could help control rooms track down activities faster.

[Interviewer]: Exactly, so the main goal of this project is to make this possible at a large scale. So the issues we have is that we can detect people, but this is done on one camera or a limited number, but this doesn’t always scale to large networks of cameras the Police may be operating.

[Person]: So are you trying to implement facial analysis and body mapping, and that sort of thing?

[Interviewer]: So the idea of this project is to put the tools in place so that someone can come and extend the pipeline to add features such as that, depending on their use case. But the project itself is there as a basis for supporting this further development. So the main reason why you are such a viable candidate to review this is due to your ability to see what use cases this could be suitable for in the real world. We have developed a small use case, taking the single Abbey Road camera feed, from the Beetles cover, and we analyze whether all the different stages work together. We do the initial object tracking, we send this to server correctly, and we can extrapolate walking and running then detect anomalies. With the idea of this being that if we can prove this all works, with the technologies it uses, then we can show that is the Police came and wanted to deploy the system, say the Houses of Parliament, this would be a baseline for a software development company to come in and ask well what exactly is it you need to do with the CCTV? Do you want to detect walking or running, or detect someone holding a knife?

[Person]: Even vehicles driving at speed.

[Interviewer]: Exactly, the idea of this project is that someone can come to it and say, we need this behavior, we don’t want to have to write the entire end to end system, we just want to detect objects in this particular way, and then you can plug this behavior in to the rest of the unedited pipeline.

[Person] So it’s the skeletal part of it? And everyone else puts the add-ons and flesh on in the particular area that they want?

[Interviewer]: Exactly, and it has some built in stuff, so it detects cars and people out of the box, along with walking, running and standing activities. The anomaly detection, you could say an anomaly is always someone on the left hand of the camera for instance, you could develop a module that does that, however our one currently is unsupervised, so it just uses statistics to say

how far do you deviate from the norm? And we use all the parameters currently to work out what the norm is. However, if you had a team looking at this all the time, they could say this cluster over here, although it is a cluster is not the norm and you could mark that as anomalous.

[Person]: So theoretically, you could have it where you do it with body mapping to have people carrying backpacks and walking in a strange anomaly?

[Interviewer]: Exactly.

[Person]: So you could then deploy this near big sporting events, with initial warnings to direct someone in a CCTV monitoring room to look at this person or camera as they may be suspicious.

[Interviewer]: Exactly that. (Shows example of the system running with people and car tracking). To evaluate the system, we would really like to know what would the system be used for in your opinion and would it actually add benefit to the Police force? Along these lines I have a few structured questions to format the interview, the first being: the idea of the system is to be able to provide these real-time insights into what is occurring within the video stream, do you think this would help deter crime? What benefits do you think this would bring to the use of CCTV, and would it help the Police force to have something like this in place?

[Person]: Well obviously, not just the Police force, the Prison Service, people who monitor large buildings would all benefit from this. With respect to the Police, we often get cluster problems, for instance we get a series of crimes with vehicle crimes, where certain streets in an area you get multiple cars broken into and vehicles stolen. So we can put static cameras out, the council can, on any lamp post, in any street, to monitor anti-social behavior, whatever; so if you’ve got software and one of these cameras you can put out at a minutes notice, you can have it to look for these specific patterns of behavior, and this would be great if you have a link to warning

system. You can detect crime quicker, identify patterns of behavior in a shorter time span, before that pattern of behavior goes out of control, which it often does.

[Interviewer]: So would you say that the Police's current reaction to crime is often to deploy static cameras, so the improvement would then be to not just use these cameras for tractability but to actively use them to stop criminals?

[Person]: Yes, we often use something called rat-traps. So if you have a series of cars being broken into, we often put out a dummy car. It will have valuables in the windscreen, and be fitted with a camera. The offender would then come along, break into the car, the video would capture him, and then afterwards we would find out who the offender was, if we were lucky. Often, they could be masked up or whatever. I've enhanced images, and we have arrested people, to stop a series of crime. If this is done quickly, and people monitoring close by, you can get people in the act. We do often have a few of these cars out, with only a single team monitoring them, so if we could have alerts come through in real-time saying we detected someone we would then be far more likely to catch them in the act. If we don't catch them in the act, we have to go to their address if we track them down, arrest them, and most likely they still have the items in their possession which further corroborates the case. If we have something active we can go out and catch them virtually in the act.

[Interviewer]: Awesome, so what sort of scale of operation would you use for this, how many cameras would you look at deploying on average?

[Person]: It depends on the premise of the case. If you have an industrial state with lots of different units, some high value some less value. If it's in a remote area and the response time to it is slow, obviously criminals are not stupid, and they do a time and look and cause an alarm to see the response time. They then know the response time and know where police are at what

time, and time there crime to make sure they are out before Police arrive. If you can put in a camera system, or adapt an existing camera system, then you've got a purpose built monitoring system. Because you get patterns of crime. If an industrial estate gets hit, it is often found that in 12 months' time they get hit again. Criminals often wait until the insurance pays out from the first robbery, the suppliers will restock with their new televisions etc., and then they will rob them again. So if you've got that model, you could put a system in pre-emptively to possibly catch them by placing officers nearby during high risk times, as criminals are people of habit, so there are lots of ways in which the system could be used to help catch them.

[Interviewer]: Awesome, so if this system was to roll out to, let's say the Police force, what sort of typical scales would you see if used on? Would you say 50 – 500 cameras a time? Or more?

[Person]: Well I think 500 is excessive for most Police operations, however if you're the government and wanted to put cameras up for anti-terrorism, or the problems we have at the minute with regards to knife crime or murders, you could put all those cameras out if they could be moved and put out easily. Then have that on all the time in this high value areas, maybe permanently. But for a Police force, I would suspect that they would have a few near hard targets, for instance if you have Army bases in your area, or high profile MP's who maybe targets, which could then be monitored. They'd probably be more in a pro-active approach though, so they may go out to jobs and place them up there.

[Interviewer]: So for that sort of system to work, they would want to be able to have pre-built models or configurations that are made for, for instance, a terrorist threat, or anti-burglary configuration?

[Person]: I suspect that would be ideal. If you have a plug and play system that can just connect to the Cloud, and it goes to your monitoring station in Police headquarters or wherever, then that

would be ideal. However, you have to think beyond that, think Airport systems or motorway systems for accidents, you might be able to pick up erratic driver falling asleep at the wheel. You might be able to stop an accident before it has a chance to happen. Especially with resources and traffic officers being limited, you haven't got the amount of people out on the road to monitor this, so if you can do anything to reduce fatalities in that would be a significant step forward.

[Interviewer]: So for that use case, you would look to deploying the system with some modules to identify normal driving? Then the output could then be a best placement of the officers you do have to catch these people?

[Person]: We've all been on the motorway witnessing drivers screeching well over the speed limit, undertaking and being dangerous. You could very easily make up a model that detected this kind of behavior.

[Interviewer]: Then the output of this would be a best placement of officers in order to catch these people?

[Person]: Well that, or, it would be like the Ambulance scenario where we place the vehicles at the half way point for the areas they cover, rather than at an Ambulance station. This allows an Ambulance to react to an incident in any area, say within 30 minutes, however if it was kept at the station it may take an hour to get to an incident far away.

[Interviewer]: For sure. If you were to use this system, out of the box it can do basic people and car detection, and where they go through frames, what would you like to see the framework used alongside? For instance, traffic cameras, Police radios, Police body cams.

[Person]: In the real world today, we have ANPR (automatic number plate registration), which can pick out registrations of interest, so something like that would be in straight away. Beyond the realms of possibility, facial recognition would be a great use case. If the person is linked to a

system of wanted people, were not talking about innocent people here and monitoring everything they do, but if you have someone really wanted by Interpol for example, then the system could give you a flash warning saying they are in this location. When that warning happens, the system then tracks him to keep hold of him until people could respond. You would have to have certain safeguards, such as freedom to privacy and freedom of movement, however if you wanted to use this for anti-terrorism you could place the images of wanted people into the system, and if they came into the system it would notify you. This could then link up to international agencies, if they all ran the system, and allow you to track people through foreign agencies, not just domestic Police forces. The ramifications can be scary though, in a Big Brother kind of way.

[Interviewer]: So one of the use cases I currently thought of, as the system doesn't currently store the video only the extrapolated data within a graph database, which means we store relationships and nodes between people. frames and activities (shows example of Neo4J database), this means if you were tracking someone and wanted to know who they worked alongside, you'd be able to query that here and see the relationships a person has to other people commonly seen in the same frames they are identified within.

[Person]: Just think of this not in a crime scenario, but in a missing person point of view. The Police could utilize a system like this to find, for instance, Madeline McCann. With the help of facial recognition, and what she might look like now, and that was imbedded in systems everywhere, the likelihood is the system will get a hit. It might be someone similar too, with lots of false positives being produced, but it's still a strong chance of finding her.

[Interviewer]: So it sounds like Facial recognition would a very critical feature?

[Person]: Yes, just think people can wear disguises, but they can't disguise their build or the way they walk, and a system that could support all the other analysis and be able to be extended to support these kind of features would be massively beneficial to the Police force.

[Interviewer]: So facial recognition is one, but also you mentioned that body mapping and gate analysis would also be really useful features?

[Person]: Yes, how a person walks is very unique and it's very hard to disguise this.

[Interviewer]: So from a camera point of view, where we extrapolate this information is the video analysis point of the system, as storing the video is very expensive to do, along with causing a lot of processing overhead, so when we do information gathering a good area to investigate is the ability to identify specific people through these methods?

[Person]: Well the thing is, if you have a lots of cameras out, say in London, if you can detect anomalous behavior, you don't have to have the system constantly identifying people's faces if you didn't want to. But once that strange behavior is detected, you could have a different system latch on straight away and begin facial recognition within the camera feed the anomaly was detected within. As you might then be able to discount the anomaly if the facial recognition doesn't match any known targets.

[Interviewer]: So a good suggestion is maybe on an anomaly not to email someone, but to turn a separate system on that is able to perform some more in depth analysis before alerting a real person?

[Person]: Snapshot picture, send it in, automatically gets matched against the database, then if that matches you have a corroboration and you can alert someone.

[Interviewer]: Ok awesome.

[Person]: In respect to the data side, we always have problems with data size and override problems, as there is only such finite data you can store from these CCTV systems. There is always going to be a problem of overwrite. We've put out cameras for witness protection, in cases like child protection cases, and the DVR systems are made in such a way, that you have to download in real-time. So we have had to go and take the DVR systems away and replace them, and these cost like 5 or 6 thousand pounds apiece. So you can imagine doing 3 or 4 or 5 of those over the last few months, it's an expensive things. So if you can reduce the data, this would give a massive improvement over money spent on a case. Current systems do try and reduce this, by only recording changes in frames, so if someone moves across a camera it only records the changes in pixels for when they move, and the rest of the image is static. But if you could reduce it further, and make it searchable, this would be another great improvement.

[Interviewer]: So a big focus of this project is to get away from storing all this video, and it sounds like a lot of the work you conducted was to go out and collect this video, and search the video. The idea of this, is because were storing the data in structured text (cut off).

[Person]: Well what we always called them was events. Where we have something dramatic happen the systems often store the event in a separate database, such as an unusual movement, so then the rest of the hard drive can be overwritten and the events can be investigated and kept/removed as needed.

[Interviewer]: Awesome. This framework can be deployed easily to existing CCTV, there's a small program that communicates with the video feeds itself, which must then have access to the internet, and the rest is deployed to the Cloud. That basically does the object tracking and turns the video into the analysis. With this being in the Cloud, and easily setup, if you worked in the Police in the IT side of the organization, we hoped it would be easy to deploy and use. So the

question then is, do you think it would be beneficial to the Police to offer this out to small businesses so you can use jus the analysis to track and aid in crime prevention?

[Person]: Well the thing is, you have intelligence units and systems in all Police forces and the military. Business model wise, Police forces always outsource a lot of their stuff. We would always go out to an outside agency, and get the cheapest quote, so in that respect they would probably go to an outside agency. If you had a very specific event, you have to get a RIPPER, which is a court order allowing you to put up the CCTV cameras, so it would be hard to stream this information from small businesses to the Police without the correct permissions.

[Interviewer]: So with the out sourcing, if someone approached the Police, and said if you are streaming all this CCTV for the Police, would you say there's a benefit to having a system like the one proposed, for this real-time feedback, even in its current state?

[Person]: Oh, definitely. What I'm saying is there's a market for this because often Police officers and the forces want to target crime areas for a specific reason. There's a lot of people sat in police monitoring rooms, I've sat in them many times with multiple monitors all night, so if you have a way of remotely having that system in place, and link into it wirelessly over the Cloud, as long as you have legislation, and then set parameters for the activities you want to monitor, you don't have to watch it constantly. You could have it linked to your mobile phone, at the first sign of anomalous behavior you could then get up and monitor the CCTV streams.

[Interviewer]: The idea of this system is completely that. So that sort of goes back to the idea of having pre-made configurations for burglary, terrorism etc. that they can then deploy as necessary.

[Person]: If you think of it not only pro-actively but after the fact. If you have an area where you know a murders taken place, and you've got data from the CCTV, and you had those models that

could run through the video without viewing and identify the anomalous behavior and then flag only those areas for viewing, you would save hundreds of man hours. Even if you didn't find anything, it could save time, if it was a case of a kidnapping of children where its time restrictive, this could be the difference between life and death. With the system you've shown being modular, this would be a great way of reusing parts of it for more than just the real-time analysis.

[Interviewer]: So the benefits you'd see from this framework, would be the ability to deploy the individual services to meet unique use cases?

[Person]: Yes this would give great flexibility in where we could use the system.

[Interviewer]: Awesome. So the system currently can track and detect people and cars, and their movement patterns, and detect anomalies based on this. With that in mind, where would you like to see the system used? Where would you see the most benefit if the system could be placed in there tomorrow?

[Person]: Well there's lots of scenarios, a big one I've been to lots of times is where people have died after being involved in a fight. So if you could detect a fight or aggressive posturing, you could have warnings flag before people ring in and say there's a fight going on, allowing you to already be reacting to the event. There's lots of pro-active things that you could do, such as systems outside schools, such as people hanging around when they shouldn't in order to look after child welfare. A normal family playing is completely different someone skulking around.

[Interviewer]: We've evaluated the system with a single camera to make sure it does work, to really evaluate the pipeline from your point of view, what would the system have to have for you to deem it worthwhile to bring in?

[Person]: Ok let’s give one example, if you’ve got a way of detecting vehicles speeding up or driving in an area driving should not occur, this could help stop or respond to loss of life events and terrorist activities. Furthermore, this system could be linked to automatic bollards which would be raised when the system alerts to this anomalous behavior. So in this case, the system is currently ready to be deployed to that kind of scenario with more benefit being given if we could extend it to interact with those kind of in place bollards. This could reduce the amount of people killed or injured so quickly. That’s the same for terrorism etc., it could be 9 times out of 10 that the anomaly is normal behavior, but if it gets it correct that 1 time you could save a hell of a lot of lives like that.

[Interviewer]: Fantastic. So around this, the system works separately but requires that integration with the CCTV camera feed, would you say a system like this would be compatible with what the Police force currently has operating?

[Person]: I know from personal experience, we have control rooms in different places that are Police, council and community monitored. There is people in place and locations, and there are ones in other places that aren’t up to that standard. You would be able to do this in some areas, particularly mentioning London, but other areas may require better equipment being put in place. But certainly vehicle movements with traffic monitoring cameras, you could put your system in place almost instantly. Obviously if you wanted to roll it out generically, everyone’s system is different, so you would have to account for that in your integration component.

[Interviewer]: Ok great.

[Person]: In the days of the troubles (referring to the IRA) this sort of system would be ideal. It could inform the guards and foot patrols around military establishments, of where cars are located and to go and check them. The guards may not know there’s a car on the other side of the

hill, and even the anomaly detection you've spoken about so far, would help inform and guide patrols.

[Interviewer]: Awesome, I've not thought of that.

[Person]: These always reduce the possibility of something happening, and for that reason it would be very beneficial to use a system like this.

[Interviewer]: If someone was to come and present this pipeline to yourself, or to the Police force in general, as a tool to build upon, would you see it as a viable avenue for the Police to pursue?

[Person]: If you have large metropolitan cities like London, if you had a system in place like this, you would be able to get that work straight away with the current system. For other areas, I think you would have to have a specific model, such as burglaries, erratic driving etc. So if you have a model that is able to detect these things, and put this system in allowing officers to respond, yeah I think they'd probably snap your hands off to have something like that.

[Interviewer]: And what would you say the biggest feature the system would have to have to be so appealing?

[Person]: It would have to be structured in a way in which you have set parameters, such as burglary models, that type of thing. So that you could put that model in on a mobile camera, or in an area with high crime. If you could have a system where we could say we want to hire that model for this period of time, because the crime is costing us thousands, then this would be massively appealing.

[Interviewer]: To summarize then, for the Police to adopt this, we would want plug and play models for specific use cases, but the more likely use case over Smart Cities and thousands of cameras, is the ability to deploy the system as part of these unique systems to fight a specific crime.

[Person]: I would put them in places of cluster crime problems, such as vehicle crime and anti-social behavior, alongside the less widely used Smart City adoption, which obviously would only occur for Police divisions working in Metropolitan areas.

[Interviewer]: You also eluded to military establishments wanting to adopt it, and possibly link the data from all 3 deployments.

[Person]: Well ideally the system would want to be used worldwide in the end. If you have a car picked up on a motorway, maybe its stolen, and its transported abroad you still want to be able to track it either before it is able to be transported, or once it reaches its destination.

[Interviewer]: Ok, you mentioned you might want caching to occur, so if we do spot something we save that clip of video?

[Person]: Well not all actions may lead to something, so what we want is an intelligence picture to be built up. Camera footage can be retrieved, but the intelligence enables us to act.

[Interviewer]: So would you say the Police force would move towards wanting this type of smart analysis?

[Person]: Certainly initially the big Metropolitan areas would immediately want this. Other areas of the country, forces are combining to combat their low resourcing, so it would be uncertain as to whether they would be in a position to pursue such a system. But you could still have your mobile units going out and coming in for the specific hot spot problems. But if you can do it any kind of way where you can link to any sort of CCTV system, then it’s feasible. A plug and play system is a significantly more likely to get attention and input from the Police over a generic system.

[Interviewer]: So you have two main avenues you see this system being used, in specific use cases with directed models, and in the blanket application as the systems current state to large Smart Cities such as London?

[Person]: Yeah, especially for airports and major military installations also. Anywhere where theirs is a known target. I presume the hard part is the getting these models, but there is existing CCTV from the past so it's completely feasible to build some models of this footage to get some useful models for these cases. They may not be exact, but if you detect one 1 out of 100 terrorism acts or bomb threats, and it manages to deter even one, then it's paid for itself.

[Interviewer]: Have you heard of the Police adopting anything like this at present?

[Person]: No, the only thing I've heard is remotely like this is the CCTV that reduces its storage size by only saving pixel movements, but that's nothing like this. Obviously, I am only speaking from my contact with the Police and there could easily be projects ongoing that I am unaware of. I'm sure security agencies have something like this, but that is separate to the Police. Obviously a lot of Police decisions come down to cost still.

[Interviewer]: Currently the base framework is free, and open source as it's meant as a tool for helping aid these further development projects.

[Person]: Well I can see that making it massively viable, and help the Police in improving its use of CCTV. I can see lots of applications for the system for sure.

[Interviewer]: Thank you for your time.