Next Generation Machine Learning Based Real Time Fraud Detection

Dissertation

Rui Filipe Laranjeira da Costa
up199600952
ei12150@fe.up.pt

Universidade do Porto

Faculdade de Engenharia

FEUP

Supervision: João Pedro Mendes Moreira
Company: WeDo Technologies

February 24, 2020

Next Generation Machine Learning Based Real Time Fraud Detection

Rui Filipe Laranjeira da Costa

Master in Informatics and Computing Engineering

February 24, 2020

Abstract

In telecommunications there are several schemes to defraud the telecommunications companies causing great financial losses. We can considerer three major categories in telecom fraud based on who the fraudsters are targeting. These categories are: Traffic Pumping Schemes, Defraud Telecom Service Providers, Conducted Over the Telephone. Traffic Pumping Schemes use "access stimulation" techniques to boost traffic to a high cost destination, which then shares the revenue with the fraudster. Defraud Telecom Service Providers are the most complicated, and exploit telecom service providers using SIP trunking, regulatory loopholes, and more. Conducted Over the Telephone, also known as "Phone Fraud", this category covers all types of general fraud that are perpetrated over the telephone. Telecommunications fraud negatively impacts everyone, including good paying customers. The losses increase the companies operating costs. While telecom companies take every measure to stop the fraud and reduce their losses, the criminals continue their attacks on companies with perceived weaknesses. The telecom business is facing a serious hazard growing as fast as the industry itself. Communications Fraud Control Association (CFCA) stated that telecom fraud represented nearly $30 billion globally in 2017 [1]. Another problem is to stay on top of the game with effective anti-fraud technologies. The need to ensure a secure and trustable Internet of Things (IoT) network brings the challenge to continuously monitor massive volumes of machine data in streaming. Therefore a different approach is required in the scope of Fraud Detection, where detection engines need to detect risk situations in real time and be able to adapt themselves to evolving behavior patterns. Machine learning based online anomaly detection can support this new approach. For applications involving several data streams, the challenge of detecting anomalies has become harder over time, as data can dynamically evolve in subtle ways following changes in the underlying infrastructure. The goal of this paper is to research existing online anomaly detection algorithms to select a set of candidates in order to test them in Fraud Detection scenarios. Define a real time monitoring architecture that can scale as the network of devices monitored grows. From the research work carried out and the knowledge about the nature of the business, it was possible to develop a clustering methodology over the data streams that allows to detect patterns on entities. The methodology used is based on the concept of micro-cluster, which is a structure that maintains a summary of the patterns detected on entities.

Acknowledgements

I would like to express my very great appreciation to professor João Moreira for all the support and guidance to carry out this work. I would like to offer my special thanks to Carlos Martins, from Wedo Technologies, for all the clarifications and support material provided.I am particularly grateful for the assistance given by Raul Azevedo, from Wedo Technologies.

Rui Filipe Laranjeira da Costa

Contents

List of Figures

List of Tables

Abbreviations

IoT	Internet of Things
GSM	Mobile Communications
MSC	mobile switching center
BSC	base station controller
EDRs	Event Detail Records
CDRs	Charging Detail Records
HEOM	Heterogeneous Euclidean-Overlap Metric
UFFT	Ultra Fast Forest of Trees
IRSF	International Revenue Sharing Fraud
PBX	Private Branch Exchange
MINAS	Multi-Class Learning Algorithm for data streams
PSTN	Public Switched Telephone Network

Chapter 1

Introduction

A telecommunications company has a large volume of data for incoming and outgoing calls. The company wants to detect fraudulent calls in real time so that they can notify customers or shut down service for a specific number. One type of SIM fraud involves multiple calls from the same identity around the same time but in geographically different locations. To detect this type of fraud, the company needs to examine incoming phone records and look for specific patterns—in this case, for calls made around the same time in different countries/regions. Any phone records that fall into this category are written to storage for subsequent analysis. Data analysis is done long after the fraudulent situation occurs. Although it was noted that there was a fraudulent situation nothing was done in real time to prevent fraud, continuing the attacker to perpetuate their fraudulent actions.

One of the most reported scam scenarios is the use of premium rate numbers. These premium rate numbers are usually to a high cost destination. The owner of the number will offer to share the revenue generated from calls to these numbers with anyone who sends them traffic.This means that a fraudster who generates bogus or stimulated traffic to that destination will receive a part of the profit for each completed call. This type of scheme is known as the International Revenue Share Fraud (IRSF).

As new and diverse technologies emerge, new techniques also emerge to defraud the telecommunications companies. Large company telecom systems called ”private branch exchanges” (PBX), cellular communications, and virtual private networks are a few examples of some of the new technologies that telecommunications companies have had to protect from fraudsters.

For a better understanding of telecommunications fraud, it is necessary to understand how information is exchanged and who are the stakeholders in these communication exchanges [2].

In the Global Mobile Communications System (GSM), a cell phone is a portable phone that receives or makes calls through a base station or transmission tower. Radio waves are used to transfer signals to and from the cell phone, Radio base station controller (BSC) controls base stations or cell sites that receive or transmit radio signals, a mobile switching center (MSC) is mostly associated with communications switching function, such as call set-up, release and routing. Example of communication between two mobile devices of different operators: the mobile phone that initially calls is connected to an initial operator BSC which in turn is connected to an MSC of that operator. The MSC provides connection to the Public Switched Telephone Network(PSTN) network that routes the call to the MSC of the final operator, MSC checks which BSC its subscriber is registered in and forwards the call to that BSC which in turn communicates by radio signals with the final mobile device. In all this process, the price to be paid for a service between operators of different networks is based on a prior agreement between operators. The communication between the switches always has a fee to be charged for the transport of the traffic. The more switch you need to make a

connection, the more expensive the service will be. For a better understanding of telecommunications fraud it is always necessary to analyze the monetary amounts involved in each communication as well as who can benefit from the amounts distributed by the various carriers by which the service is taken. Communications are not always so simple to control because of the various types of traffic. National traffic is between subscribers from the same country, and can be divided into On-Net and Off-Net. On-Net is traffic between subscribers of the same operator, Off-Net is traffic between subscribers of different operators. International Traffic Off-Net is between subscribers of different operators but whoever receives the call is in another country. Roaming is when a user leaves the network in their home country and uses the network in another country to make communications. Fraud in telecommunications can be classified into two dimensions: the technique used and the type of fraud. We must take into account the context in which fraud is carried out. This information is useful for determining the set of attributes for the two dimensions of the model. Enabler Technique is the method or technique of getting access to the goods or service and perpetrating fraud. Fraud Type is the fraud committed to get illegal benefits. Given an inhibiting technique and a type of fraud, we can select in the data structure of an instance a set of attributes with the ability to build a clustering model to detect patterns in the data stream that lead us to conclude that a certain entity is committing fraud. The attributes to be selected can variate according to the inhibiting technique and a type of fraud, so the deep knowledge of how a fraud is perpetuated is fundamental for the construction of the clustering model.

1.1 Context

This work is in context of WeDo Technologies and is supported by SonaeIM.LAB@FEUP project. WeDo Technologies (www.wedotechnologies.com) is the market leader in Revenue Assurance and Fraud Management for Telecommunications, developing and providing software products and specialized services globally.

With Clients in more than 100 countries, including major references like Orange, Vodafone, BT, Telefonica, Verizon, Etisalat, among many others, WeDo Technologies has more than 600 employees, worldwide. The product development unit is based in Braga, Portugal.

1.2 Motivation and Goals

Anomaly detection in data streams is a heavily studied area of data science and machine learning. Many anomaly detection approaches exist, both supervised (e.g. support vector machines and decision trees) and unsupervised (e.g. clustering), yet the vast majority of anomaly detection methods are for processing data in batches, and unsuitable for real time streaming applications. On the other hand, the detection of fraud is not a topic that is widely addressed in scientific studies, as it comprises knowing the nature of the business in order to be able to detect fraud, however the techniques and methodologies used in anomaly detection in data streams can serve the basis for fraud detection. Streaming applications impose unique constraints and challenges for machine learning models. These applications involve analyzing a continuous sequence of data occurring in real time. In contrast to batch processing, the full dataset is not available. Data streams produce a huge amount of data that introduce new constraints in the design of learning algorithms: limited computational resources in terms of memory, cpu power, and communication bandwidth. Although some batch techniques may work well in certain situations the focus of this paper is on methods for online anomaly detection and the unsupervised approach.

International Revenue Sharing Fraud (IRSF) is the most persistent type of fraud within the telecom industry. It represents around $10 billions annual loss to Telecom providers. Fraudsters often utilize illegal resources to gain access to an operator's network in order to bring traffic to phone numbers obtained from

an International Premium Rate Number provider. This is the most challenging fraud to eliminate due to the complexity of the mobile network system and the involvement of multiple operators, it is usually perpetuated by organized groups that use illegal connections to bring a high volume of calls at a high cost 'revenue share' service numbers, taking advantage on the roaming capabilities of SIM cards.

Another scenario of fraud in IRSF has as an inhibitory source the Wangiri scheme, that comes from Japanese, one touch and then hangs up. Criminals use this technique to trick you into calling premium rate numbers. A fraudster will set up a system (for instance using botnets) to dial a large number of random phone numbers. Each calls rings just once, then hangs up, leaving a missed call on the recipients' phone. Subscribers are tempted to return the call but the call is made for an international premium rate number. In this situation the detection of patterns is fundamental because this type of attack is usually done by an application that makes random calls in a fixed time interval. The search for these patterns can prevent this type of fraud.

1.3 Structure of the Dissertation

In addition to the introduction, this dissertation contains 4 more chapters.

In chapter 2, the state of art is described.

In chapter 3 is presented Apache Kafka, used in the experiments, and Kafka architecture. Is analyzed the data structure of the xDRs instance in Kafka stream.

In chapter 4 the data flow clustering method is described and the clustering method is presented. The experiments and results are described.

In chapter 5 some conclusions are written about this article and possible future works to be developed in this area. Possible solution is presented to detect novelties in the data stream.

Chapter 2

Related Work

The focus of the state-of-art was the clustering for evolving data streams with the goal of detecting the outliers that maybe be considered an anomaly and consequently be placed in fraud scenario. In the analyzed works [3–7] the data stream clustering algorithms are divided into three main approaches: data summarization clustering, online (real-time) clustering, and time-series clustering. For a first approach to the proposed challenge for detecting real-time anomalies in data streams involving mobile communications it was understood that the approach of a data stream summarization algorithm would be the most appropriate and the DenStream Algorithm was selected for this first approach.

Stream clustering algorithms analyzed:

- StreamKM++: It computes a small weighted sample of the data stream and it uses the k-means++ algorithm as a randomized seeding technique to choose the first values for the clusters. To compute the small sample, it employs coreset constructions using a coreset tree for speed up.
- CluStream: It maintains statistical information about the data using micro-clusters. These micro-clusters are temporal extensions of cluster feature vectors. The micro-clusters are stored at snapshots in time following a pyramidal pattern. This pattern allows to recall summary statistics from different time horizons.
- ClusTree: It is a parameter free algorithm automatically adapting to the speed of the stream and it is capable of detecting concept drift, novelty, and outliers in the stream. It uses a compact and self-adaptive index structure for maintaining stream summaries.
- DenStream: It uses dense micro-clusters (named core-micro-cluster) to summarize clusters. To maintain and distinguish the potential clusters and outliers, this method presents core-micro-cluster and outlier microcluster structures.
- CobWeb: One of the first incremental methods for clustering data. It uses a classification tree. Each node in a classification tree represents a class (concept) and is labeled by a probabilistic concept that summarizes the attribute-value distributions of objects classified under the node.

In all the works found on machine learning in data streams the data structures of the presented instances have only numeric attributes. The attributes of data structures in telecommunications are not always numerical, so Heterogeneous Euclidean Overlap Metric (HEOM) [8] will be presented as the metric to calculate the distance between two instances of the data stream.

There are not many works that specifically talk about data stream clustering to detect online anomalies, and those who approach the subject do so in an empirical way, and in the telecommunications context no work was found that addressed this specific topic

2.1 Clustering For Data Streams

Clustering is a very important task on the approach for mining of evolving data streams. Is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters). Clustering methods are widely used in data mining. They are either used to get insight into data distribution or as a preprocessing step for other algorithms. The most common approaches use distance between examples as similarity criteria. The data stream clustering problem is defined as to maintain a continuously consistent good clustering of the sequence observed so far, using a small amount of memory and time. The issues are imposed by the continuous arriving of instances, and the need to analyze them in real time. Beside the limited memory and one-pass constraints, the nature of evolving data streams implies the following requirements for stream clustering: no assumption on the number of clusters, discovery of clusters with arbitrary shape and ability to handle outliers. From the studied algorithms, DenStream was chosen for this first approach.

2.2 Heterogeneous Euclidean-Overlap Metric (HEOM)

We need to deal with both continuous and nominal attributes and for that we need to use a heterogeneous distance function that uses different attribute distance functions on different kinds of attributes. One approach that has been used is to use the overlap metric for nominal attributes and normalized Euclidean distance for linear attributes. The function

$$d_a(x,y) = \begin{cases} 1, & \text{if x or is unknown, else} \\ overlap(x,y), & \text{if a is nominal, else} \\ rn\_diff_a(x,y) & \end{cases}$$

defines the distance between two values x and y of a given attribute a, where

$$overlap(x,y) = \begin{cases} 0, & \text{if } x = y \\ 1, & \text{otherwise} \end{cases}$$

and

$$rn\_diff_a(x,y) = \frac{|x-y|}{range_a}$$

The value $range_a$ is used to normalize the attributes, and is defined as:

$$range_a = max_a - min_a$$

This means that it is possible for a new input vector to have a value outside this range and produce a difference value greater than one. However, such cases are rare, and when they do occur, a large difference may be acceptable anyway. The normalization serves to scale the attribute down to the point where differences are almost always less than one. The overall distance between two (possibly heterogeneous) input vectors x and y is given by the Heterogeneous Euclidean-Overlap Metric function $HEOM(x,y)$:

$$HEOM(x,y) = \sqrt{\sum_{a=1}^{m} d_a(x_a, y_a)^2}$$

This distance function removes the effects of the arbitrary ordering of nominal values, but its overly simplistic approach to handling nominal attributes fails to make use of additional information provided by nominal attribute values that can aid in generalization.

2.3 DenStream

Density-Based Clustering algorithm is based on the traditional DBSCAN. We can even say that it is an incremental DBSCAN adapted to the data streams that aims at the discovery of clusters with arbitrary shape. It is based on micro-clusters to summarize a clusters with arbitrary shape. The online component uses two buffers called potential micro-clusters (p-micro-clusters) and outliers micro-clusters (o-micro-clusters). When a new object can not be allocated in p-micro-clusters, it is stored in o-micro-clusters until it is promoted or removed, which in fact characterizes it as a real outlier. Periodically an analysis on the buffers is performed, searching for micro-clusters that should be promoted to p-micro-clusters or removed from the buffers. This analysis depends on input parameters provided by the user. DenStream was used in an early phase of the investigation. However, the algorithm did not produce the desired results for this kind of context..

2.3.1 Fundamental Concepts

Cluster partitions on evolving data streams are often computed based on certain time intervals (or windows). There are three well-known window models: landmark window, sliding window and damped window.

DenStream use the damped window model, in which the weight of each data point decreases exponentially with time `t` via a fading function that is described next.

2.3.2 Fading Function

In most data stream scenarios, more recent data can reflect the emergence of new trends or changes in data distribution. The study carried out on the nature of the business led us to conclude that the damped window model would be the option to be used. In order to determine whether or not a given micro-cluster should be discarded in order to allow other micro-clusters to be created, the following fading function is used.

$$f(t) = 2^{-\lambda.t}$$

where `t` is the time and $\lambda > 0$. The exponentially fading function is widely used in temporal applications where it is desirable to gradually discount the history of past behavior. The higher the value of λ, the lower importance of the historical data compared to more recent data. And the overall weight of the data stream is a constant

$$W = v(\sum_{t=0}^{t=t_c} 2^{-\lambda.t}) = \frac{v}{1-2^{-\lambda}}$$

where $t_c(t_c \rightarrow \infty)$ is the current time, and v denotes the speed of the stream, i.e., the number of points arrived in one unit time. With Kafka streams we do not need to worry about losing an instance, because an instance is only sent after the later one is consumed. This is particularly useful because, due to large CPU usage in a short time or another limited resource, the instance consumption may slow slightly. However, as soon as the feature being used as a bottleneck is released, instances can be retrieved and re-take the normal time window. It is important that, in the configuration of the parameters in the DenStream algorithm, there is some margin in the resources used to allow this type of recoveries. If resources are used to the limit of their capabilities and if a slowdown occurs, the delay time relative to the last input instance is increased, and after a certain time the older instances are discarded by the Zookeeper, making it impossible to consume. Zookeeper is a top-level software developed by Apache that acts as a centralized service and is used to synchronization within Kafka streams.

2.3.3 Core Object

A core object is defined as an object, in whose ϵ neighborhood the overall weight of data points is at least an integer μ.

2.3.4 Density-area

A density area is defined as the union of the ϵ neighborhoods of core objects.

2.3.5 Core-micro-cluster

A core-micro-cluster, abbr. c-micro-cluster, at time t is defined as $CMC(\omega, c, r)$ for a group of close points $p_{i_1}, ..., p_{i_n}$ with time stamps $T_{i_1}, ..., T_{i_n}$.

$$\omega = \sum_{j=1}^{n} f(t - T_{i_j})$$

$\omega > \mu$, is the weight.

$$c = \frac{\sum_{j=1}^{n} f(t - T_{i_j}) p_{i_j}}{\mu}$$

is the center.

$$r = \frac{\sum_{j=1}^{n} f(t - T_{i_j}) dist(p_{i_j}, c)}{\mu}$$

$r \leq \epsilon$, is the radius, where $dist(p_{i_j}; c)$ denotes the Heterogeneous Euclidean Overlap distance between point p_{i_j} and the center c.

The weight of c-micro-clusters must be above or equal to μ and the radius must be below or equal to ϵ. Because of the constraint on radius, N_c, the number of c-micro-clusters is much larger than the number of natural clusters. On the other hand, it is significantly smaller than the number of points in the data stream due to its constraint on weight. Since each point is uniquely assigned to one of the c-micro-clusters, N_c is below or equal to $\frac{W}{\mu}$. When a clustering request arrives, each c-micro-cluster will be labeled to get the final result, as illustrated in Figure 2.1.

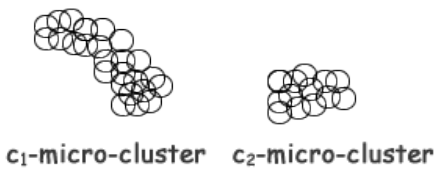

Figure 2.1: Representation of c-micro-cluster

2.3.6 Potential c-micro-cluster

A potential c-micro-cluster, abbr. p-micro-cluster, at time t for a group of close points $p_{i_1}, ..., p_{i_n}$ with time stamps $T_{i_1}, ..., T_{i_n}$ is defined as $\{\overline{CF^1}, \overline{CF^2}, \omega\}$.

$$\omega = \sum_{j=1}^{n} f(t - T_{i_j})$$

$\omega \geq \beta\mu$, is the weight. $\beta, 0 < \beta \leq 1$, is the parameter to determine the threshold of outlier relative to c-micro-clusters.

$$\overline{CF^1} = \sum_{j=1}^{n} f(t - T_{i_j}) p_{i_j}$$

is the weighted linear sum of the points.

$$\overline{CF^2} = \sum_{j=1}^{n} f(t - T_{i_j}) p_{i_j}^2$$

is the weighted squared sum of the points.

$$c = \frac{\overline{CF^1}}{\omega}$$

is the center of p-micro-cluster.

$$r = \sqrt{\frac{\left|\overline{CF^2}\right|}{\omega} - \left(\frac{\left|\overline{CF^1}\right|}{\omega}\right)^2}$$

is the radius of p-micro-cluster, where $r \leq \epsilon$.

2.3.7 Outlier micro-cluster

An outlier micro-cluster, abbr. o-micro-cluster, at time t for a group of close points $p_{i_1}, ..., p_{i_n}$ with time stamps $T_{i_1}, ..., T_{i_n}$ is defined as $\{\overline{CF^1}, \overline{CF^2}, \omega, t_0\}$. The definitions of $\overline{CF^1}, \overline{CF^2}, \omega$, center and radius are the same as the p-micro-cluster. $t_0 = T_{i_1}$ denotes the creation time of the o-micro-cluster, which is used to determine the life span of the o-micro-cluster. However $\omega < \beta\mu$. That is, because the weight is below the threshold of outlier, the micro-cluster corresponds to outliers.

P-micro-clusters and o-micro-clusters can be maintained incrementally.

2.3.8 DenStream Pseudocode

```
DEN-STREAM(Stream, λ, μ, β)
    Input: a stream of points, decaying factor λ,
        core weight threshold μ, tolerance factor β

▷ Online phase
T_p ← ⌈(1/λ) log(βμ/(βμ−1))⌉
for each new point that arrives
    do try to merge to a p-microcluster; if not possible,
        merge to nearest o-microcluster
            if o-microcluster weight > βμ
                then convert the o-microcluster to p-microcluster
                else create a new o-microcluster

▷ Offline phase
if (t mod T_p = 0)
    then for each p-microcluster c_p
            do if w_p < βμ
                then remove c_p
        for each o-microcluster c_o
            do if w_o < (2^{−λ(t−t_o+T_p)} − 1)/(2^{−λT_p} − 1)
                then remove c_o
        apply DBSCAN using microclusters as points
```

Figure 2.2: The DenStream algorithm.

The DenStream method has two phases, an online phase and an offline phase. Their pseudocode is shown in figure 2.2. In the online phase, every time a new point arrives, DenStream first tries to merge it into one of the potential micro-clusters. If this is not possible, it then tries to merge the point with an outlier micro-cluster. If the weight of the outlier micro-cluster has increased enough to be a potential micro-cluster, the micro-cluster is promoted. Otherwise, DenStream starts a new outlier micro-cluster. The offline phase consists of removing the micro-clusters that have not reached a sufficient weight, and performing a batch DBSCAN clustering.

2.3.9 Pattern Learning in Data Streams

As the study on the nature of fraud was completed and how it perpetuated, we concluded that the detection of fraud in the data stream could be solved using the search for statistically relevant patterns between instances of the same entity.

The problem of finding sequential patterns in static databases had been studied extensively in the past years, however mining sequential patterns in the data streams is still an active field for researchers. Unlike mining static databases, mining data streams poses many new challenges. First, it is unrealistic to keep the entire stream in the main memory or even in a secondary storage area, since a data stream comes continuously and the amount of data is unbounded. Second, traditional methods of mining on stored datasets by multiple scans are infeasible, since the streaming data is passed only once. Third, mining streams requires fast, real-time processing in order to keep up with the high data arrival rate and mining results are expected to be available within short response times. In addition, the combinatorial explosion of itemsets exacerbates mining frequent itemsets over streams in terms of both memory consumption and processing efficiency.

Upon completion of the implementation of the first approach, it was quickly realized that for this type of instance this approach was totally unrealistic, and from the analysis of the results it was possible to conclude that for quantitative attributes, its value does not represent any kind of weight relative to another instance. We can conclude nothing about the proximity of one instance relative to another by its quantitative values. Nothing can be said for the distance between two consecutive values being closer or further from other nonconsecutive values. Their quantitative attributes should be considered as qualitative attributes. These qualitative characteristics should be considered as discrete variables, i.e. they appear in the data stream with few variations.

The conclusions drawn from the study of data streams led us to conclude that the best approach for fraud detection would be to uses the pattern learning from data streams.

Chapter 3

Kafka the Distributed Messaging Platform

Distributed messaging platform like Apache Kafka is used as a data streaming pipeline, for real time streams of data. Kafka can collect big data or to do real time analysis or both which is used for program scaling purposes. The streams analyzed in the scope of this project are managed by Apache Kafka. So, it is better to look how is the Apache Kafka architecture and how the flow of data is proceeded. This allow us to better decide which amount of memory, cpu or bandwidth will be necessary to build a model for a fraud scenario.

Kafka was first built by the LinkedIn technical team and was used to collect user activity data on their portal. We have to analyze both the logical and physical architecture of Kafka. Every message in Kafka topics is a collection of bytes. This collection is represented as an array. Producers are the applications that store information in Kafka queues. They send messages to Kafka topics that can store all types of messages. Every topic is further differentiated into partitions. Each partition stores messages in the sequence in which they arrive. There are two major operations that producers/consumers can perform in Kafka. Producers append to the end of the write-ahead log files. Consumers fetch messages from these log files belonging to a given topic partition. Physically, each topic is spread over different Kafka brokers, which host one or two partitions of each topic. A Kafka cluster is basically composed of one or more servers (nodes). A

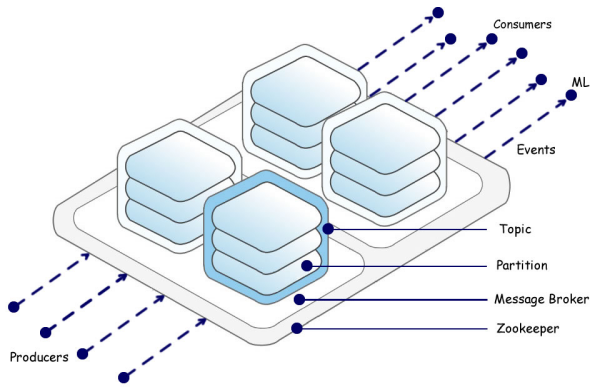

Figure 3.1: Data Streams Kafka

typical Kafka cluster consists of multiple brokers. It helps in load-balancing message reads and writes to the cluster. Each of these brokers is stateless. However, they use Zookeeper to maintain their states. Each topic partition has one of the brokers as a leader and zero or more brokers as followers. The leaders manage

any read or write requests for their respective partitions. Followers replicate the leader in the background without actively interfering with the leader's working. As illustrated by figure 3.1 this project will work by subscribing an topic in Apache Kafka and doing machine learning over the data stream received. Each event, that can be either making a phone call, or terminating a phone call, it will be used to build our clustering model. Our model will decide if a given event is a valid event or if it is a outlier event as that can be placed in a fraud scenario.

3.1 Data Structure

Telecommunication companies have to keep knowledge of every service that is used by his subscriber for the purpose of charging the subscriber for the service provided. All information about the service provided, is grouped together by the MSCs and sent to the billing system through an offline, file based protocol. These records are known as Charging Detail Records (CDRs) or Event Detail Records (EDRs), or the combination of the two, known as xDRs records.
In telecommunications, the xDRs data format has the attributes in table 3.1. Each attributes has different values according with service required.

- EVENT_TYPE - Possible Values
  MOC - Mobile Originated Call
  MTC - Mobile Terminated Call
  SMSMO - SMS Originated
  SMSMT - SMS Terminated
  GPRS - Packet Switch Data Session
  MSESS - Mobile Session (Voice over LTE)
  MSG - SMS over IP

- SERVED_ENTITY_ID - Description
  The Mobile Subscriber ISDN number. The number from the subscriber who has used the network.

- IMSI - Description
  International Mobile Subscriber Identity from the subscriber who used the network.

- IMEI - Description
  The International Mobile Equipment Identity number that identifies the equipment used by the subscriber during the event.

- A_NUMBER - Description
  The number from which the call was originated in the case of terminated calls/SMS. The number representation must be in International format, meaning the number must begin with the Country Code.

- B_NUMBER - Description
  The number for which the call was terminated. The number representation must be in International format, meaning the number must begin with the Country Code.

- C_NUMBER - Description
  The number identifying the original calling number in a call forwarding context. The number representation must be in International format, meaning the number must begin with the Country Code.

- START_DATE - Description
  Event start time given in the local time.

- UTC_TIME_OFFSET - Description
  The difference between local time and UTC time.

- DURATION - Description
  Event duration in seconds.

- CALL_TYPE - Possible Values
  1 - OnNet (when the call origin/destination is also a operator subscriber)
  2 - OffNet (when the call origin/destination is a number their national network)
  3 - International (when the call origin/destination is an international number)

- CHARGE_AMOUNT - Description
  The charge applied to the event.

- CAUSE_FOR_TERMINATION - Possible Values
  0 - normal release
  3 - unsuccessful call attempt
  4 - abnormal release
  16 - volume limit
  17 - time limit

- GGSN_ADDRESS - Description
  IP Address from the GGSN used

- SGSN_ADDRESS - Description
  IP Address from the SGSN used

- UP_VOLUME - Description
  Uplink volume (in bytes)

- DOWN_VOLUME - Description
  Downlink volume (in bytes)

- APN - Description
  Access Point Name

- CELL_ID - Description
  The identity of the cell from where the event is originated or terminated, depending on the event direction.

- ROAMING_INDICATOR - Possible Values
  0 - No Roaming (subscriber using the operator network)
  1 - Roaming Inbound (roamer in using the operator network)
  2 - Roaming Outbound (subscriber using an external network)

- VISITED_COUNTRY_CODE - Description
  Country code from the visited country in a roaming scenario (only for roamers out)
- LOCATION_AREA_CODE - Description
  Identification of the Location Area Code of the mobile equipment handling the call
- IN_TRUNK_GROUP - Description
  Incoming trunk group where the communication signal was transmitted
- IN_TRUNK_NUMBER - Description
  Incoming trunk number where the communication signal was transmitted
- OUT_TRUNK_GROUP - Description
  Outgoing trunk group where the communication signal was transmitted
- OUT_TRUNK_NUMBER - Description
  Outgoing trunk number where the communication signal was transmitted
- SERVED_PDP_ADDRESS - Description
  IP address used in the Packed Data Protocol
- MOBILE_SESSION_SERVICE - Description
  The mobile session service refers to the direction of a VoLTE call
  Possible Values:
  1 - MO Voice over LTE
  2 - MT Voice over LTE
- MESSAGING_EVENT_SERVICE - Description
  The messaging event service refers to the direction of a SMS over IP
  Possible Values:
  1 - MO SMS over IP (IMS based SMS)
  2 - MT SMS over IP (IMS based SMS)
- CALL_FORWARDING - Possible Values
  1 - Call Forwarding
  0 - No Call Forwarding
- CALL_CONFERENCE - Possible Values
  1 - Call Conference
  0 - No Call Conference
- CALL_HOLD - Possible Values
  1 - Call Hold
  0 - No Call Hold
- VOICEMAIL - Possible Values
  1 - Voicemail
  0 - No Voicemail
- VOICEMAIL_CALLBACK - Possible Values
  1 - Voicemail call back
  0 - No Voicemail call back

- EMERGENCY_CALL - Possible Values
398 1 - Emergency Call
0 - No Emergency Call

#	Column Name	Mandatory	Type
1	EVENT_TYPE	Yes	String
2	SERVED_ENTITY_ID	Yes	String
3	IMSI	No	String
4	IMEI	No	String
5	A_NUMBER	Yes	String
6	B_NUMBER	Yes	String
7	C_NUMBER	No	String
8	START_DATE	Yes	String
9	UTC_TIME_OFFSET	Yes	String
10	DURATION	Yes	Number
11	CALL_TYPE	Yes	Number
12	CHARGE_AMOUNT	No	Number
13	CAUSE_FOR_TERMINATION	No	Number
14	GGSN_ADDRESS	No	String
15	SGSN_ADDRESS	No	String
16	UP_VOLUME	Yes	Number
17	DOWN_VOLUME	Yes	Number
18	APN	No	String
19	CELL_ID	No	String
20	ROAMING_INDICATOR	Yes	Number
21	VISITED_COUNTRY_CODE	Yes	Number
22	LOCATION_AREA_CODE	No	String
23	IN_TRUNK_GROUP	No	String
24	IN_TRUNK_NUMBER	No	String
25	OUT_TRUNK_GROUP	No	String
26	OUT_TRUNK_NUMBER	No	String
27	SERVED_PDP_ADDRESS	No	String
28	MOBILE_SESSION_SERVICE	Yes	Number
29	MESSAGING_EVENT_SERVICE	Yes	Number
30	CALL_FORWARDING	No	Number
31	CALL_CONFERENCE	No	Number
32	CALL_HOLD	No	Number
33	VOICEMAIL	No	Number
34	VOICEMAIL_CALLBACK	No	Number
35	EMERGENCY_CALL	No	Number

Table 3.1: xDrs Attributes

Chapter 4

Clustering Process

4.1 General Description

The method implemented is a method of clustering data streams, based on the concept of micro-clusters. Micro-clusters are data structures which summarize a set of entities from the stream, and is composed of a set of statistics which are easily updated and allow fast analysis.

The method has two phases. In the online phase, a set of micro-clusters are kept in main memory; each instance coming from the input stream can then be either appended to an existing micro-cluster or created as a new micro-cluster. Space for the new micro-cluster is created by deleting a micro-cluster (by analyzing its last edit timestamp) using the fading function (subsection 2.3.2) described on Denstream. In the delete process we can either discard the micro-cluster or save the micro-custer, for further analysis, by its density. The offline phase will try to discovery patterns in the micro-cluster, to obtain the final density for the micro-cluster in the clustering process.

4.2 Implementation

ClusterProcess is the class that is responsible for reading xDRs instances from the data stream and is the class that control time. It is also the class where the fading process ocurrs. The number of micro-clusters parameter (m), controls how many micro-clusters are kept. Two more classes deal with the implementation of the clustering algorithm: MicroCluster and Clustering.

We can have several notions of time. The unit of time may vary according to our interest and the quantity and quality of the data. We can have the second, minute, or even hour as a unit of time. We can also say that the unit of time is the number of instances that reach the clustering process.

MicroCluster is the main data structure keeping the online micro-clusters updated. It has a last edit timestamp, a current timestamp and a density function to calculate the new density of an arrived instance to the micro-cluster. The density function go through all points in the micro-cluster to calculate the number of identified patterns in the micro-cluster for the arrived instance. Based on the identified pattern, sets a new density to the micro-cluster.

The clustering class is responsible to insert or delete micro-clusters from the cluster process. AutoExpandVector is a class that implements a vector with the capability of automatic expansion and is used by the Clustering class to keep the online buffer.

4.3 Experiments

The Kafka stream was installed in the server. Another machine served as a producer. It reads the dataset and sends it to the server. Another machine has the role of consumer, reading the xDRs instances that come from the producing machine through the server and doing the clustering process.

The dataset was provided by the WeDo Technologies company. It is a labeled dataset for which we can draw conclusions about the effectiveness of the method presented.

The dataset has 78124 records where there were events identified as prism fraud, smishing attempt fraud, risky destinations fraud, risky destinations fraud roaming, risky destinations cloning fraud, wangiri callback fraud, risky destinations imeisim swap fraud, trafic subscriber fraud, wangiri attempt fraud, prism fraud roaming and smishing callback fraud. 65848 records were considered normal instances.

```
Total Records: 78124
Stop Cluster...
#78123
#78124
Clustering Size: 1273
2020/02/02 13:16:30
2020/02/02 13:17:24

 : 65848
prism.fraud : 1240
smishing.attempt.fraud : 1581
risky.destinations.fraud : 310
risky.destinations.fraud.roaming : 310
risky.destinations.cloning.fraud : 1550
wangiri.callback.fraud : 1581
risky.destinations.imeisim.swap.fraud : 620
trafic.subscriber.fraud : 682
wangiri.attempt.fraud : 1581
prism.fraud.roaming : 1240
smishing.callback.fraud : 1581
```

Figure 4.1: Fraud events in dataset

The clustering process took about 30 seconds to execute and presented the following results.

```
Confusion matrix Total
 +-----------+------------+-----------+
 |           | Fraud      | No Fraud  |
 +-----------+------------+-----------+
 |Fraud      | 12106      | 0         |
 +-----------+------------+-----------+
 |No Fraud   | 170        | 65848     |
 +-----------+------------+-----------+

TPR(Sensitivity) = 0.9861518

TNR = 1.0

PPV (Precision) = 1.0

NPV = 0.99742496

prism.fraud : 1220/1240 : 98%
smishing.attempt.fraud : 1581/1581 : 100%
risky.destinations.fraud : 290/310 : 93%
risky.destinations.fraud.roaming : 285/310 : 91%
risky.destinations.cloning.fraud : 1460/1550 : 94%
wangiri.callback.fraud : 1581/1581 : 100%
risky.destinations.imeisim.swap.fraud : 605/620 : 97%
trafic.subscriber.fraud : 682/682 : 100%
wangiri.attempt.fraud : 1581/1581 : 100%
prism.fraud.roaming : 1240/1240 : 100%
smishing.callback.fraud : 1581/1581 : 100%
Cluster Stopped.
```

Figure 4.2: Clustering Process Results

In the rows, we have the forecasts in columns and the actual values. As you can notice we are dealing with unbalanced data. In real problems, when we set up the confusion matrix it is normal to find a greater number of false negatives than false positives.In our case we don't have false positives.

In the case of false positive and false negative problems, one way to work with unbalanced data is to increase the performance of predicting the observations with less quantity.

For an instance by itself to be considered fraud it has to be absorbed by a micro-cluster in which patterns have already been found in the data stream.

The decrease in observations would cause many micro-clusters to be forgotten more quickly, preventing other instances from being cataloged as fraud and the values on confusion matrix would certainly be different.

The implemented method has 98,6% of success rate, and the statistics are presented in figure 4.2.

To be able to get an online sense of the evolution of the method over time, a website was built that received information from the clustering process every second.

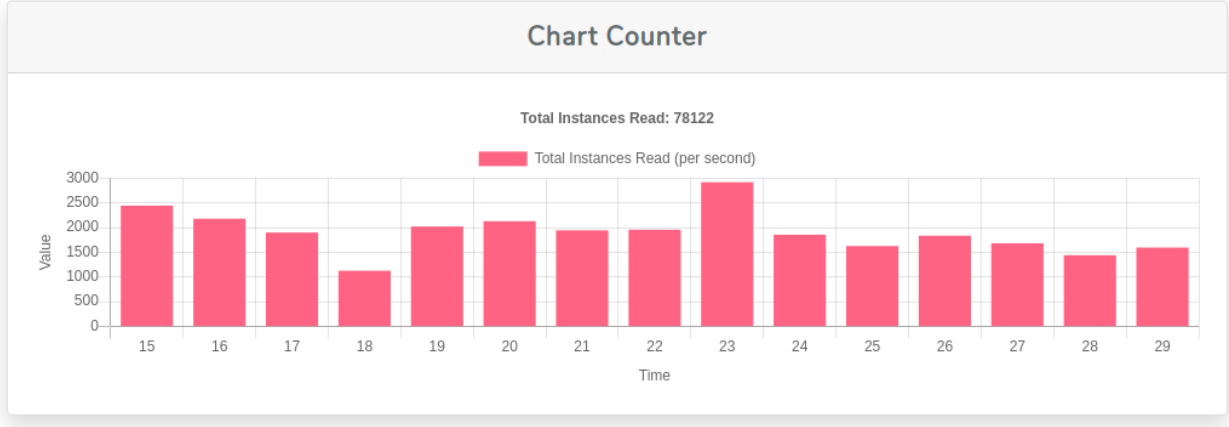

Figure 4.3: Instances read per second

Figure 4.3 shows that the algorithm can read and process 2500 instances per second. The values can be different depending on the processing capacity and the memory available for the clustering process. The processing capacity used in the experiments was nothing special and the results show the effectiveness of the method, even with a low level of processing.

Regarding the problem of keeping the instance in memory, we can notice in the chart of Figure 4.4 the performance of the forgetting function that discards lower weight micro-clusters, in order to allow other instances to enter the buffer.

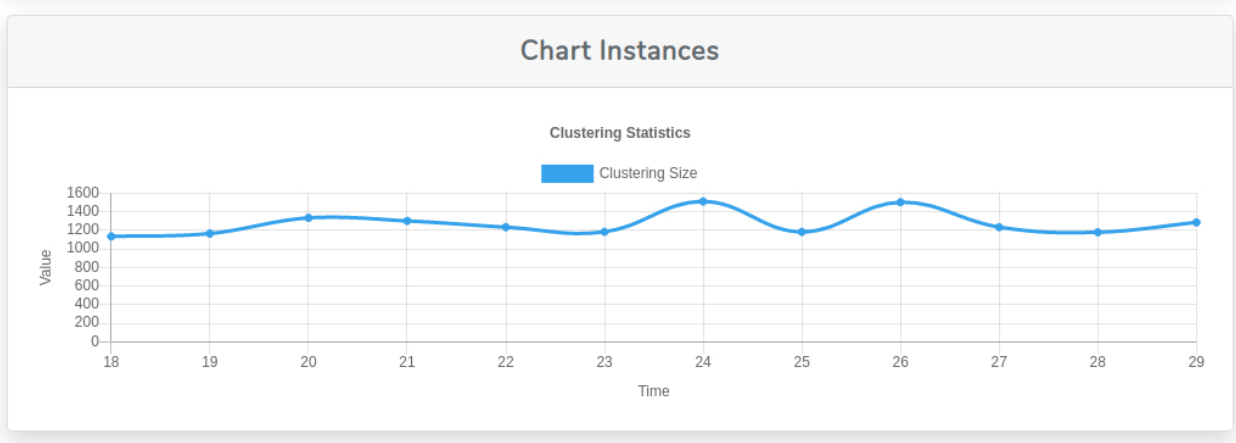

Figure 4.4: Micro-clusters in buffer

We can see in Figure 4.4 the evolution of the clustering process and the number of instances kept in the buffer at each instant of time. It is noticeable that the clustering process can keep the used memory level constant over time, this aspect being the most fundamental for clustering in data streams.

Chapter 5

Conclusion and Future Work

In order to build a clustering model for real-time fraud detection, it is extremely necessary to have a concrete knowledge of the type of fraud and the source of such fraud, so that the right attributes can be selected for the construction of our model. One model may be good for detecting one type of fraud, but it may not be enough to detect another type of fraud. For each type of fraud and each enabler technique we must have our own model adapted to each situation.

Since Apache Kafka is a distributed system it is possible to have, in real time, several models in operation at the same time. For future work, combining multiple fraud detection models can facilitate knowledge of user behavior and prevent the spread of a fraud scenario more quickly.

The final objective of this work will always be the implementation of an algorithm to deal with Novelty Detection, since changes in the data pattern of data streams may lead to faster discovery of new fraud situations. For future work, we intend to improve existing techniques, and lately find new techniques for detecting different novelties in data streams.

5.1 Novelty Detection

When new techniques of fraud are implemented, the discovery of these new techniques in data streams can take some time. During this time, telecom companies can have great losses. Novelty detection makes it possible to recognize novel concepts, which may indicate the appearance of a new concept. The method used and the implementation made gives a large margin for detection of new patterns on the data stream. The ability to launch multiple analysis instances over multiple attributes, allows attempts to detect multiple pattern types at the same time. By correlating these instances of analysis at a given point in time, we can easily come up with new patterns in the data stream that, once analyzed, could translate into a new kind of fraud. Through such an approach one can define a new pattern type in data stream. These patterns can be arithmetic or geometric patterns of the evolution of the numeric attribute. This search for these new patterns would initially be for entities for which a certain type of pattern had already been detected and, for the events of that entity, which has not yet detected any type of fraud, they would try to do an analysis to discover arithmetic or geometric patterns of the numeric attribute. If a new pattern was detected, a new analysis instance can be launched, with the aim of detecting that pattern in new instances from the data stream. There are a large number of possibilities for combining the various attributes of the xDRs instances, so there will be many possibilities for detecting arithmetic or geometric patterns on those possibilities.

Bibliography

[1] C. F. C. Association, "Telecom fraud loss survey." https://www.cfca.org/fraud-loss-survey.

[2] Q. Zhao, K. Chen, T. Li, Y. Yang, and X. Wang, "Detecting telecommunication fraud by understanding the contents of a call," *Cybersecurity 2018 1:8*, 2018. https://doi.org/10.1186/s42400-018-0008-5.

[3] A. Bifet, "Adaptive stream mining: Pattern learning and mining from evolving data streams," vol. 207, 2010.

[4] A. Bifet, R. Gavaldà, G. Holmes, and B. Pfahringer, *Machine Learning for Data Streams with Practical Examples in MOA*. MIT Press, 2018. `https://moa.cms.waikato.ac.nz/book/`.

[5] L. Rettig, M. Khayati, P. Cudré-Mauroux, and M. Piórkowski, "Online anomaly detection over big data streams," *IEEE International Conference on Big Data (Big Data)*, 2015.

[6] J. Gama, *Knowledge Discovery from Data Streams*. Chapman & Hall/CRC, 2010. `https://dl.acm.org/citation.cfm?id=1855075`.

[7] M. Chenaghlou, M. Moshtaghi, C. Leckie, and M. Salehi, "Online clustering for evolving data streams with online anomaly detection," *Advances in Knowledge Discovery and Data Mining*, 2018.

[8] D. R. Wilson and T. R. Martinez, "Improved heterogeneous distance functions," *Journal of Artificial Intelligence Research*, pp. 1–34, 1997.

[9] F. Cao, M. E. W. Qian, and A. Zhou, "Density-Based Clustering over an Evolving Data Stream with Noise," *In SAC '04: Proceedings of the 2004 ACM symposium on Applied computing, pages 632–636, New York, NY, USA*, 2006.

[10] R. A. A. Habeeba, F. Nasaruddina, A. Ganib, I. A. T. Hashemb, E. Ahmedc, and M. Imrand, "Real-time big data processing for anomaly detection: A survey," 2018. https://doi.org/10.1016/j.ijinfomgt.2018.08.006.

[11] J. Gama, P. Medas, and R. Rocha, "Forest trees for on-line data," *In SAC '04: Proceedings of the 2004 ACM symposium on Applied computing, pages 632–636, New York, NY, USA*, 2004.

[12] E. R. Faria, J. Gama, and A. C. P. L. F. Carvalho, "Novelty detection algorithm for data streams multi-class problems," *SAC '13 Proceedings of the 28th Annual ACM Symposium on Applied Computing Pages 795-800*, 2013.

[13] M. Ester, H.-P. Kriegel, J. Sander, and X. X. A, "density-based algorithm for discovering clusters in large spatial databases with noise," *In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD 96), Portland, Oregon, USA, pages 226–231*, 1996.

[14] Schulzrinne, Henning, State, Radu, Niccolini, and S. (Eds.), "Principles, Systems and Applications of IP Telecommunications. Services and Security for Next Generation Networks," *Second International Conference, IPTComm 2008, Heidelberg, Germany.Revised Selected Papers*, 2008.

[15] J. Gama, R. Sebastião, and P. P. Rodrigues, "Issues in evaluation of stream learning algorithms," *15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 329–337, 2009.

[16] K. Xie, X. Li, X. Wang, J. Cao, G. Xie, J. Wen, D. Zhang, and Z. Qin, "On-line anomaly detection with high accuracy," *IEEE/ACM TRANSACTIONS ON NETWORKING*, vol. 26, no. 3, 2018.

[17] Y. Mirsky, A. Shabtai, B. Shapira, Y. Elovici, and L. Rokach, "Anomaly detection for smartphone data streams," *Pervasive and Mobile Computing*, vol. 35, pp. 83–107, 2017.

[18] Yogita and D. Toshniwal, "A framework for outlier detection in evolving data streams by weighting attributes in clustering," *2nd International Conference on Communication, Computing & Security (ICCCS-2012)*, 2012.

[19] M. Toyoda, Y. Sakurai, and Y. Ishikawa, "Pattern discovery in data streams under the time warping distance," *The VLDB Journal*, vol. 22, pp. 295—-318, 2013.

[20] L. Chena, S. Gao, and X. Cao, "Research on real-time outlier detection over big data streams," *International Journal of Computers and Applications*, 2017.

[21] S. Microsystems, "Xdr: External data representation standard," *International Journal of Computers and Applications*, 1987.

[22] H. Sun, K. Deng, F. Meng, and J. Liu, "Matching stream patterns of various lengths and tolerances," *Proceedings of the 18th ACM conference on Information and knowledge management*, pp. 1477–1480.

[23] A. Bifet, G. Holmes, B. Pfahringer, and R. Gavaldà, "Mining frequent closed graphs on evolving data streams," *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, p. 591–599.

[24] C. C. Aggarwal, Y. Li, P. S. Yu, and R. Jin, "On dense pattern mining in graph streams," *Proceedings of the VLDB Endowment*, vol. 3 Issue 1-2, pp. 975–984, 2010.

[25] E. R. Faria, R. C. Barros, J. Gama, and A. Carvalho, "Improving the offline clustering stage of data stream algorithms in scenarios with variable number of clusters," *SAC '12: Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pp. 829–830, 2012.