Uncovering the Misuse of AI Services

Sumaiyah Y. Kola
Clare College

UNIVERSITY OF CAMBRIDGE

A dissertation submitted to the University of Cambridge in partial fulfilment of the requirements for the degree of Master of Engineering in Advanced Computer Science

University of Cambridge
Computer Laboratory
William Gates Building
15 JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom

Email: sk940@cam.ac.uk

June 3, 2021

Declaration

I Sumaiyah Y. Kola of Clare College, being a candidate for the Part III in Advanced Computer Science, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

Total word count: 11739

Signed: S. Y. Kola

Date: June 3, 2021

Uncovering the Misuse of AI Services

Abstract

Artificial Intelligence as a Service (AIaaS) offers pre-built AI models to customers looking to leverage AI capabilities with minimal cost and risk. This thesis is predicated on the fact that various pre-built models are being offered as a service, on demand, often with little oversight. Given the generic functionality of the AI models available, and the lack of oversight from service providers, there is a large scope for misuse. This thesis implements and evaluates a variety of mechanisms for uncovering possible AIaaS misuse that can identify 'signals' and 'signatures' in customer usage data that could indicate potential bad behaviour. It explores the feasibility of various approaches that would enable service providers to be more proactive in ensuring their AI services are used responsibly. It examines two specific scenarios of misuse, namely the misuse of facial recognition software for surveillance and the misuse of language models for processing harmful speech, after first exploring a generic approach to usage landscaping and misuse detection. Given a lack of existing AIaaS usage data, the thesis also involves generating synthetic, realistic traces of AIaaS data from analogous, real-world datasets to indicate various possible scenarios of misuse. Although there is currently no one-size-fits-all approach to AIaaS misuse detection, the thesis concludes that existing algorithms can be used to build efficient generic, scenario-specific, and misuse-specific methods for monitoring.

Word Count: 11739

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Artificial Intelligence as a Service (AIaaS) refers to the growing trend of third-party vendors offering AI solutions out-of-the-box. Given the increasing interest in AI, generic functionality of pre-built AI models, ease of accessibility, and lack of oversight, there is a large scope for misuse. There is the potential for customers to utilise these services to support problematic, controversial, or possibly even illegal applications. Service providers should therefore monitor the use of their AI services to uncover particular situations that may warrant further investigation.

In this project we propose, and demonstrate the feasibility of, a range of techniques to analyse customer behaviour when using AI services in various scenarios. The aim of each technique is to uncover different 'signals' and 'signatures' in the usage data that may identify potential situations of misuse. Given the growing public discourse surrounding accountable AI and the increasing popularity of AI services, we hope to attract greater attention to the concerns surrounding AIaaS and propose a range of potential solutions.

1.1 Motivation

In general, artificial intelligence (AI) refers to a collection of algorithms that allow machines to solve complex tasks by generalising over large amounts of data. The International Data Corporation estimates that global spending on AI will exceed $110 billion by 2024 [1]. To remain competitive, businesses are looking for inexpensive and convenient ways to integrate AI into existing applications, products, and services. Many of these businesses lack the technical expertise, compute resources, or data required to develop AI systems of their own [2].

In response to this, AIaaS providers offer customers powerful and customisable AI functionality on-demand, without the otherwise required computational expenses and in-house expertise. This dramatically increases the accessibility to AI, as customers can easily and inexpensively obtain pre-built models 'off-the-shelf'. Unsurprisingly, the tech giants (Microsoft, Google, Amazon, and IBM) are the most prominent AIaaS providers, given their huge cloud infrastructures, access to data, and powerful AI platforms. We can only expect interest in AIaaS to grow as the barrier to entry to AI lowers.

While AIaaS providers make complex and sophisticated AI functionality readily accessible, they typically provide minimal supervision over customer usage. As a result, there is a large scope for these generic AI services to be deployed unchecked at scale, potentially supporting harmful applications. A sophisticated facial recognition model could be misused for population surveillance or to invade the privacy of a single person. The same text-to-speech software might be used to build an accessible application, or to process hateful speech. While there has been a lot of discussion surrounding the potential dangers of AI as a whole [3], when delivered as a service, it has received much less attention.

With the rise of AIaaS technologies, there is an incentive for service providers to protect their reputation and manage their potential obligations and liabilities. As mentioned, the pre-built AI models could easily be used to underpin and support controversial or problematic applications. If it is disclosed that a controversial application is 'powered by' a specific service provider, the public backlash and negative publicity that follows might severely damage their reputation and, as a result, their revenue.

Many organisations and applications may come to rely on AIaaS as the primary means for implementing AI. Given the growing public concern surrounding emerging AI technologies [4], there is a compelling case to be made that if service providers are providing the functionality necessary for these applications to run, they must also ensure that their customers are using the services responsibly. Furthermore, new regulatory frameworks appear to be forcing large service providers to assume a greater responsibility for their products and how their platforms are used – see proposals for the EU AI White Paper [5] and Digital Services Act [6]. In conclusion, as interest in AI technology grows, monitoring the usage of AIaaS is a problem that deserves attention.

1.2 Related Work

Intelligent systems are rapidly being deployed by state agencies, corporations, and individuals to solve a myriad of problems. As AI systems in general become more widely used, it is important to understand how they may fail or be abused. The AI Incident Database (AIID) is a public, systematised collection of over $1,000$ real-world incident reports, caused by the deployment of intelligent systems [3]. According to the authors, the AIID may be useful for product managers defining product requirements for new systems; being aware of previous incidents may help to avoid future ones.

It is possible for AI service providers to mitigate against particular types of misuse, both Microsoft and Amazon currently enforce rate limits for their face and text APIs. In the case of AI services, however, the entire business model is predicated on service providers developing generic models that can be used in a variety of scenarios. As a result, service providers cannot mitigate against all potential misuse scenarios for every API without compromising the generalisability of the AI models. Instead, we propose that service providers monitor customer behaviour in order to detect and respond to any potential misuse.

In the field of computer science, misuse detection is not a novel concept.

Credit card fraud detection software is widely deployed and identifies patterns of typical and atypical customer behaviour, as well as anomalous individual transactions, to highlight potential fraud. Intrusion detection systems (IDS) are also popular in the field of computer network security. An IDS is a programme that continuously monitors a network for malicious activity. We apply principles from current misuse detection approaches in this project to detect instances of AI misuse in various scenarios. In particular, we explore the potential of cluster based analysis techniques for anomaly detection.

We build on the work of Javadi et al. who, in 2019, provided a conceptual framework for monitoring AIaaS usage for misuse, proposing several solutions [7]. The authors offered technical implementations for some of these solutions earlier this year, in particular concentrating on generic usage landscaping as well as uncovering instances of potential human surveillance [8].

We expand on this previous work and perform a comprehensive evaluation of the performance of, and overheads associated with, a variety of misuse detection methods in different scenarios. In the context of usage landscaping, we evaluate the efficacy of different algorithms for extracting distinct behaviours. We expand on misuse detection in the surveillance exemplar by comparing a variety of approaches for detecting both large-scale and small-scale surveillance. We also investigate a new misuse scenario in which natural language processing services are used to process potentially hateful speech. The next section contains a detailed outline of the project.

1.3 Approach

AIaaS providers offer customers accessible, web-based APIs for powerful, complex models. Customers can easily, and affordably, 'plug' state-of-the-art AI functionality straight into their applications. In general, an API request will specify the required model along with the raw data from users, and the service provider will process the request and return the model results. Figure 1.1 depicts the route of this data when building an application that involves a document translation model.

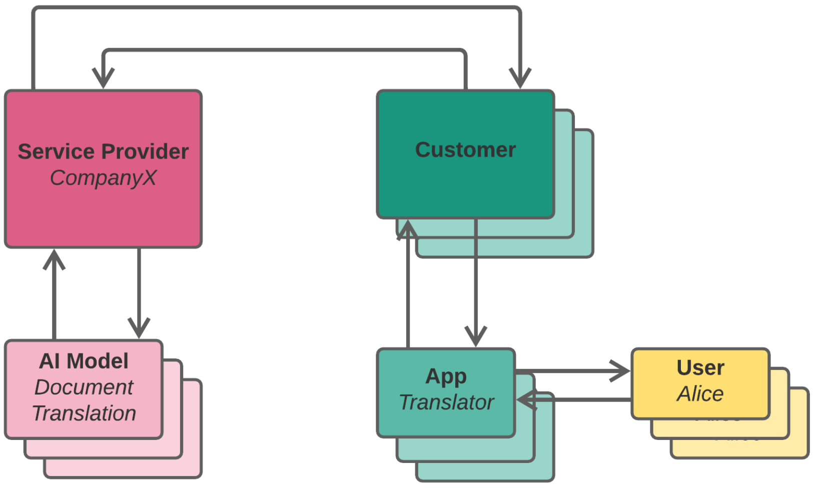


Figure 1.1: A simplified illustration of how data travels when using AIaaS

In terms of monitoring for misuse, there is a decision to be made in terms of which data is processed and to what extent the data is processed. Service providers already produce transaction logs that contain information about service usage, for billing purposes. Typically, transaction logs only include transaction metadata and no user data. Service providers can easily utilise these existing transaction logs to identify suspicious usage patterns. However, without access to any contextual information, such as model inputs and outputs, it may be difficult to detect misuse in a given case, e.g. when using face analysis software, without looking at the images being processed. Looking at user data for the purpose of monitoring misuse, on the other hand, raises additional concerns regarding the privacy and processing of user data. A full discussion of these issues can be found in section 5.2.3.

We assess the performance and overheads associated with a number of different misuse detection methods, each of which differs in the degree to which the service provider processes the data. We look at the advantages and disadvantages of processing the data to varying degrees. For the sake of completeness, in each chapter we also focus on a different real-life scenario for potential misuse. As AIaaS is still relatively new, there is a lack of public data detailing customer usage or misuse. We first leverage analogous, real-world datasets to generate realistic traces of AIaaS data.

In chapter 2 the misuse detection methods first examine the transaction logs, with no information about the context of the AI service or user data. We perform a generic analysis of the usage landscape of customer behaviour to extract patterns, group behaviours, and detect anomalies. Next in chapter 3 we explore misuse in the context of surveillance and process data generated as a byproduct of using an AIaaS. In this case, the misuse detection methods analyse preprocessed input data from the user in the form of embedded face vectors. There is no additional manipulation of the user data here, merely analysis. Lastly in chapter 4 we examine misuse in the case of language models, specifically for the detection of harmful speech. The raw text data from the user is processed, specifically for this misuse detection, before being passed into the hate speech detection classification models. Chapter 5 concludes with a summary of our findings and the broader implications of this work.

1.4 Key Contributions

In this thesis:

1. We discuss the issues of AIaaS misuse and the importance of monitoring for such misuse, providing a justification for this work.
2. We evaluate the potential of generic usage landscaping, blindly extracting behavioural patterns from usage data, in the context of misuse detection.
3. We illustrate the feasibility of technical approaches to misuse monitoring in a variety of specific real-life scenarios, including image and text processing services.
4. We explore the practical considerations and overheads associated with different approaches to misuse monitoring in the different contexts.

5. We discuss the broader implications of misuse monitoring and ideas for future, potentially multidisciplinary, research in this field.

1.5 Summary

The project is driven by the fact that numerous, generic, pre-built AI models are available as a service, on-demand. Integrating AI functionality into applications is now easier, and more scalable, than ever. Given the lack of oversight from service providers however, there is a large scope for misuse. We explore the feasibility of different methods to uncover the misuse of AIaaS.

Chapter 2

Generic Usage Analysis

From facial recognition and speech translation to content moderation and medical transcriptions, AIaaS providers offer functionality for a vast range of generic and specialised applications. With a large variety of services accessible, developing mechanisms to uncover general patterns of customer behaviour would be beneficial. In this chapter we conduct a feasibility study of such mechanisms, using a cluster based analysis to explore and understand the landscape of customer behaviour, and highlight areas that may warrant further attention.

As mentioned earlier, AIaaS providers already generate information containing service usage metadata in the form of transaction logs. The logs are effectively a record of the behaviour of the customers over time. Our goal is to use these transaction logs to build a generic misuse indicator that can uncover different classes of customers, based on their behaviours.

Generic methods that aim to explore the usage landscape and uncover potentially abnormal behaviours are known as discovery-based methods. Alternatively, trait-based methods are context-specific, and identify known patterns of behaviour. Building trait-based indicators for specific misuse scenarios requires prior knowledge, in the form of data. As the field of AIaaS is relatively new, there is little data available that reflects specific scenarios of misuse.

Additionally, each misuse scenario would require a specific trait-based indicator. As a result, running all trait-based detection algorithms, in the first instance, would be impractical and expensive.

We instead choose to implement a generic, discovery-based misuse indicator. Uncovering general patterns of customer behaviour can help service providers to initially identify potential areas of concern and determine where further investigation, with trait-based indicators, is required. In this chapter we investigate the feasibility of generic, discovery-based indicators to explore the usage landscape and analyse the behavioural patterns of customers.

2.1 Approach

We implement generic mechanisms to explore the landscape of customer behaviour, through usage logs, to uncover instances of potential misuse. Given the lack of public AIaaS usage data, we first construct realistic logs of AI service use, from analogous real-world cloud service usage records.

Outlier detection is the process of identifying unusual patterns or rare occurrences of behaviour that differ substantially from the norm. This abnormal behaviour may raise suspicions and warrant further attention from service providers. Several strategies for detecting anomalies have been proposed in literature [9] including Hidden Markov models, neural networks and clustering algorithms.

There are a wide range of AI services available, as well as a diverse range of customers. As a result, ‘normal’ behaviour in each transaction log can look different; common patterns of usage for one service could be cause for concern in another. Some usage landscaping and anomaly detection approaches require you to define what constitutes abnormal behaviour. Cluster analysis based approaches, on the other hand, allow you to group data in categories based on similar behaviours, with no prior knowledge about the data.

In this chapter we consider a generic, exploratory approach, to usage landscaping with cluster analysis based detection models.

We first conducted an exploratory analysis of a range of clustering algorithms, including DBSCAN, Gaussian Mixture Models (GMMs) and k-means. A full description of the DBSCAN algorithm can be found in section 3.2.1. Both GMMs and k-means require the specification of k, the number of clusters. Although the clustering performance of both algorithms was similar, the time complexity of GMM was substantially larger. The DBSCAN algorithm, on the other hand, independently computes k. Although the algorithm is efficient, the hyperparameters are very sensitive to scale and drastically affect the clustering performance of the algorithm.

As mentioned, the usage logs can take different forms and contain a range of behaviour. As a result, the approach for examining the usage landscape must be consistent, robust and generalisable. Following this, we opted to evaluate the performance of two variants of the k-means clustering algorithm, described below.

2.2 Clustering Approaches

In order to group different types of behaviour and identify potential outliers, we perform a cluster based analysis on the usage logs. There are a wide variety of clustering algorithms available, each with a different 'cluster' definition and method to efficiently identify clusters.

We explore the unsupervised k-means clustering algorithm. The k-means algorithm requires the number of clusters to be specified as a parameter. In this section we explain two variations of the k-means algorithm, Euclidean k-means and Dynamic Time Warped k-means.

2.2.1 Euclidean k-means

k-means is an extensively used, centroid-based, machine learning algorithm that aims to partition the data into k clusters. Each cluster is associated with its center, or 'centroid'. A cluster centroid is the arithmetic mean of all the points belonging to that cluster. Optimal clustering with k-means is achieved when each point is closer to its own cluster centroid than any other centroid. In the Euclidean k-means algorithm the similarity between data points is measured using the Euclidean distance between them.

Initially, the cluster centroids are randomly selected. The positions of the centroids are then iteratively optimised until either the positions stabilise or the specified number of iterations is reached.

2.2.2 Dynamic Time Warped k-means

Usage logs are a registry of events, captured at regular time intervals, between the service provider and customers. The distance metrics used in standard clustering algorithms, such as Euclidean distance, are often inappropriate for such time series data, as they are invariant to time shifts. The Euclidean distance algorithm only computes the distance between coordinating pairs of points in each series.

An alternative approach is to replace the Euclidean distance metric with a metric designed for comparing time series, such as Dynamic Time Warping (DTW) [10]. DTW calculates the smallest distance between all points in the time series, comparing each element in time series X with each element in time series Y. $DTW(X, Y)$ is calculated as the square root of the sum of squared distances between each element of X and its nearest point in Y. This creates a temporal alignment between X and Y that minimises the Euclidean distance between them.

We can extend the popular k-means algorithm to work with time series data using a DTW distance metric instead of Euclidean [11]. DTW is first used to group time series of similar shapes. Cluster centroids are computed as the barycenters with respect to DTW. In the DTW space, a barycenter is the average sequence of a set of time series.

Regardless of temporal changes among the members, cluster centroids now imitate the general shape of cluster members. The Python tslearn library offers an option to use DTW as the core distance metric in the k-means algorithm [12]. DTW, computed with dynamic programming, has a time complexity of $O(|X||Y|)$.

2.3 Experimental Setup

Before evaluating the performance of the clustering algorithms, we first construct the AIaaS usage logs from real-world customer data.

2.3.1 Synthetic Dataset

While there is a lack of publicly accessible usage data from AIaaS providers, a dataset containing real-world usage data for a cloud service has recently been made public by Microsoft [13]. It contains customer usage of Microsoft's Azure Function as a Service (FaaS). Similar to AIaaS, FaaS also involves invoking specific functions through a web-based API. As such, in terms of mapping a usage landscape that is grounded in reality, we may consider this dataset reasonably analogous to AIaaS customer usage. We use this dataset to generate logs.

The dataset contains real-world customer metadata collected over a period of 14 days between July 15th and July 28th, 2019. For each day, the dataset includes the:

- Function invocation count (FIC), per function, per minute over 24 hours.
- Function execution duration (FED), per function. The data is averaged and presented as a set of weighted percentiles and summary statistics over 24 hours.
- Application memory allocation (AMA), per application. The data is averaged and presented as a set of weighted percentiles and summary statistics over 24 hours.

Each customer can be associated with one or more applications and each application is associated with one or more function calls. A full schema of the dataset can be found on GitHub [14].

Preprocessing

The FIC data is provided at minute-level granularity. The FED and AMA data, on the other hand, are provided as a set of weighted percentiles every 24 hours. These percentiles are representative of the underlying probability distribution of the data. The first step in preprocessing is to regenerate representative FED and AMA data, at minute-level granularity, from the percentiles.

The FED is provided, per function, as a set of percentiles and summary statistics over 24 hours. We assume that the data has a normal distribution. In order to sample this distribution the mean and standard deviation is required. Although the summary statistics for FED provide the mean for each function call, they do not include the standard deviation. Fortunately, the standard deviation can be estimated using known heuristics and the data given.

The empirical, or three-sigma rule, claims that nearly all values lie within three standard deviations of the mean. In particular, it predicts that 68% of observations falls within the first standard deviation ($\mu \pm \sigma$), 95% within the first two standard deviations ($\mu \pm 2\sigma$), and 99.7% within the first three standard deviations ($\mu \pm 3\sigma$). We can compute an approximation for the standard deviation by substituting in the cumulative percentages, or percentiles. This process is repeated on the AMA dataset.

The minute-level FIC, FED and AMA data is then standardized. Categorical attributes are numerically encoded to render them understandable to the models. The remaining numerical features are log normalised so that the attributes have similar magnitudes. This prevents features with larger values from carrying more significance in the network.

We construct a variety of logs from the preprocessed FIC, FED, and AMA datasets for experimentation. The minute-level granularity FIC, FED and AMA datasets over 7 days are combined with an ‘outer’ join to prevent data loss. Empty cells are assumed to indicate no function calls on that day, and are subsequently replaced with 0. Unlike FIC and FED, AMA is only available per application. To compute AMA per function, we assume a uniform distribution and distribute the memory usage value uniformly across the functions of each application. Lastly, we generate logs at different granularity levels; from the 1 minute-level FIC, FED, AMA data we compute the average values every x minutes for x in $[1, 10, 60]$.

2.4 Results

In this section we evaluate the performance of a cluster-based analysis on the generated usage logs. We begin with an overview of the performance of the Euclidean k-means algorithm on the entirety of the dataset. This is followed by an analysis of the behavioural patterns from a single customer. Lastly we compare the performance of the Euclidean k-means algorithm with the Dynamic Time Warped k-means algorithm, developed in particular for the analysis of time series data.

Overview of all Customer Behaviour with Euclidean k-means

Table 2.1 contains an overview of the performance of the Euclidean k-means algorithm on the combined FIC, FED and AMA data sampled every 1,10 and 60 minutes. Each log comprises $62, 130$ unique entries. In each case we see a consistent pattern of one dominant cluster. The majority of the remaining clusters are medium in size, with a few small clusters towards the end.

Further inspection reveals that a large proportion of the entries in the log are sparse, with only a few function calls throughout the course of the seven days. The dominant cluster encapsulates this behaviour. The remaining clusters represent different, smaller, categories of behaviour. We see that changing k effectively adjusts the sensitivity of the algorithm and specificity of behaviours captured in the clusters. This allows for more micro or macro analysis of behaviour, as needed by the service provider. As k increases, the size of the dominant cluster shrinks as more detailed behavioural categorisation becomes available.

Data Granularity	k	Cluster Sizes	Time (s)
1 minute	2	[49472, 12658]	609.1
	3	[48439, 7857, 5834]	648.1
	5	[45146, 5540, 5376, 3743, 2325]	976.5
	9	[41972, 4755, 4317, 3420, 2988, 2284, 1623, 455, 316]	1654.6
	30	[35777, 3386, 2855, 2291, 2253, ..., 190, 155, 150, 120, 79]	5057.9
10 minute	2	[51872, 10258]	22.7
	3	[49563, 8030, 4537]	45.8
	5	[46292, 6351, 5514, 2691, 1282]	68.4
	9	[44725, 4665, 3401, 2515, 1967, 1543, 1387, 1069, 858]	123.5
	30	[41209, 4464, 1726, 1325, 1130, ..., 192, 180, 144, 81, 52]	305.8
60 minute	2	[48269, 13861]	6.3
	3	[45468, 11012, 5650]	9.7
	5	[41647, 7405, 5249, 4429, 3400]	23.6
	9	[39769, 5624, 3940, 3222, 3069, 2234, 1738, 1614, 920]	25.9
	30	[33405, 4560, 2322, 1698, 1596, ..., 299, 223, 193, 185, 174]	52.1

Table 2.1: Euclidean k-means on combined FIC, FED and AMA data

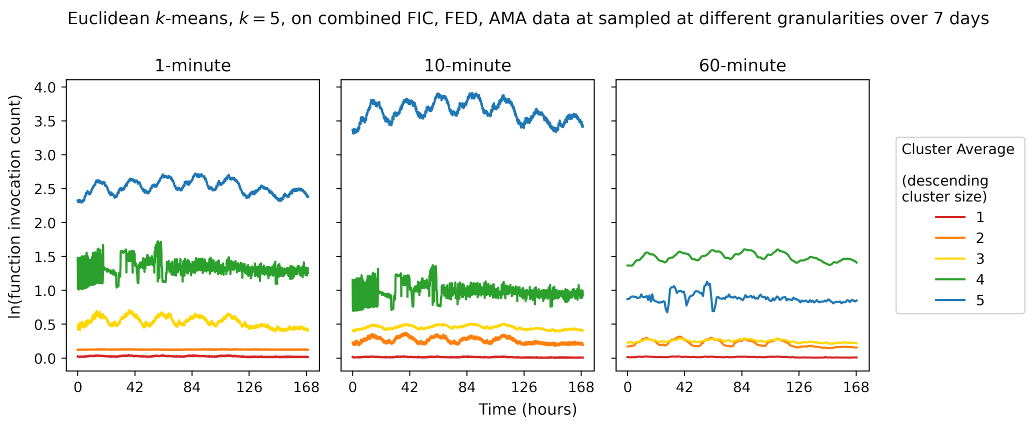


Figure 2.1: The average shape of data in each cluster generated by Euclidean k-means, $k = 5$, on combined FIC, FED and AMA data sampled at different granularities over 7 days

To reduce the log processing cost, we experiment with sampling the transaction log at different rates, effectively smoothing the data. Figure 2.1 illustrates the results from table 2.1 when $k = 5$ and plots the average function invocation count in each cluster over time, sampled at different granularities. As previously stated, the dominant cluster (cluster 1) consistently encapsulates infrequent usage behaviour. The results indicate that as the cluster size decreases, the average function invocation count increases. When the data is sampled every 60 minutes, however, the trend is less evident. Here there does not appear to be a clear definition between the behaviours captured by clusters 2 and 3. It is also the case that the smallest two clusters, 4 and 5, have switched positions.

As expected, we discover that key behavioural information is retained until the sampling rate is too low, at which point it becomes difficult to distinguish between different behaviours. Given how rapidly cloud usage logs can grow, we can see in table 2.1 that even a small degree of data smoothing can drastically cut processing time, without sacrificing performance.

Exploring Single Customer Behaviour with Euclidean k-means

As described earlier, the Azure dataset contains information about customer usage of a cloud service. Each customer can be associated with one or more applications and each application is associated with one or more function calls.

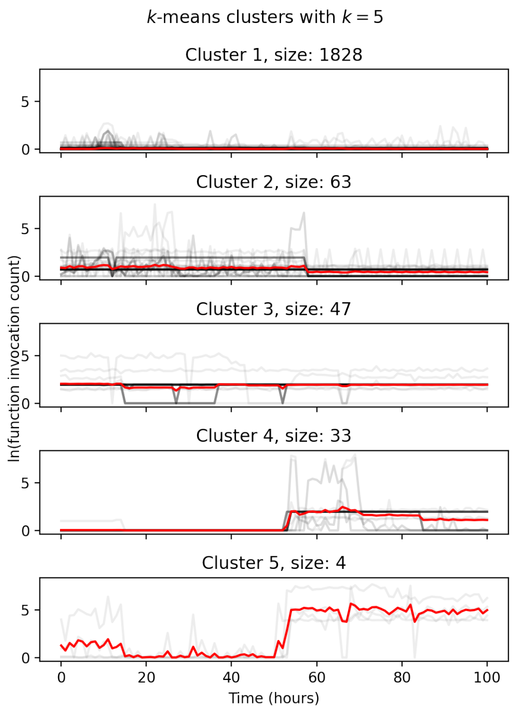


Figure 2.2: Contents of clusters generated by Euclidean k-means for a single customer over 7 days, plotting each time series (grey) in and the average FIC (red) of the cluster

We explore the performance of Euclidean k-means when analysing the behavioural patterns of a single customer. We take a subset of the previous log, sampled every 10 minutes, containing all 1975 entries from the most popular customer. Figure 2.2 illustrates the performance of Euclidean k-means with $k = 5$ on this dataset. For each cluster, we have drawn each time series in grey. In red we plot the average function invocation count in each cluster, highlighting the overall shape of each cluster. Again we see the largest cluster, cluster 1, with a low average, indicating instances of low use. cluster

5, the smallest, encapsulates some abnormal long term intense usage of the service that may warrant further investigation.

A service provider may wish to observe the behaviour of a single customer over a period of time in order to build a profile of their typical behaviours. We have shown that clustering algorithms can be employed effectively in this situation. Service providers however may wish to compare the behavioural patterns observed between various customers. Given the nature and accessibility of cloud services, customers are located all over the world. As a result, developing a clustering method to group together similar behaviours, independent of time, would be ideal.

Euclidean k-means vs Dynamic Time Warped k-means

The last experiment explores the potential of a clustering algorithm specific for time series data, Dynamic Time Warped (DTW) k-means. We execute this experiment exclusively on FIC data from day 1, sampled every 10 minutes, in anticipation of the $O(n^2)$ time complexity of DTW k-means. The log comprises $46,412$ unique entries and the performance of the clustering algorithms can be seen in table 2.2.

Algorithm	Cluster Sizes	Time (s)
Euclidean k-means	[39849, 4202, 1428, 675, 258]	3.5
DTW k-means	[38164, 5438, 1820, 810, 180]	7151

Table 2.2: Performance of k-means, $k = 5$, with different distance metrics

Figure 2.3 illustrates the overall shape and information captured in each cluster by each algorithm. The DTW k-means method appears to identify certain specific patterns in the data, whereas the Euclidean k-means approach appears to group data by a specific mean value in each cluster. This suggests that DTW k-means groups the data based on more specific information.

Although the size of clusters generated by Euclidean k-means and DTW k-means is similar, we examine the overlap in cluster assignments. Figure

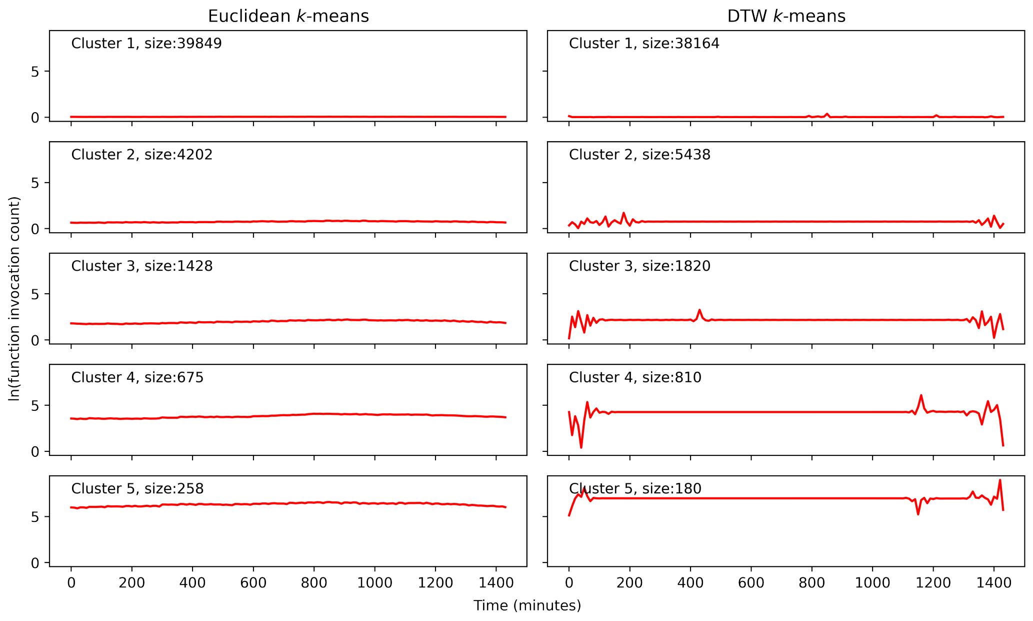


Figure 2.3: Contents of clusters generated by k-means on FIC data from day 1, plotting each time series (grey) in and the average FIC (red) of the cluster

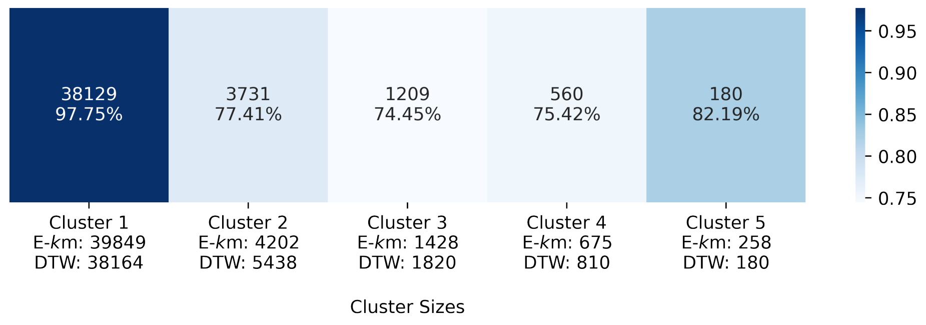


Figure 2.4: Agreement matrix, values in each cell state the number and proportion of overlapping cluster assignments between the Euclidean (E-km) and DTW k-means

2.4 illustrates an agreement matrix where the value in each cell states the number and proportion of overlapping instances in each cluster. As mentioned earlier, the largest cluster typically encapsulates the low service use behaviour, while the smallest clusters typically encapsulate high use, and the middle clusters specific behaviours. We see the majority of overlapping occurs in the largest and smallest clusters. The fact that the medium-size clusters have less overlap shows that each algorithm has produced somewhat distinct behaviour descriptions. This confirms the previous results from figure 2.3 that suggest that DTW k-means groups the data based on different, potentially more specific, information.

Although the clustering performance of Euclidean k-means and DTW k-means is similar, the time complexity of DTW k-means is substantially, almost $2,000$ times, larger. For this reason alone, it would not be feasible to deploy DTW k-means at scale.

2.5 Summary

The main takeaways from this chapter:

- Given how recent AIaaS is, there is a lack of public usage data. We instead generated realistic AIaaS traces from real-world data.
- It is possible to use clustering algorithms to identify different patterns of behaviour. We have shown the potential for a generic, cluster analysis based approach to misuse monitoring.
- Although there is currently no efficient 'off-the-shelf' time series clustering method available, it would be more appropriate in the future to group similar patterns of behaviour, independent of time.

Chapter 3

Surveillance Detection

In this chapter we explore the feasibility of methods to detect specific types of AI service misuse and choose to do so in the context of facial recognition software. Facial recognition is a popular computer vision task that involves recognising and verifying an individual from a photograph of their face. Many of the major AIaaS providers offer products to seamlessly detect, identify and even analyse faces. Facial recognition software however, raises major concerns about privacy and the enabling of surveillance.

In June 2020, after many concerns that the technology was being used to promote racism, IBM announced that it would no longer develop or sell facial recognition software [15]. The concerns were legitimate; the majority of facial recognition algorithms performed poorly on non-white faces, according to a report from the US National Institute of Standards and Technology [16]. Notably, not long after IBM, Amazon announced that they would impose a one-year ban on police use of Rekognition [17], their cloud-based computer vision platform. Amazon is the largest provider of facial recognition technology to U.S. law enforcement, including federal immigration agencies and the FBI [18]. As of May 2021, the moratorium will remain in place, until further notice [19]. Following Amazon, Microsoft revealed the following day that they too would stop supplying this software to police forces until the technology is regulated by federal legislation [20]. The misuse of facial recognition software is easy to imagine, and the high-profile reaction to law enforcement use of face services legitimizes concerns.

In recent years state departments internationally have begun using facial recognition technologies on videos captured during political protests ([21], [22]). The software allows law enforcement agencies to automatically identify, monitor and track protesters at a low cost. The Office of the UN High Commissioner for Human Rights published a report in 2020 examining the impacts of these new technologies on human rights, in the contexts of peaceful protests [23]. They claim that the increasing use of facial recognition software has had a substantial effect on citizens' right to freely assemble and discourages people from openly expressing their views in public. Furthermore, facial recognition technology has the potential to reinforce and amplify discrimination against minorities. In January 2021, Amnesty International called for a ban on 'dangerous facial recognition technology that amplifies racist policing' [24].

In this chapter we explore two specific scenarios of potential misuse in the facial recognition domain. The first considers the possibility of large-scale monitoring, or surveillance, while the second considers the use of AI face services to repeatedly profile, or stalk, an individual.

3.1 Approach

The purpose of the misuse indicators is to attract the attention of service providers to specific customer behaviours that may indicate misuse and warrant further attention. We demonstrate the potential of two trait-based indicators of potential misuse and highlight the practical overhead considerations of the different detection approaches available.

The misuse indicators examine the usage logs of a facial recognition service. Here the usage logs describe the information gathered as a byproduct of using an AIaaS face service. As such, each entry in the log file is a vector encoded representation of the face in each image. As the faces are encoded as vectors, we can easily interpret them as points in the Cartesian coordinate system. For two images of the same person, the distance between the encoded vectors will be small; for different individuals, the distance will be larger [25].

A typical usage log would contain a large number of unlabelled faces. We choose to implement a cluster based analysis as clustering algorithms are useful for blindly exploring and grouping a large collection of unlabelled data. We group the images in the log into the individual identities present, with one cluster for each person. Although the use of clustering has been extensively explored in pattern recognition, machine learning and statistical scenarios [26], it is relatively recent in the field of face recognition.

Otto et al. provide a brief review of clustering face images by identity [27]. According to the authors, since there is no universal face representation or distance metric, the performance of a clustering algorithm in this scenario is dependent on the clustering algorithm chosen as well as the quality of the underlying face representation. The face representation we have chosen is described in section 3.3.2 while the clustering algorithms are discussed below.

3.1.1 Defining the Misuse Indicators

We first implement a trait-based misuse indicator that highlights potential wide-scale surveillance. The indicator examines the usage logs to determine the number of people a customer is processing. A customer using a facial recognition service to process the images of a large number of individuals could indicate large-scale biometric processing and inappropriate population monitoring and should be flagged for further review.

The concerns surrounding readily-available facial recognition technologies focus mainly on wide-scale surveillance. A secondary concern lies in the potential for surveillance on a much smaller scale, in particular the targeting of an individual. We also evaluate the feasibility of a misuse indicator that can detect certain usage patterns that may be indicative of stalking. Here the misuse indicator flags instances where a customer is processing an unusually

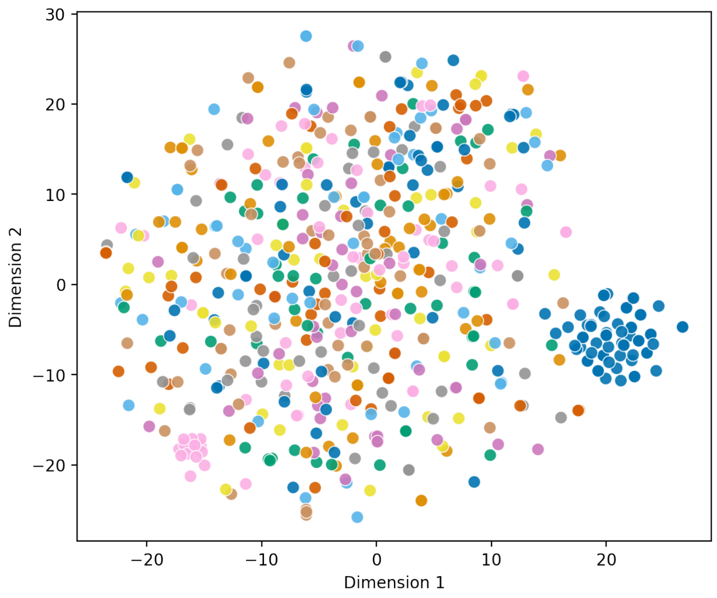


Figure 3.1: T-SNE visualisation of 600 face encodings, 60 from same person

large number of images of a single person. Figure 3.1 illustrates a T-SNE plot of a usage log made up of 600 face encodings. Of the 600 encodings 10%, 60, images are of the same person. One cluster is significantly larger than the rest, representing the individual disproportionately represented in the dataset.

3.2 Clustering Approaches

The usage logs we examine describe the information gathered as a byproduct of using an AI service for image processing and contain a large number of unlabelled images. The misuse detector employs a cluster-based analysis to determine the number of individual identities present in the data. For the majority of clustering algorithms, the number of clusters, k, must be specified as an input parameter.

In this chapter we explore two alternative clustering methods, namely DB-SCAN and Chinese Whispers that independently compute k and use it to estimate the number of unique individuals in the images.

3.2.1 DBSCAN

The Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm is one of the most widely used density-based clustering algorithms and has proved useful for imaging tasks [28]. The DBSCAN algorithm requires the definition of two parameters:

- $minPoints$: minimum number of points required to form a dense region
- ϵ: maximum distance between two points such that they be considered part of a cluster

DBSCAN will independently determine the number of clusters. DBSCAN can successfully discover clusters of different shapes and sizes from large noisy datasets with outliers. It works by determining whether the minimum number of points are close enough to one another to be considered part of a single cluster. Since ϵ is a fixed value for the maximum distance between two points in a dense region, the algorithm is very sensitive to scale.

A Euclidean distance metric is used to determine whether two images are likely of the same person. Here ϵ describes the maximum distance between two face encodings that can be considered the same person. The method we use to select the value of ϵ can be found in section 3.3.4. Given that it is common for people to have only one image in a log, we set $minPoints = 1$.

3.2.2 Chinese Whispers

Chinese whispers (CW) is a hard partitioning, randomised, flat graph-clustering algorithm that works on undirected graphs [29]. A graph represents objects and their relations. Instead of using a distance metric, the similarity between objects is encoded in the edges of the graph. It is an efficient algorithm with a time complexity that is linear in the number of edges. As such, the algorithm is suitable for very large graphs.

The CW algorithm begins with every node in the graph assigned to a distinct class with one node in each class. Nodes are then repeatedly chosen randomly, and assigned the 'strongest' class in their neighbourhood. The strongest class is the class whose total edge weights to the current node is the largest. In the case of a tie, one class is randomly selected. Once the process converges, or the predetermined number of iterations is reached, the algorithm is complete and the emerging classes represent the network clusters.

Although the CW algorithm is most frequently used in the field of NLP [30], it is beginning to be used in computer vision tasks. Recently, Klip used the algorithm to group images of faces by identity for a forensic investigation task [31]. In section 3.3.2 we describe how we transform a log of face images into an undirected, weighted graph.

3.3 Experimental Setup

As mentioned, we conduct two main experiments to explore the potential for clustering faces. The experiments are all conducted on external machines using the High Performance Computing Service (HPCS). Each experiment was run five times, the results averaged.

3.3.1 Face Image Datasets

To explore the potential of the misuse indicators described, we outline experiments leveraging three public datasets:

- A subset of the CelebA dataset [32]: comprising 40995 face images from 2000 individuals.
- ColorFeret by NIST [33]: comprising 11338 face images from 994 individuals.
- Labelled Faces in the Wild (LFW) [34]: comprising 13233 face images from 5749 individuals.

There is only one face in each image.

CelebA is used to tune the parameters of the clustering algorithms (see section 3.3.4). The ColorFeret and LFW datasets are then used to generate the usage logs and evaluate the performance of the clustering algorithms.

3.3.2 Image Preprocessing

The CelebA, ColorFeret and LFW datasets are a collection of images, with each image containing one face. Preprocessing of the images involves (1) face detection and (2) feature extraction. We use an implementation of the Multi-Task Cascaded Convolutional Neural Network (MTCNN) for face detection and a TensorFlow implementation of FaceNet to generate an embedding for each detected face [35].

Face Detection

Face detection is the process of automatically identifying and localising one or more human faces in an image. A bounding box is drawn around the extent of each face. We use an implementation of MTCNN, a state-of-the-art deep learning model for face detection based on the 2016 paper from Zhang et al. [36].

Feature Extraction

FaceNet uses a deep convolutional neural network to extract high quality features from an image of a face [25]. The facial recognition system takes an image as input and returns a vector of size 128. FaceNet extracts the most important features and essentially compresses the image into an embedding. Here we use a pre-trained Keras FaceNet model [37] trained on the large MS-Celeb-1M dataset [38], comprised of over 10 million face images of approximately $100,000$ individuals. The model expects colour images of size 160×160 with standardised pixel values.

3.3.3 Graph Construction

For the graph-based CW algorithm, the log of image encodings $[e_1, e_2, ..., e_n]$ is transformed into an undirected weighted graph G. Pseudocode for the graph construction is detailed in algorithm 1.

The algorithm iterates over each encoding, e_i, in the list. A node is first initialised for e_i (line 4). The algorithm then iterates over the remainder of

Algorithm 1 Graph Construction

Input: A list of image encodings, $encodings = [e_1, e_2, ..., e_n]$ and a threshold value, $threshold$

Output:A weighted undirected graph, G

```
 1: $nodes \leftarrow \emptyset$
 2: $edges \leftarrow \emptyset$
 3: for $e_i \in encodings$ do
 4:     $nodes \leftarrow nodes + Node(e_i)$
 5:     for $e_j \in encodings \setminus e_i$ do
 6:         $distance \leftarrow Distance(e_i, e_j)$
 7:         if $distance > threshold$ then
 8:             $edges \leftarrow edges + Edge(Node(e_i), Node(e_j), distance)$
 9:         end if
10:     end for
11: end for
12: $G = Graph(Nodes, Edges)$
13: returns: $G$
```

the encodings in the list after e_i (line 5). For each encoding e_j, we compute the Euclidean distance between e_i and e_j (line 6). If the distance between the encodings is above a certain threshold, an edge is created between the nodes for these two encodings. The edge is weighted by the distance between the encodings (line 8). Given the final list of nodes and edges, the algorithm builds and returns the graph.

Although CW is itself a parameter-free, unsupervised clustering algorithm, a threshold parameter is required to construct the graph in algorithm 1. The methodology used to determine the value of this parameter is explained below.

3.3.4 Hyperparameter Tuning

As mentioned in section 3.3.1, the CelebA dataset is used to determine the optimal hyperparameters for each of the clustering algorithms.

Firstly, we analyse the performance of DBSCAN on the entire CelebA dataset with a variety of values for ϵ to discover the ideal value. The value of ϵ that generates the smallest estimation error is chosen. The estimation error is defined as,

$$\frac{|n_{clusters} - n_{people}|}{n_{people}} * 100$$

where $n_{clusters}$ is the number of clusters generated by the algorithm and n_{people} is the number of unique individuals present in the log. Based on the results (figure 3.2) we select $\epsilon = 9.8$.

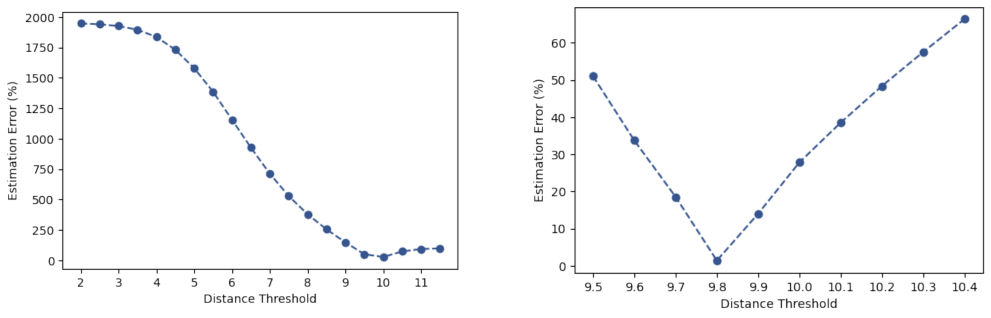


Figure 3.2: DBSCAN estimation error for different ϵ, optimal $\epsilon = 9.8$

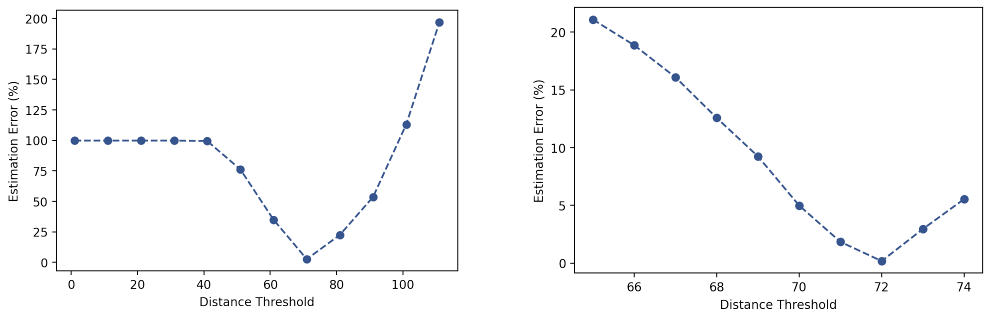


Figure 3.3: CW error for different *threshold*, optimal *threshold* = 72

Although CW is itself a parameter-free, unsupervised clustering algorithm, it requires a weighted graph as input. In order to generate this graph a threshold value for distance is required. Using the CelebA dataset, we explore a range of values for the distance threshold and again select the value that produces the lowest estimation error. Based on the results (figure 3.3), we select a *threshold* value of 72. Overall these findings show that the hyperparameter settings used in both DBSCAN and CW are extremely sensitive. This is discussed further below.

3.4 Results

The clustering algorithms are evaluated on usage logs of face embeddings. The face encodings are generated from images sampled from both the LFW and ColorFeret datasets that contain 24517 images of 6743 people in total.

3.4.1 Wide-scale Surveillance

To independently explore the effects of both the log size and the number of individuals in each log, two types of logs are generated, **fixed logs** and **varied logs**. Fixed logs are fixed in their length, but the number of different people represented can vary, whereas we fix the number of number of individuals and vary the log length in varied logs. To generate the required number of entries and individuals in each log, some encodings in the log are repeated.

For each generated log of face encodings, we record the number of clusters and use this to predict the number of unique people represented in the log. We execute the DBSCAN ($\epsilon = 9.8$) and CW ($threshold = 72$) clustering algorithms over the logs, measuring the estimation error and clustering time in each instance. Table 3.1 shows the performance and overheads of the DBSCAN and CW clustering algorithms on fixed and varied logs.

Firstly, we explore the performance of the clustering algorithms when the log size is fixed at 1000 and $25,000$ and the number of individuals present in the log, n_{people} is varied. Here the DBSCAN algorithm surpasses the CW algorithm in terms of accuracy, consistently predicting the number of people represented in the data with a substantially smaller estimation error. DBSCAN also surpasses CW in terms of clustering time, with lower and more consistent values. When the log size is fixed to $25,000$ DBSCAN clustering times have a range of 0.31s and CW a range of 12.01s. We can also observe that increasing the number of people in the log has no influence on the time complexity of the clustering methods.

The performance of the clustering methods is next investigated when the log

		DBSCAN			Chinese Whispers		
Log Size	n_{people}	$n_{clusters}$	Error (%)	Time (s)	$n_{clusters}$	Error (%)	Time (s)
1000	50	52.4	4.80	0.03	55.4	10.80	0.17
1000	100	104.0	4.00	0.02	113.0	13.00	0.12
1000	250	250.0	0	0.03	262.6	5.04	0.13
1000	400	381.2	4.70	0.03	371.0	7.25	0.18
1000	500	466.8	6.64	0.02	446.6	10.68	0.25
25000	500	466.2	6.76	6.38	682.0	36.40	13.81
25000	1000	907.6	9.24	6.21	1126.8	12.68	9.95
25000	2500	2057.6	17.70	6.23	2113.0	15.48	10.63
25000	4000	3134.2	21.65	6.30	2687.6	32.81	14.47
25000	5000	3773.4	24.53	6.52	2693.4	46.13	21.96
500	500	474.6	5.08	0.01	444.6	11.08	0.01
1000	500	466.8	6.64	0.02	446.6	10.68	0.25
2500	500	465.4	6.92	0.16	504.8	0.96	0.57
4000	500	475.2	4.96	0.33	534.2	6.84	2.35
5000	500	470.4	5.92	0.55	558.8	11.76	1.98
5000	5000	4385.7	12.29	0.33	2286.4	54.27	0.40
10000	5000	4104.0	17.92	0.97	1864.6	62.71	7.83
25000	5000	3773.4	24.53	6.52	2693.4	46.13	21.96
40000	5000	3752.0	24.96	15.74	3266.0	34.68	33.01
50000	5000	3774.2	24.52	24.42	3388.0	32.24	35.89

Table 3.1: Clustering algorithm performance when log size and n_{people} varied

size is varied between 500 and $50,000$ and the number of persons represented is kept constant at 500 and 5000. As expected, the clustering time is proportional to the log size. Once again, the estimation errors and clustering times from DBSCAN consistently outperform CW. When the log size and number of persons in the log are similar, i.e. the log likely comprises a relatively small number of photos of each person, the CW estimate error appears to be the worst.

Overall, for the DBSCAN algorithm in particular, the number of clusters returned roughly corresponds to the number of distinct faces in the dataset. From this we can see that it is feasible to use clustering algorithms to uncover the different identities present in a group of faces.

Overheads

In the worst case, DBSCAN can have a time complexity of $O(n^2)$, where n is the number of data points. The CW algorithm on the other hand, boasts a linear time complexity of $O(n)$. In table 3.1, when the log size increases from $40,000$ to $50,000$, the clustering time increases by $8.7s$ for DBSCAN and only $2.9s$ CW.

However, none of the image preprocessing is taken into account in the clustering times reported. The preprocessing, which involves transforming raw images into face encodings for both DBSCAN and CW, is described in section 3.3.2. Given the nature of popular face analysis models and services today, we may expect that these encodings will already be computed by service providers. As a result, this preprocessing step would not add any further overhead.

However, the CW algorithm requires an additional step, converting the face encodings into a weighted graph. The graph construction algorithm implemented in section 3.3.3 has a considerable overhead, given the $O(n^2)$ time complexity. Other, more efficient, graph construction algorithms may be available. It is important to note that the choice of graph construction algorithm will have an impact on the clustering performance of the CW algorithm, so further experimentation would be needed.

3.4.2 Small-scale Surveillance

To be indicative of potential stalking, each log now contains a disproportionate number, $\approx 10\%$, of images from a single individual.

	Cluster Sizes (descending)		
Log Length	True	DBSCAN	Chinese Whispers
100	10, 2, 2, 2, ...	10, 3, 2, 2, ...	10, 2, 2, 2, ...
1000	102, 15, 5, 4, ...	205, 15, 5, 4, ...	104, 16, 10, 8, ...
2500	252, 52, 22, 19, ...	667, 65, 22, 18, ...	276, 67, 28, 19, ...
5000	505, 92, 47, 28, ...	1597, 94, 47, 28, ...	522, 136, 58, 57, ...
10000	1053, 217, 86, 34, ...	4563, 46, 36, 31, ...	1194, 294, 108, 87, ...
100	14, 13, 11, 11, ...	14, 13, 11, 11, ...	14, 13, 11, 11, ...
1000	222, 191, 108, 101, ...	222, 183, 180, 108, ...	222, 185, 108, 101, ...
2500	273, 267, 261, 259, ...	514, 272, 264, 251, ...	272, 264, 258, 251, ...
5000	558, 506, 498, 496, ...	536, 498, 496, 495, ...	540, 498, 496, 495, ...
10000	1041, 1018, 1015, 1011, ...	2027, 1992, 1964, 1015, ...	1018, 1015, 1009, 1000, ...

Table 3.2: Clusters generated from different data distributions in the log

The first half of table 3.2 highlights the performance of the DBSCAN and CW clustering algorithms when 10% of the images are from a single person. The CW algorithm can successfully identify the cases of stalking by extracting clusters that match the true distribution of the underlying data. It is important to note the additional preprocessing costs required by the CW algorithm, mentioned earlier.

The DBSCAN algorithm fails in this scenario and cannot identify the true distribution of the data. From a log length of 1000 onwards, the algorithm significantly overestimates the size of the largest cluster each time. It is unclear if the fact that one cluster is much larger than the others affects the performance of DBSCAN, or the magnitude of the largest cluster itself. We subsequently generate new logs, where each person is now represented in $\approx 10\%$ of the images.

The findings show that the DBSCAN algorithm accurately extracts clusters that match the underlying data distribution in the vast majority of situations. This shows that the prior DBSCAN results were skewed because one cluster was significantly larger than the rest.

3.5 Summary

The main takeaways from this chapter:

- We have shown the potential for clustering algorithms in monitoring for specific misuse of a specific service, namely different types of surveillance using facial recognition services.
- The DBSCAN algorithm performed well in the large-scale surveillance scenario, correctly identifying the number of individuals in the log of faces. In the stalking scenario however, CW outperformed DBSCAN and was able to successfully identify the individual disproportionately represented in the log.
- The performance of the misuse-specific detection method is highly dependant on the choice of algorithm and hyperparameters.

Chapter 4

Hate Speech Detection

The automated analysis of natural language, such as speech and text, by machines is known as natural language processing (NLP). Driven by the dramatic increase in smart device use, as well as the transition to customer service chatbots, NLP has quickly become one of the most heavily researched subjects in the field of AI. According to the 2020 NLP industry survey, over three-quarters of NLP users utilise a cloud NLP service [39], with Google Cloud the most popular. Despite the global COVID-19 pandemic, respondents indicated that spending on NLP had still consistently increased, with 31% of participants reporting that their budget was at least 30% higher, in comparison to 2019.

All of the major service providers, including Google, AWS, Azure and IBM, offer pre-built powerful NLP models, trained on huge document corpuses, as-a-service. An issue can arise when service providers are processing potentially harmful content, in particular hate speech. In this section we consider mechanisms to detect the misuse of these NLP services for processing potentially problematic speech.

According to the Cambridge dictionary, hate speech 'expresses hate or encourages violence toward a person or group based on something such as race, religion, sex etc' [40]. The majority of the current discourse surrounding hate speech refers to social media posts, and has attracted substantial investment from governments, businesses, and academics. We are instead monitoring the data passed into cloud language models in order to identify and further investigate any potentially problematic or controversial applications, such as a racist chatbot or a hate speech generator.

On Twitch, a popular live streaming platform, users can donate to streamers with a message that will be read aloud, live, by text-to-speech software. In May of 2019, Twitch suspended a steamer for 24 hours after a viewer made a donation that used the text-to-speech function to repeat a racial slur [41]. The existing measures to prevent this use word filters and are easy to bypass. Building a comprehensive, robust system to detect instances of hate speech in real-time is a difficult task.

Firstly, we must acknowledge the processing overheads associated with various approaches to misuse detection. To avoid situations like the Twitch streamer above, service providers should ideally identify hate speech passed into their models in real time. Given the growing popularity of AIaaS and the potential size of each application's user base, this may not be feasible on a large scale.

There is also a lack of consensus as to what actually constitutes 'hate speech'. Some researchers, including Ross et al. [42], believe that a consistent, clear description of hate speech would make hate speech annotations more reliable and easier to do. As a result, hate speech would be easier to detect. Unfortunately however, since the line between hate speech and acceptable free speech is vague, some are hesitant to assign a clear definition. In this research we make use of a variety of existing hate speech datasets with a range of requirements and definitions to capture the true variance of real-world data.

Although the problem of offensive and harmful speech online is serious, hate speech only accounts for a relatively limited portion of overall discourse. This means that simple data collection methods can result in highly unbalanced datasets. For this task we have chosen datasets where the actual percentage of hate speech present is slightly inflated to allow for effective model training. A full description of the datasets can be found in section 4.2.1. The context-dependent nature of natural language classification also contributes to the difficulty of this task [43]. In the context of AI service misuse, we would rather be cautious and flag all possible cases of misuse for further investigation.

To detect instances of hate speech fed to the models, the service providers would need to directly process the raw user data, specifically for the purpose of misuse monitoring. The usage logs will be made up of raw text documents users. In the next section we discuss the theory supporting the different approaches to document classification we consider.

4.1 Text Classification Approaches

Over the past decade, a vast number of techniques have been developed for automatically detecting hate speech. In tandem with the perceived importance of the task, the field of hate speech detection has been steadily increasing, since its conception in 2012 [44]. In this section we present a lexicon-based approach as well as several machine learning approaches to the task of hate speech detection.

4.1.1 Lexicon-based

A lexicon-based approach to hate speech detection will classify a document by comparing the words and phrases present in the document with a lexicon, a list of predefined terms. A lexicon-based approach to classification is simple, efficient and does not rely on a labelled training dataset of documents. The lexicon itself is easy to modify and decisions made by the model are easy to trace and explain.

For this task, we leverage a large hate speech lexicon assembled by Hatebase.org, the world's largest online repository of terms and phrases identified as hate speech. The lexicon is regularly updated, with users able to log on and add examples of hate speech from their own communities. At the time of writing, Hatebase comprises $3,877$ terms that cover 98 languages. For this project we consider the $1,556$ English terms. Any document that contains a term from the Hatebase lexicon is classified as 'harmful' and flagged for further review.

4.1.2 Machine Learning Approaches

The use of artificial intelligence (AI) technologies to identify harmful material has been the subject of extensive academic and industrial research [45]. In this section we explain three supervised machine learning approaches to hate speech detection, Naive-Bayes, logistic regression and support vector machines.

The raw documents are first preprocessed and converted into a matrix of Term Frequency - Inverse Document Frequency (TF-IDF) features. We first evaluate the performance of the three machine learning models using the TF-IDF features alone. We then evaluate the performance of the machine learning models using the sentiment of the document as an additional feature during classification.

In this section we explore the theory behind the three machine learning algorithms as well as the mechanism used to determine the sentiment of each document.

Naive-Bayes

Naive-Bayes is a probabilistic based classification algorithm, derived from Bayes' rule

$$P(\,c\,|\,d\,) = \frac{P(c)P(\,d\,|\,c\,)}{P(d)}$$

and assumes conditional independence of features f_i given document d's class,

$$P_{NB}(\,c\,|\,d\,) = \frac{P(c)(\prod_{i=1}^{m} P(\,f_i\,|\,c\,)^{n_i(d)})}{P(d)}$$

Naïve-Bayes assigns d, with feature vector $\vec{d}$, the most probable class $\hat{c}$,

$$\hat{c} = \arg\max_{c \in \mathcal{C}} P(\, c \,|\, \vec{d}\,) = \arg\max_{c \in \mathcal{C}} P(c) \prod_{i=1}^{m} P(\, f_i \,|\, c\,)$$

Despite the oversimplified assumptions, Naive-Bayes works well in many real-world applications [46].

Logistic Regression

Logistic regression is a fundamental classification technique. It uses the logit function to model conditional probability and explain the relationship between one dependent variable and one or more independent variables. The logit function is the natural logarithm of the likelihood of a particular outcome,

$$logit(P(x)) = ln\left(\frac{p(x)}{1-p(x)}\right)$$

where $p(x)$ is the probability of an event x.

In Logistic regression the logit of the probability is said to be linear with respect to x. As a result, the logit function is mapped to a linear combination of inputs, and logistic regression can be expressed as,

$$logit(P(x)) = \beta_0 + \beta_1 x$$

Equating and rearranging the two equations reveals that,

$$ln\left(\frac{p(x)}{1-p(x)}\right) = \beta_0 + \beta_1 x$$

$$\Rightarrow \frac{P(x)}{1-P(x)} = e^{\beta_0+\beta_1 x}$$

$$\Rightarrow P(x) = \frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}}$$

This final equation models the relationship between the variables as a Sigmoid curve. The regression gradient coefficients, β_i, are then calculated using maximum likelihood estimation (MLE) [47].

In this task each document d is a vector of n features. As such, the logit equation can then be expanded to handle multiple gradients,

$$logit(P(\mathbf{x})) = (w_0x_0 + w_1x_1 + ... + w_nx_n) + b = \mathbf{w}^T\mathbf{x} + b$$

Support Vector Machines

Support Vector Machines (SVM) are a generalisation of simple maximal margin classifiers, formally defined by a separating hyperplane. As mentioned, each d is a vector n features.

Given labelled training data in the form of n-dimensional vectors, the algorithm outputs an optimal $(n-1)$-dimensional hyperplane, maximising the margin between vectors in each class. The predicted class of a test instance is determined by which side of the hyperplane it falls on.

Sentiment Analysis

Sentiment analysis is the process of 'computationally' deciding whether a piece of writing is positive, negative, or neutral. It is assumed that the majority of hate speech exhibits a higher degree of negative polarity, in comparison to non hate speech [48]. We investigate the impact of using the sentiment of a document as an additional feature during classification, to determine whether this additional computational overhead is worthwhile.

We compute the sentiment of each document using an implementation of a sentiment analyser. VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is uniquely tailored to sentiments expressed in social media [49]. VADER is included in the Natural Language Toolkit (nltk) package and can be applied directly to raw unlabelled text data.

4.2 Experimental Setup

We conduct experiments to explore the feasibility and potential performance overheads of the different hate speech detection algorithms presented. The experiments are conducted on a personal computer, specifically with the 8th gen Intel Core i7, 16GB Ram, 500GB SSD, running Windows.

4.2.1 Hate Speech Datasets

In their 2020 paper, Derczynski et al. systematically review over 60 publicly available English-language datasets created to train abusive language classifiers [50]. For this task we consider four of these labelled datasets, Davidson et al. [51], Founta et al. [52], Waseem & Hovy [53], and Stormfront [53].

The Davidson et al. dataset features tweets labelled by a minimum of three annotators. As per previous research with this dataset, we consider only tweets for which a majority of annotators (at least half) agree on a label. The Hatebase lexicon was used to extract a set of twitter users that have tweets containing these specific terms. Tweets are randomly sampled from the timelines of these users. The authors use crowdsourcing to annotate the tweets.

Waseem and Hovy also provide a dataset from Twitter. This time the authors themselves annotated the tweets. Once labelled, the authors had a gender studies expert review the annotations. The original Founta et al. dataset contains around 100K randomly-collected tweets with one of four labels. Unfortunately, due to deleted tweets or suspended user accounts, we were only able to collect $53,633$ tweets out of the $99,799$ tweets IDs. The tweets were labelled by 20 crowdsourced amateurs.

Unlike the previous datasets that contain data from Twitter, the Stormfront dataset contains posts from a white supremacist forum. Here the posts are annotated at sentence level. The dataset is ordered and the ‘relation’ sentences indicate sentences that do not contain hate speech on their own, but the combination of several surrounding sentences does. Sentences that are not written in English, or that do not contain information that can be classified into ‘hate’, ‘neutral’ or ‘relation’ are labelled ‘skip’. The authors of the Stormfront dataset annotated the data themselves.

Dataset	Source	Label	Count
Davidson et al.	Twitter	hateful*	1429
		offensive*	19188
		neutral	4163
Waseem & Hovy	Twitter	sexism*	4217
		racism*	25
		none	12663
Founta et al.	Twitter	hateful*	2018
		abusive*	5010
		spam	8474
		normal	38131
Stormfront	Online forum	hate*	1196
		relation	168
		skip	73
		neutral	9507

Table 4.1: Summary of the hate speech datasets (*denotes harmful labels)

Table 4.1 contains a summary of the datasets. We use Davidson et al. to train the hate speech classifiers. The remaining datasets are merged and used to generate representative usage logs and evaluate the performance of the different approaches.

4.2.2 Preprocessing

The usage logs we examine for this task describe the data that would be passed to an NLP cloud service and contain raw text. Prior to passing the data as input to the classification models, several preprocessing steps are required to clean the data. Preprocessing is performed on all the datasets.

Preprocessing involves tokenizing, stemming, case folding and stop-word removal. Tokenization converts each document into a list of words, or tokens. The Porter stemmer [54] then converts each of the words into their root forms. We remove stopwords using the Natural Language Toolkit (nltk) library that contains 2,400 stopwords for 11 languages. All of the documents are converted into lowercase. Lastly we remove all punctuation, additional whitespace, links and usernames.

We then extract features from the words and n-grams, contiguous sequences of n items, present in the preprocessed documents. Term Frequency - Inverse Document Frequency (TF-IDF) is a feature extraction technique [55]. The tf-idf vectorizer converts a collection of raw documents to a matrix of TF-IDF features. The TF-IDF vectorizer normalizes the count according to the frequency of the n-gram in question in the whole corpus. Some words will be relatively common in a large text corpus (for example, 'the' in English), conveying relatively little meaningful information about the actual contents of the document. Rarer terms, on the other hand, convey more information.

The aim of this misuse indicator is to flag any instances of harmful speech. As a result, we choose to model hate speech detection as a binary classification task such that a document either contains harmful speech that requires attention, or does not. We encode the labels in each of the datasets to reflect this. The asterisk symbols in table 4.1 denote the classes in each dataset we consider 'harmful'.

4.2.3 Evaluation Metrics

Evaluation metrics are defined to assess the performance of the hate speech detection algorithms. As mentioned earlier, the dataset labels have been binarised into positive (harmful language) and negative (non-harmful language) classes. Positive documents are flagged to service providers for further review.

Each log is made up of n documents, $d_1, d_2, ..., d_n$. Each document d_i has a true class label, y_i. The class predicted by an algorithm for d_i is denoted $\hat{d_i}$. The evaluation metrics compare the true class labels of each document with the predicted labels and are defined in terms of the false negatives (FN), false positives (FP), true positives (TP) and true negatives (TN).

In a 2019 research article, MacAvaney et al. discuss the challenges and approaches to automating hate speech detection [45]. The authors were unable to find a consensus in the literature as to which evaluation metrics are most appropriate for this task. However, they agree that concentrating on both accuracy and macro-averaged F_1 scores can provide useful information about the relative strengths and weaknesses of each approach to classification.

To reflect the distribution of real data, the class labels in the usage logs are extremely imbalanced. As such we compute the macro-averaged F_1 score that treats each class equally. For a cloud service provider however, the implications of a false negative are much more serious than a false positive. As such, in addition to the accuracy and macro-averaged F_1 scores, we also report on the positive class recall.

Accuracy

Accuracy is the ratio of correctly labelled documents to all documents,

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Although accuracy is a popular and intuitive evaluation metric, it can be incredibly misleading for imbalanced datasets [56].

Recall

In this context, positive class recall measures the proportion of all the documents containing harmful language that are detected,

$$recall = \frac{TP}{TP + FN}$$

It acts as a measure of the sensitivity of the hate speech detection algorithms.

F_1

The F_1 score is the harmonic mean of the precision [57] and recall scores,

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall}$$

With a macro-averaged F_1 score, all the classes are treated equally, regardless of how often they appear in the dataset. The F_1 scores are first computed for each class, then averaged.

4.3 Results

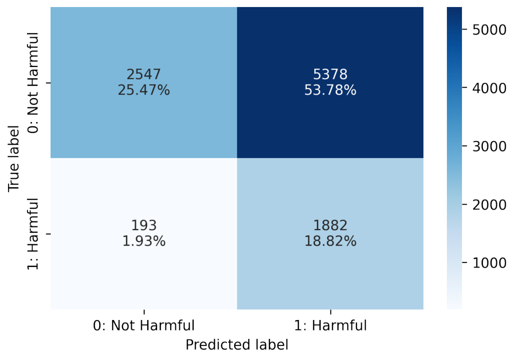
In this section we evaluate the performance of (1) A lexicon-based approach, (2) Machine learning approaches using TF-IDF features alone and (3) Machine learning approaches TF-IDF features as well as sentiment scores. We also investigate the performance overheads associated with various processing techniques including real-time, multi-batch and one-batch processing.

Table 4.2 provides an overview of the performance of the various hate speech detection approaches applied to a log of $10,000$ documents. Although the lexicon-based approach has the highest accuracy (0.79), the 0.08 recall score indicates that it is in fact the weakest hate speech identification strategy, labelling almost every document as ‘not harmful’. The lexicon-based technique is able to handle apparent profanity with ease, but it is restricted to this and fails to recognise the nuance of expression in hate speech.

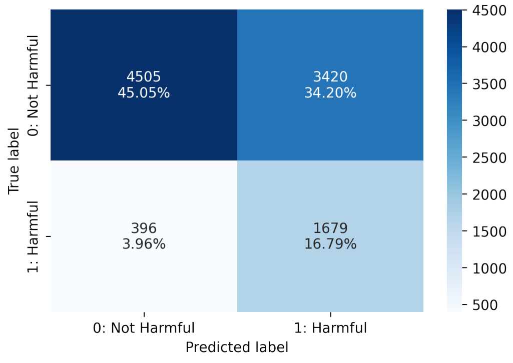
Hate Speech Detection Approach	Accuracy	Positive Class Recall	Macro-averaged F_1
Lexicon-based	0.79	0.08	0.44
Naive-Bayes	0.62	0.69	0.57
Logistic Regression	0.44	0.91	0.44
Support Vector Machine	0.72	0.61	0.64
Naive-Bayes+Sentiment	0.62	0.69	0.57
Logistic Regression+Sentiment	0.62	0.81	0.59
Support Vector Machine+Sentiment	0.73	0.62	0.66

Table 4.2: Performance algorithms on a log of $10,000$ documents

The machine learning approaches are much more sensitive to hate speech, with positive class recall scores of ≈ 0.74. Despite being the most sensitive to hate speech (recall = 0.91), the Logistic Regression classifiers have the lowest accuracy. We see in the confusion matrices (figure 4.1) that these classifiers incorrectly label a large number of documents as harmful.



(a) Logistic Regression



(b) Logistic Regression+Sentiment

Figure 4.1: Confusion Matrix for Logistic Regression approaches on a log of $10,000$ documents

In terms of positive class recall, both the Naive-Bayes and SVM classifiers perform similarly, with SVM achieving a 10% greater accuracy overall. We also investigate the implications of including the sentiment of each document as an additional feature during classification. Although the performance of the Naive-Bayes classifier remains unchanged, the SVM classifier displays a significant ($p < 0.05$) improvement. Given the impressive performance of SVM+Sentiment, we choose to evaluate this approach further.

In the previous experiments we have explored the performance of the hate speech detection methods applied to the entirety of the usage log. We now explore the feasibility and performance overheads associated with various log processing options on a log of $70,000$ documents.

Service providers can choose how to process usage logs. In addition to processing the log as a whole, in one batch, we explore the overheads of batch processing with a batch size of 1000, and real-time processing, classifying one document at a time. Figure 4.2 illustrates the processing times for each method. The processing times are split into preprocessing, feature extraction and hate speech classification using SVM+Sentiment.

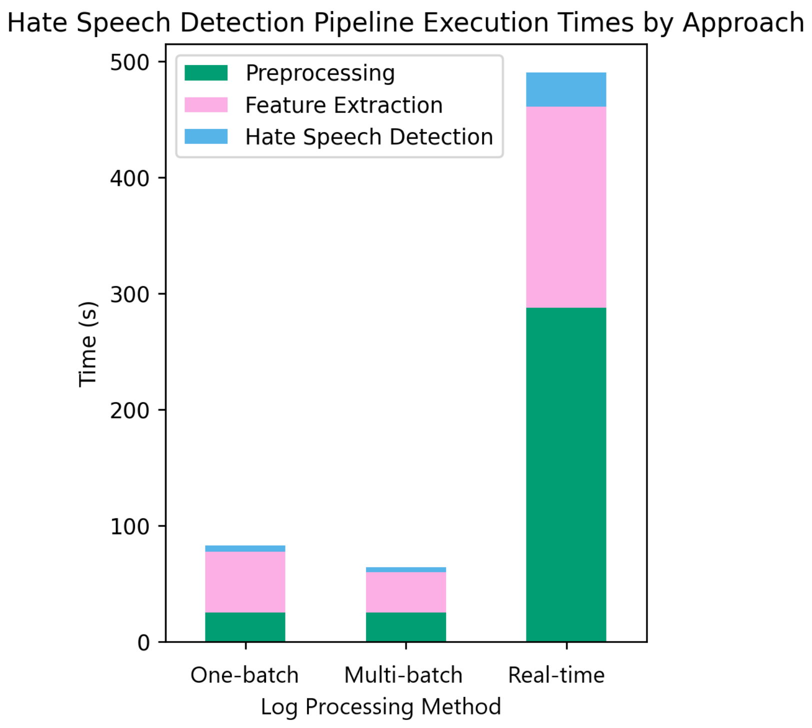


Figure 4.2: Performance overhead on a log of $70,000$ documents classified with SVM+Sentiment

It is clear that real-time processing is infeasible on a large scale, with individual document preparation accounting for the majority of the cost. The A multi-batch processing strategy makes sense for service providers. In addition to being the most time efficient processing method, with a set batch size, the time complexity of misuse detection would be consistent. The batch size can also be adjusted by service providers as needed.

4.4 Summary

The main takeaways from this chapter:

- We have explored the potential for service and misuse specific monitoring that detects instances of harmful speech in the context of cloud NLP services.
- We find that machine learning approaches to hate speech detection are the most sensitive and robust, with SVMs the most successful.
- In terms of processing, we find that a multi-batch processing approach is the most time efficient.

Chapter 5

Discussion

This thesis presents a feasibility study for monitoring customer behaviour while using AI services. Depending on the service provider, customers, service type and relevant legislation, among other factors, the type of monitoring and extent to which customer data is processed, will vary. This chapter begins with an overview of our findings before moving into a more in-depth discussion about the broader implications of this work and ideas for future research.

5.1 Summary of Results

In this thesis we have demonstrated the feasibility and potential for misuse indicators, and highlighted their practical implications. We have established that it is possible to deploy these misuse indicators in a reasonable manner.

With the right clustering algorithm, misuse detection approaches based on cluster analysis can be extremely useful

We demonstrate the utility of cluster based analysis approaches for misuse detection, first in generic, service-independent, usage landscaping, and subsequently in monitoring for specific patterns of behaviour for a particular service, namely face analysis. The purpose of usage landscaping is to identify patterns and group similar behaviours. Our results suggest that clustering methods that require the number of clusters, k, to be specified, such as k-means, successfully allow for flexible, generic, behavioural analysis. Tuning k essentially enables zooming in and out, that is, categorising behaviours into more specific or broad groupings, as required by the service provider.

We also employ clustering approaches that independently compute k, when investigating the potential of algorithms to detect instances of surveillance when using a facial recognition service. The number of clusters is used to estimate the number of individuals represented in the log of faces. We discover that while the DBSCAN algorithm is effective at identifying large-scale population surveillance, it fails to detect recurrent profiling, or stalking, of an individual. The Chinese Whispers algorithm, on the other hand, fails at population monitoring but succeeds in the stalking scenario. Even when monitoring a specific service for a general behaviour, such as surveillance, no one clustering method was successful in every misuse-specific scenario. Future research might focus on designing a novel clustering algorithm that is potentially hybrid in nature.

Time series clustering is expensive

In reality transaction logs are time series data. We discover that existing time series clustering algorithms have a temporal complexity that is too large to be practical, and offer a performance similar to their regular clustering counterparts. Existing approaches are designed for comparing a small set of time series. In order to effectively process and analyse behavioural logs and identify all instances of misuse, future work could develop a clustering algorithm that can efficiently cluster large datasets of time series.

Machine learning based approaches are successful but opaque

When evaluating the techniques for misuse-specific behavioural analysis, in the context of NLP, we find that machine learning approaches are the most successful. However, successful strategies for detecting misuse for AIaaS must also be transparent and accountable. Based on misuse highlighted by these machine learning algorithms, a service provider may deny clients access to their services. Given how fundamental these AI services may be to the customers' applications, revocation of access would be severe. As a result, any machine learning algorithms that are deployed must be able to explain and justify the decisions made. Though explainable AI is beyond the scope of this thesis, it remains a very active field ([58], [59]).

5.2 Broader Implications

5.2.1 Platform and Customer Incentives

Monitoring AI services for misuse raises larger questions about the incentives, responsibility and accountability of service providers. It will also be necessary to persuade customers to agree to the additional monitoring.

Arguably, there are several incentives for service providers to not intervene. By turning a blind eye, service providers can continue to profit, while avoiding any additional overhead costs associated with monitoring customer behaviour (figure 4.2). When a behaviour is flagged for further review, a decision must be made as to whether the customer is guilty of misuse. Misuse of a service can extend beyond criminal activity, resulting in potentially controversial decisions that are open to public scrutiny. There is also the idea that ignorance is bliss, and once the service provider learns of a problem, they now have a moral, and potentially legal, obligation to intervene.

On the other hand, there are significant repercussions to remaining ignorant or failing to act. We have already discussed the ramifications on the reputation of a company revealed to be supporting a controversial application. It is worth noting that the most prominent AIaaS providers are the large technology companies, including Amazon, Google, Microsoft and IBM, with the financial and technical capabilities to build and deploy AI systems. These huge corporations can easily afford the additional overheads associated with monitoring AIaaS. We have discussed the various benefits and overheads associated with different methods for misuse detection. Service providers can use this information to determine which are the most appropriate, while balancing the expenses.

Given that the major AIaaS providers are these large corporations, customers may be concerned that increased surveillance may result in the spying and collection of confidential corporate information. Nothing is preventing a service provider from terminating access to their AI services at any moment, and deploying their own version of the product. These fears are not completely unfounded, historically this has been in the business strategy of many large corporations. It was recently reported that Amazon employees were using data about independent sellers on the platform to develop their own competing products [60].

As the public becomes increasingly critical of these large service providers, it is unlikely that they will continue to ignore what is happening with their resources. Customers are also unlikely to be able to circumvent any new standards and policies put in place, given the unfortunate reality that the majority of these sophisticated AI services are offered by only a few large platforms.

5.2.2 Platform Power and Gatekeeping

With the low cost and barrier to entry, it is easy to envisage many customers coming to rely on AI services that may be intrinsic to the functionality of their application. There is an obvious issue of platforms gatekeeping who has access to these services, and, as a result, who may profit from them.

In response to ongoing content moderation concerns, we have observed a tendency in large tech platforms designating quasi-judicial, non-elected oversight bodies to establish guidelines and have the final word on what behaviour is permitted [61]. As previously stated, the major AIaaS providers are the tech 'giants', and monitoring AIaaS for misuse entails service providers making these judgements about which behaviours constitute misuse.

Service providers can choose how they define 'misuse'. Given their own commercial interests, 'misuse' to a service provider may even include using the service to build something competitive to a product of their own. Platforms have a significant amount of control over which applications can leverage their services and profit, raising significant monopolistic concerns. To ensure that behaviour of service providers is fair and consistent, any actions taken in response to suspected misuse must be made on a transparent and accountable basis.

Although in the context of hate speech detection, we found machine learning models the most successful, they are in fact the least explainable. Given how fundamental these services can be to an application, customers should be given the chance to appeal service suspensions and be notified for their wrongdoings. Service providers must be transparent in both their definitions of misuse and the procedures they employ to identify abuse.

5.2.3 Legal Aspects of AIaaS

AIaaS effectively serves as a collection of 'building blocks', allowing users to integrate AI into their systems with ease. Unfortunately, AIaaS vendors have no control over what customers construct with the technology. The service providers are also unaware of the data coming their way e.g. for an image analysis service, this data could easily be 'personal data' in the context of GDPR. In this dissertation we investigated misuse monitoring in three scenarios, each time processing the user data to a different extent. Monitoring for misuse involves processing the user data for a purpose that is intrinsically different to the original AI service e.g. passing text for a text-to-speech service through a hate speech classifier. Both the customer and end user must agree on how this data is processed. In general this could be addressed with appropriate contacts, terms and conditions. Although the full legal details are beyond the scope of this thesis, a comprehensive overview of the legal aspects and implications of AIaaS can be found in the recent work of Cobbe and Singh [62].

5.3 Concluding Remarks

By making state-of-the-art AI capabilities widely accessible at scale, often with little oversight, there is a large scope for the development of controversial, problematic and potentially illegal applications. We have illustrated the feasibility of a variety of approaches that would enable service providers to be more proactive in ensuring their AI services are used responsibly.

We have shown that it is possible to build generic usage landscaping, scenario specific and misuse specific methods for monitoring. Although there is currently no one-size-fits-all approach, we find that a cluster analysis based approach is an incredibly effective first step, as it can group similar behaviours with little information. We have also seen that the performance of misuse indicators is highly dependent on the choice of algorithm and hyperparameters. Future research should focus on developing a generalisable, low-cost and time-invariant approach to behavioural analysis.

Bibliography

[1] Worldwide spending on artificial intelligence is expected to double in four years, reaching $110 billion in 2024, according to new idc spending guide, Aug 2020.

[2] Mauro Ribeiro, Katarina Grolinger, and Miriam AM Capretz. Mlaas: Machine learning as a service. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 896–902. IEEE, 2015.

[3] Sean McGregor. Preventing repeated real world ai failures by cataloging incidents: The ai incident database. *arXiv preprint arXiv:2011.08512*, 2020.

[4] Leila Ouchchy, Allen Coin, and Veljko Dubljević. Ai in the headlines: the portrayal of the ethical issues of artificial intelligence in the media. *AI & SOCIETY*, 35(4):927–936, 2020.

[5] Vasile Tiple. Recommendations on the european commission's white paper on artificial intelligence-a european approach to excellence and trust, com (2020) 65 final (the'ai white paper'). 2020.

[6] European Digital Rights (EDRi). The eu's attempt to regulate big tech: What it brings and what is missing. `https://edri.org/our-work/eu-attempt-to-regulate-big-tech/`, December 2020. (Accessed on 05/25/2021).

[7] Seyyed Ahmad Javadi, Richard Cloete, Jennifer Cobbe, Michelle Seng Ah Lee, and Jatinder Singh. Monitoring misuse for accountable'artificial intelligence as a service'. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 300–306, 2020.

[8] Seyyed Ahmad Javadi, Richard Cloete, Chris Norval, and Jatinder Singh. Monitoring ai services for misuse. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, 2021.

[9] Arthur Zimek and Peter Filzmoser. There and back again: Outlier detection between statistical reasoning and data mining algorithms. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(6):e1280, 2018.

[10] Donald J Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, USA:, 1994.

[11] Alexandra Amidon Towards Data Science. How to apply k-means clustering to time series data. `https://towardsdatascience.com/how-to-apply-k-means-clustering-to-time-series-data-28d04a8f7da3`, July 2020. (Accessed on 05/20/2021).

[12] Romain Tavenard, Johann Faouzi, Gilles Vandewiele, Felix Divo, Guillaume Androz, Chester Holtz, Marie Payne, Roman Yurchak, Marc Rußwurm, Kushal Kolar, et al. Tslearn, a machine learning toolkit for time series data. *Journal of Machine Learning Research*, 21(118):1–6, 2020.

[13] Mohammad Shahrad, Rodrigo Fonseca, Íñigo Goiri, Gohar Chaudhry, Paul Batum, Jason Cooke, Eduardo Laureano, Colby Tresness, Mark Russinovich, and Ricardo Bianchini. Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider. In *2020 {USENIX} Annual Technical Conference ({USENIX}{ATC} 20)*, pages 205–218, 2020.

[14] Microsoft. Azurepublicdataset. `https://github.com/Azure/AzurePublicDataset/blob/master/AzureFunctionsDataset2019.md`, June 2020. (Accessed on 05/01/2021).

[15] Charlotte Jee. Ibm says it is no longer working on face recognition because it's used for racial profiling. *MIT Technology Review*, 11, 2020.

[16] NIST US National Institute of Standards and Technology. Nist study evaluates effects of race, age, sex on face recognition software. `https://www.nist.gov/news-events/news/2019/12/nist-study-evaluates-effects-race-age-sex-face-recognition-software`, December 2019. (Accessed on 05/02/2021).

[17] Amazon Staff. We are implementing a one-year moratorium on police use of rekognition. `https://www.aboutamazon.com/news/policy-news-views/we-are-implementing-a-one-year-moratorium-on-police-use-of-rekognition`, June 2020. (Accessed on 05/02/2021).

[18] Katherine Anne Long. Amazon, microsoft team against facial recognition lawsuits. *The Seattle Times*, April 2021.

[19] Jeffrey Dastin. Amazon extends moratorium on police use of facial recognition software. `https://www.reuters.com/technology/exclusive-amazon-extends-moratorium-police-use-facial-recognition-software-2021-05-18/`, May 2021. (Accessed on 05/23/2021).

[20] Jay Greene. Microsoft won't sell police its facial-recognition technology, following similar moves by amazon and ibm. *Washington Post*, 11, 2020.

[21] Amnesty International. Russia: Police target peaceful protesters identified using facial recognition technology. `https://www.amnesty.org/en/latest/news/2021/04/russia-police-target-peaceful-protesters-identified-using-facial-recognition-technology/`, April 2021. (Accessed on 05/08/2021).

[22] Business & Human Rights Resource Centre. Us police are using facial recognition technology at protests - adding to systemic racism. `https://www.business-humanrights.org/en/blog/us-police-are-using-facial-recognition-technology-at-protests-adding-to-systemic-racism/`, August 2020. (Accessed on 05/08/2021).

[23] Office of the High Commissioner for Human Rights. Impact of new technologies. `https://www.ohchr.org/EN/Issues/DigitalAge/Pages/ReportDigitalAgeAssembliesandProtests.aspx`, June 2020. (Accessed on 05/08/2021).

[24] Amnesty International. Ban dangerous facial recognition technology that amplifies racist policing. `https://www.amnesty.org/en/latest/news/2021/01/ban-dangerous-facial-recognition-technology-that-amplifies-racist-policing/`, January 2021. (Accessed on 05/08/2021).

[25] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[26] Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

[27] Charles Otto, Dayong Wang, and Anil K Jain. Clustering millions of faces by identity. *IEEE transactions on pattern analysis and machine intelligence*, 40(2):289–303, 2017.

[28] Nameirakpam Dhanachandra and Yambem Jina Chanu. A survey on image segmentation methods using clustering techniques. *European Journal of Engineering and Technology Research*, 2(1):15–20, 2017.

[29] Chris Biemann. Chinese whispers-an efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of TextGraphs: the First Workshop on Graph Based Methods for Natural Language Processing*, pages 73–80, 2006.

[30] Antonio Di Marco and Roberto Navigli. Clustering and diversifying web search results with graph-based word sense induction. *Computational Linguistics*, 39(3):709–754, 2013.

[31] Roy Klip. Fuzzy face clustering for forensic investigations. 2019.

[32] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[33] P Jonathon Phillips, Harry Wechsler, Jeffery Huang, and Patrick J Rauss. The feret database and evaluation procedure for face-recognition algorithms. *Image and vision computing*, 16(5):295–306, 1998.

[34] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007.

[35] David Sandberg. Face recognition using tensorflow. `https://github.com/davidsandberg/facenet`, April 2018. (Accessed on 04/24/2021).

[36] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multi-task cascaded convolutional networks. *CoRR*, abs/1604.02878, 2016.

[37] Hiroki Taniai. Facenet implementation by keras. `https://github.com/nyoki-mtl/keras-facenet`, February 2018. (Accessed on 04/24/2021).

[38] Yandong Guo, Lei Zhang, Yuxiao Hu, X. He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *ECCV*, 2016.

[39] Gradient Flow. 2020 nlp survey report. `https://gradientflow.com/2020nlpsurvey/`. (Accessed on 04/28/2021).

[40] Hate speech — definition in the cambridge english dictionary. `https://dictionary.cambridge.org/us/dictionary/english/hate-speech`. (Accessed on 04/28/2021).

[41] Calum Patterson. Twitch ban streamer after viewer sends racist text-to-speech donation. *Dexerto*, 2019.

[42] Björn Ross, Michael Rist, Guillermo Carbonell, Benjamin Cabrera, Nils Kurowsky, and Michael Wojatzki. Measuring the reliability of hate speech annotations: The case of the european refugee crisis. *arXiv preprint arXiv:1701.08118*, 2017.

[43] L Matsakis. To break a hate-speech detection algorithm, try'love'. wired, 2018.

[44] William Warner and Julia Hirschberg. Detecting hate speech on the world wide web. In *Proceedings of the second workshop on language in social media*, pages 19–26, 2012.

[45] Sean MacAvaney, Hao-Ren Yao, Eugene Yang, Katina Russell, Nazli Goharian, and Ophir Frieder. Hate speech detection: Challenges and solutions. *PloS one*, 14(8):e0221152, 2019.

[46] Irina Rish et al. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46, 2001.

[47] Christian Gourieroux and Alain Monfort. Asymptotic properties of the maximum likelihood estimator in dichotomous logit models. *Journal of Econometrics*, 17(1):83–97, 1981.

[48] Nina Bauwelinck and Els Lefever. Measuring the impact of sentiment for hate speech detection on twitter. In *HUSO 2019: The Fifth International Conference on Human and Social Analytics*, pages 17–22. IARIA, International Academy, Research, and Industry Association, 2019.

[49] Clayton Hutto and Eric Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 8, 2014.

[50] Bertie Vidgen and Leon Derczynski. Directions in abusive language training data, a systematic review: Garbage in, garbage out. *Plos one*, 15(12):e0243300, 2020.

[51] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 11, 2017.

[52] Antigoni Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 12, 2018.

[53] Zeerak Waseem and Dirk Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California, June 2016. Association for Computational Linguistics.

[54] Martin F Porter. Readings in information retrieval. *San Francisco, CA, USA: Morgan Kaufmann Publishers Inc*, pages 313–316, 1997.

[55] Juan Ramos et al. Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 29–48. Citeseer, 2003.

[56] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.

[57] David L Olson and Dursun Delen. *Advanced data mining techniques*. Springer Science & Business Media, 2008.

[58] Zohreh Shams, Botty Dimanov, Sumaiyah Kola, Nikola Simidjievski, Helena Andres Terre, Paul Scherer, Urska Matjasec, Jean Abraham, Mateja Jamnik, and Pietro Lio. Rem: An integrative rule extraction methodology for explainable data analysis in healthcare. *bioRxiv*, 2021.

[59] David Gunning. Explainable artificial intelligence (xai). *Defense Advanced Research Projects Agency (DARPA), nd Web*, 2(2), 2017.

[60] Dana Mattioli. Amazon scooped up data from its own sellers to launch competing products. *The Wall Street Journal, April*, 23, 2020.

[61] Kate Klonick. Inside the making of facebook's supreme court. *The New Yorker*, 2021.

[62] Singh Cobbe. Artificial intelligence as a service: Legal responsibilities, liabilities, and policy challenges. SSRN, 2021.