A NOVEL APPROACH TO MOBILE MONEY SMISHING AND TRANSACTION FRAUD DETECTION AND PREDICTION USING DYNAMIC LOCALITY SENSITIVE HASHING

DJAM Xaveria Youh, TAPAMO Kenfack Hyppolyte Michel, Aminou Halidou, Atsa Roger Etoundi, Tekoh Palma Achu

ABSTRACT

Mobile money fraud has become so prevalent with users losing their hard earn income and Service Providers finding it difficult to cope with this new reality. Several authors in literature have proposed different approaches ranging from expert systems, machine learning and hybrid approaches. However, these approaches have not been able to handle the problem of efficiency and rapidity in mobile money fraud detection, while taking into consideration the imbalance nature of financial datasets and the dimensionality explosion problem associated with these datasets. This article proposes a novel dynamic Locality Sensitive Hashing approach to efficient and rapid detection and prediction of mobile money fraud related to SMS and Transaction data, based on the application of Search Based Software Engineering in machine learning while tackling the limitation brought about as a result of imbalance data inherent in financial datasets and the dimensionality explosion problem. A comparative analysis of our approach with the state-of-the-art techniques demonstrates promising results on both efficiency and rapidity. The results showed a 98.6% efficiency for SMS fraud data detection and prediction and 99% for transaction data while yielding an O(n) running time for both dataset where n is the dataset size. These results are very encouraging and will go a long way to reducing the amount of fraud on mobile money networks and hence an increase in customer trust and adoption of mobile money services.

***Index Terms**: Search Based Software Engineering, Dynamic Locality Sensitive Hashing, Mobile Money Fraud Detection and Prediction, Mobile Money SMS and Transaction*

***Reference to this paper should be made as follows:** DJAM Xaveria Youh, TAPAMO Kenfack Hyppolyte Michel, Aminou Halidou, Atsa Roger Etoundi, Tekoh Palma Achu, (2025), "A Novel Approach to Mobile Money Smishing and Transaction Fraud Detection and Prediction Using Dynamic Locality Sensitive Hashing", Int. J. of Electronics Engineering and Applications, Vol. 13, No. 1, pp. 55-78.*

Biographical notes:

***DJAM Xaveria Youh - KIMBI** is currently a Senior Lecturer in Department of Computer Science, Faculty of Science, University of Yaounde I, Cameroon. She holds a Ph.D. in Computer Science from Modibbo Adama University of Technology, (MAUTECH) Yola which she obtained in 2011. Her research interests include Search-based Software Engineering, Information System Modelling, Software Testing and Cloud Computing.*

***Hippolyte Michel Tapamo Kenfack** is a professor in the Department of Computer Science of the Faculty of Sciences of the University of Yaoundé I, Cameroon. He obtained his Ph.D. in Computer Science from University of Yaoundé I, Cameroon. He is a specialist in computer vision and artificial intelligence. He is interested in medical imaging and image analysis for the prevention of natural disaster risks such as floods and landslides.*

***Aminou Halidou** is a computer science expert born in Cameroon in 1979. He holds a Ph.D. in Computer Science from Huazhong University of Science and Technology (HUST) in Wuhan, China, which he obtained in 2014. His research expertise lies in computer vision and image processing. Since 2014, Aminou has been a lecturer at the University of Yaoundé 1, where he currently serves as the Head of the Computer Science Department.*

***Roger Atsa Etoundi** is a distinguished academic affiliated with Université de Yaoundé I in Cameroon, where he has contributed significantly to the field of information systems and technology since 2005. His research primarily focuses on the intersection of information and communication technology (ICT) and public administration, particularly within the context of developing countries. Etoundi's recent work addresses pressing contemporary issues, such as the impact of ICT in managing road traffic control during the*

COVID-19 pandemic, showcasing his commitment to applying theoretical frameworks to real-world challenges. His publications reflect a deep engagement with topics like e-government, process optimization, and the development of the digital economy in Cameroon.

***Tekoh Palma Achu** is a professional with a strong background in computer science and software engineering. He is currently serving as the Coordinator of Graduate Programs and a Lecturer at The ICT University Yaounde. His expertise includes designing, implementing, and maintaining software for large-scale distributed systems, as well as building AI systems to solve business problems.His area of research includes business process modeling, E-learning.*

I. INTRODUCTION

The world is experiencing a sweeping change in digitalization, affecting all sectors, prominent of them being the financial sector and most especially mobile money services. Mobile Money is a universal term that is centered around the use of Mobile phones and devices to carry out financial transactions. With over 1.35billion registered accounts processing $1 trillion annual transactions, roughly $2million worth of transaction per minute, and over $697.7Billion worth transactions effectuated in sub-Sahara Africa (Debray, [1]). Mobile money has drastically changed the way we view and carry out financial transactions, becoming the de facto payment mechanism of strategic and societal importance to the Unbanked and Underbanked population, most especially in developing countries (Adedoyin [2]). The swift growth of mobile money presents evidence that the telecommunications service is now a multidisciplinary one, embracing finance and commerce (Adedoyin [3]). It's rapid growth and adoption rates are due to the decrease in price of mobile phones, the ease of performing transactions and the ability to perform small value payments with great speed and lack of cumbersome financial procedures needed to perform a transaction and also the much-reduced fees charged per transaction.

Notwithstanding, mobile money faces challenges which can be viewed as being both operational and technological (Adedoyin, [3]). Operational challenges are related to regulatory frameworks, controls on cash in and cash out operations, customer identification process and other activities centered around the working of the mobile money service while technological challenges look at the challenges that can be resolved with technological tools and techniques. Our research focuses on the technological challenges most especially mobile money fraud.

Mobile money fraud is the illegal use of mobile money service to make financial gains or conceal illegal financial dealings. It mostly involves fraudsters taking advantage of the kindness and naivety of mobile money service users. Our research identifies the following fraud related problems faced by mobile money users.

The aim of this research article is to use a Novel Dynamic Locality Sensitive Hashing (DLSH) technique to efficiently and rapidly detect and equally predict mobile money fraud related to Smishing and Mobile Money transactions.

The main contributions of this research paper are in two folds:

1. Use a Novel Dynamic Locality Sensitive Hashing techniques to efficiently and rapidly detect and predict mobile money SMS fraud.
2. Use a Novel Dynamic Locality Sensitive Hashing technique to efficiently and rapidly detect and predict mobile money transaction fraud.

Problem Analysis

More and more fraudsters are gaining access to customers account by using social engineering methods such as Smishing in which a user is sent a message that trick them into revealing their secret pin code or entice them to perform a fraudulent transaction.

Also, mobile money users fall victim to fake support calls where the fraudster pretends to be a service agent or in collaboration with a service agent, a victim is called and is convinced to reveal their pin which can then be used to gain access to their mobile money account.

Once the fraudsters gain access to the account using any of the aforementioned techniques, they will proceed to empty the victim's account by either transferring money to mule accounts (that will subsequently cash out the profit) or directly using a merchant to cash out the maximum allowed. When the balance of the victim exceeds the maximum allowed, several mule accounts are used for collecting the money(Alonso et., [4]).

Becoming more prevalent is the fraud involving fraudster sending a smishing message that resembles a legitimate cash in receipt notification followed with a call in which the fraudster claims to have mistakenly send them funds and pleads that they send it back. In the process of trying to check their account balance, the fraudster initiates a withdrawal from the victim's account and the victim ends up validating the request unknowingly.

Moreover, mobile money users do misplace their mobile devices and it ends in the hands of a fraudster who would try to perform illegal transactions on the mobile money account.

Furthermore, mobile money enables users to send money to other users without a mobile money account and through the use of vouchers codes. Due to the asynchronous nature of voucher codes, they are highly vulnerable to fraud especially

when the merchant is involved (Alonso et al., [5]). The asynchronous nature leaves a window of opportunity for a third party to claim the money before the rightful owner does. The merchant can also create and already used voucher and send to the victim while keeping the money received from the sender and while at cash out the victim will realize the voucher was an already used one (Alonso et al., [5]).
When faced with all of these challenges, mobile money users are constantly being robbed of their hard-earned income and lose confidence in the service and hence a great lost to mobile money service providers and a setback to the growth of Mobile money as a whole.
In this research we focus on:

- Smishing Fraud in which Fraudulent Mobile Money Messages are sent to trick an innocent Mobile Money user to perform fraudulent transactions or reveal Pin code.
- Transaction fraud that arises when users lose their phones which ends up in the hands of fraudsters who end up effectuating fraudulent transactions

They arise the need for fraud detection and prediction systems which can mitigate against these fraudulent activities, but most mobile Money services use an arbitrary threshold assignment mechanism to combat mobile money fraud (Alonso et al., [5]).
Finally, most existing state of the art fraud detection and prediction systems are not flexible and very costly to maintain and update (Rieke et al., [13]), some act as black boxes which makes it difficult to trace the rationale used to make predictions (Botchey [8]), while others are very expensive and does not scale well on massive amounts of dataset (Adedoyin [3]).

Mathematical Analysis of Problem

Consider the real-world example of efficiently detecting fraud by the following mathematical model
Let Urepresent the universal set of all **m** Mobile Money users for a given service provider,

$$U = \{u_1, u_{2,} \cdots, u_m\}.$$

These users constantly perform a series of mobile money transactions on the mobile money system represented by T such that $T = \{T_1, T_2, \dots, T_k\}$ where each transaction T_k is made up of u attributes $T_i = \{T_k^1, T_k^2, \dots, T_k^u\}$ and M represent the set of j mobile money messages $M = \{M_1, M_2 \dots \cdot M_j\}$ with each message M_j made up of v attributes $M_i = \{M_j^1, M_j^2, \dots, M_j^v\}$

The communication C_{STMR} between these users $u_x, u_y \epsilon U$. can be modelled as follows (see Figure-1)

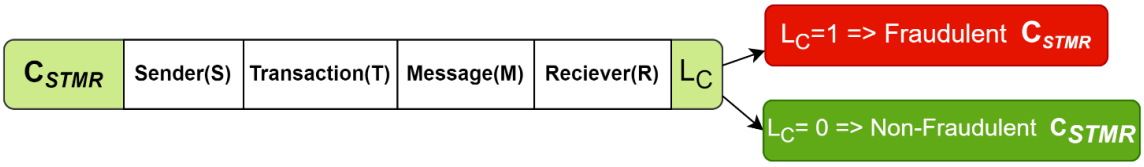

Figure-Error! No text of specified style in document.**: Representation of a Communication in a Mobile Money Network**

Where:

C_{STMR} = Model of communication on a mobile money network
C_i = i^{th} communication on a mobile money network
S = Sender or initiator of C_{STMR}
T = Set of i Mobile money Transaction with u attributes each
M = Set of j Mobile money message with v attributes each
R = Receiver or receptor of C_{STMR}
LC = Label of the Communication which can be fraudulent or non-fraudulent
$S, R \in U$

It is worth noting that in T they exist T_F and T_{NF} where T_F are fraudulent transactions and T_{NF} are non-fraudulent transactions. Also, in M they exist M_F and M_{NF} where M_F are fraudulent messages and M_{NF} are non-fraudulent messages. It is always the case that $T_{NF} \gg T_F$ and $M_{NF} \gg M_F$

Let L_C be the label for C_{STMR} such that $L_C \in \{0,1\}$, $L_C = \begin{cases} 1, \wedge\ if\ c_{STMR}\ is\ fraudulent \\ 0, \wedge\ otherwise \end{cases}$

They exist C_{STMR} where ***T*** =***Null*** or ***M*** =***Null*** but $T \neq Null$ and $M \neq Null$
For a given communication instance C_i if $T_i \neq Null$ then $\exists M_i \in M \forall C_i \in C_{STMR}$
Let's consider the mobile money Communication tree given below (see Figure-2)

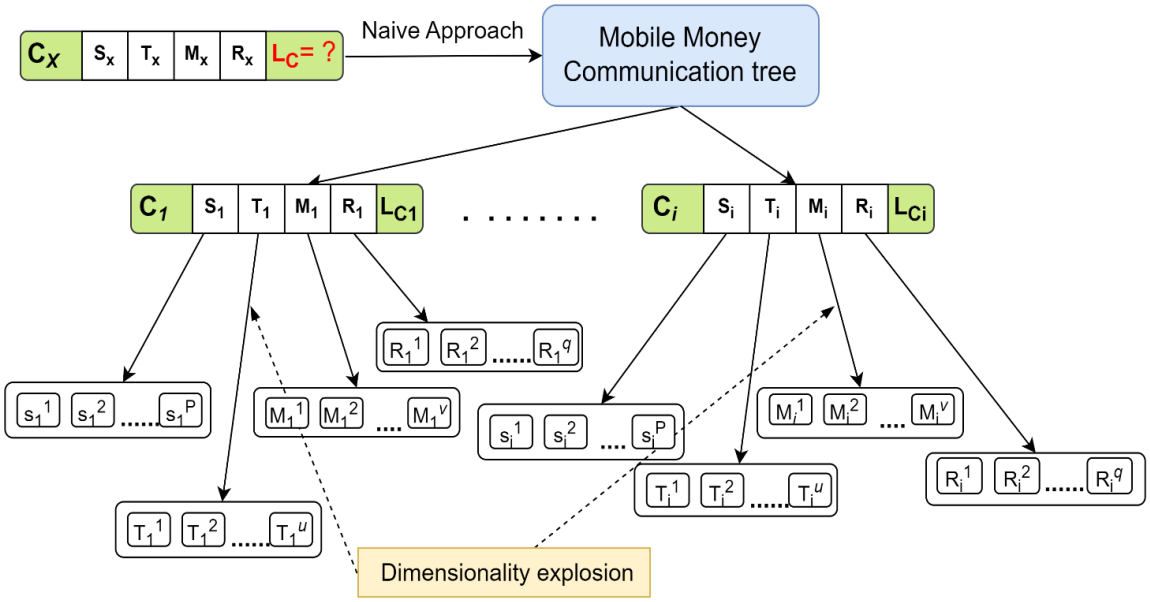

Figure-2: Dimensionality Explosion in Mobile Money Communication Tree

Given a search space of 100,000 possible candidate communications which is small by mobile money daily communication volume, and a given communication instance $C_{x,}$ we would need to perform. $\frac{100000\times99999}{2} \approx 5000{,}000{,}000$ pairwise comparison needed in order to identify communications that are similar to $C_{x,}$.

if we take a computer that performs 10,000 comparisons per second, we would need approximately 5.787**days** to finish the computation.

This computational cost is further aggravated with the explosion of dimensionality shown in Figure 1.2 which arises when handling high dimensional data (data with multiple features) such as Mobile money SMS and Transaction data. Considering a dataset of five (5) dimensions the computation now takes **28.9 days** to run to completion.

Hence, we look at how we can speed up the task of finding similar to test messages M_x and test transactions T_x by apply DLSH and hence determining L_C for a given C_{STMR} such that

$$L_C = DLSH(C_{STMR}) \begin{cases} 1,\wedge\, if\,(T_x \in T_F) \vee (M_x \in M_F) \\ 0,\wedge\, otherwise \end{cases}$$

Based on the problems identified and depicted on the illustrative example (Figure–2) in order to minimize mobile money fraud, we posed the following as main research question (RQ).

RQ: How can a novel dynamic locality sensitive hashing approach be used to efficiently and rapidly detect and predict mobile money fraud related to SMS and transactions?

Our main research question is further divided into sub research questions

RQ1: How can we use a novel dynamic locality sensitive hashing technique to efficiently and rapidly detect and predict mobile money SMS fraud?

This will help curb mobile money fraud by filtering out the messages before it reaches the intended victims.

RQ2: How can we use a novel dynamic Locality sensitive hashing technique to efficiently and rapidly detect and predict mobile money fraud related to transactions?

Providing answers for this will help reduce the ability for fraudsters to successfully perform fraudulent transactions without being predicted or detected.

The above questions will serve as our sailor's compass and will guide us as we sail through this sea of research trying to design, implement and test our proposed approach. The answers to the above questions will serve as a validation to our methodology.

The remaining part of this article is structured as follows; section 2 gives a comprehensive look at the mobile money ecosystem and also some background concepts on locality sensitive hashing, a review of literature on mobile money fraud detection, research into solving the nonexistence of real-world mobile money transaction data with the use of synthetic data and finally review on solving the problem of imbalance data inherent in financial data. Section 3 presents a detail step by step procedure involved in our methodology with an architecture of our system and a mathematical model to depict the functioning of our system. Section 4 represents the analysis, results obtained and a discussion of the results and their implications. Section 5 gives a holistic summary of our findings, recommendation for further research and a conclusion of our article.

2. Literature Review

2.1. The Mobile Money Eco-system

2.1.1 The Concept of Mobile Money

Mobile Money is a universal term that is centered around the use of Mobile phones in order to carry out financial dealings. It all started in Kenya in 2007 when Safaricom launched its M-PESA solution for peer-to-peer money transfer as a means of allowing people to send money to their relatives from their mobile phones. This service quickly spread through the world and Africa in particular, serving the unbanked and underbanked populations. 15years on, the service has recorded 316 deployments in 98 countries with over 1.35billion registered accounts, processing $1 trillion annual transactions (Awanis et al., [6]).

According to (Chadha et al., [10]), the Mobile Money ecosystem can be viewed as being made of three (3) services as shown in (Figure-3).

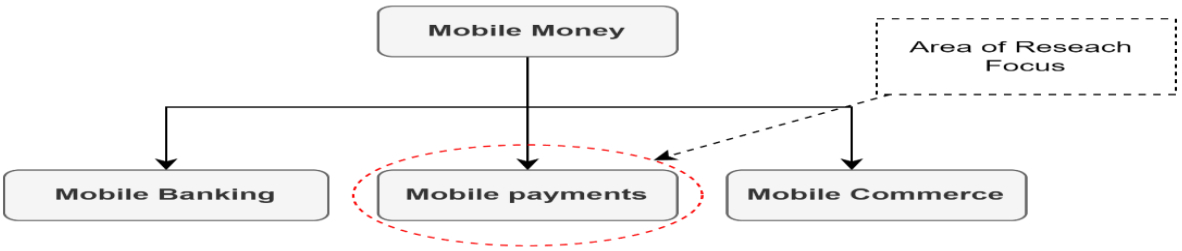

Figure-3: Mobile Money Services, adapted from [10]

Mobile Banking: using a mobile device, users can access their bank and carry out transactions such as deposit, withdrawal and payment of bills.

Mobile Payments: Often referred to as mobile money transfer (MMT), these are financial payments made using mobile devices and are conducted under a regulatory framework. They may include person-to-person transfer, payment of bills, remittances etc.

Mobile Commerce: The use of mobile devices as a means of buy and selling of goods and services on and off-site.

2.1.2 Organization of Mobile Money Ecosystem.

The Mobile money eco-system can therefore be viewed as a community of individuals, businesses and regulatory authorities who constantly interact with each other in order to provide innovative mobile money solutions to its customers who are also part of the ecosystem. These key actors are the Consumers (End-users and Service Providers), Distribution Channels (retailer and Wholesaler), Mobile Money network Operator (MNO), Commercial Banks and Central Bank. (Figure-4) depicts the mobile money ecosystem along with the interaction amongst the various actors.

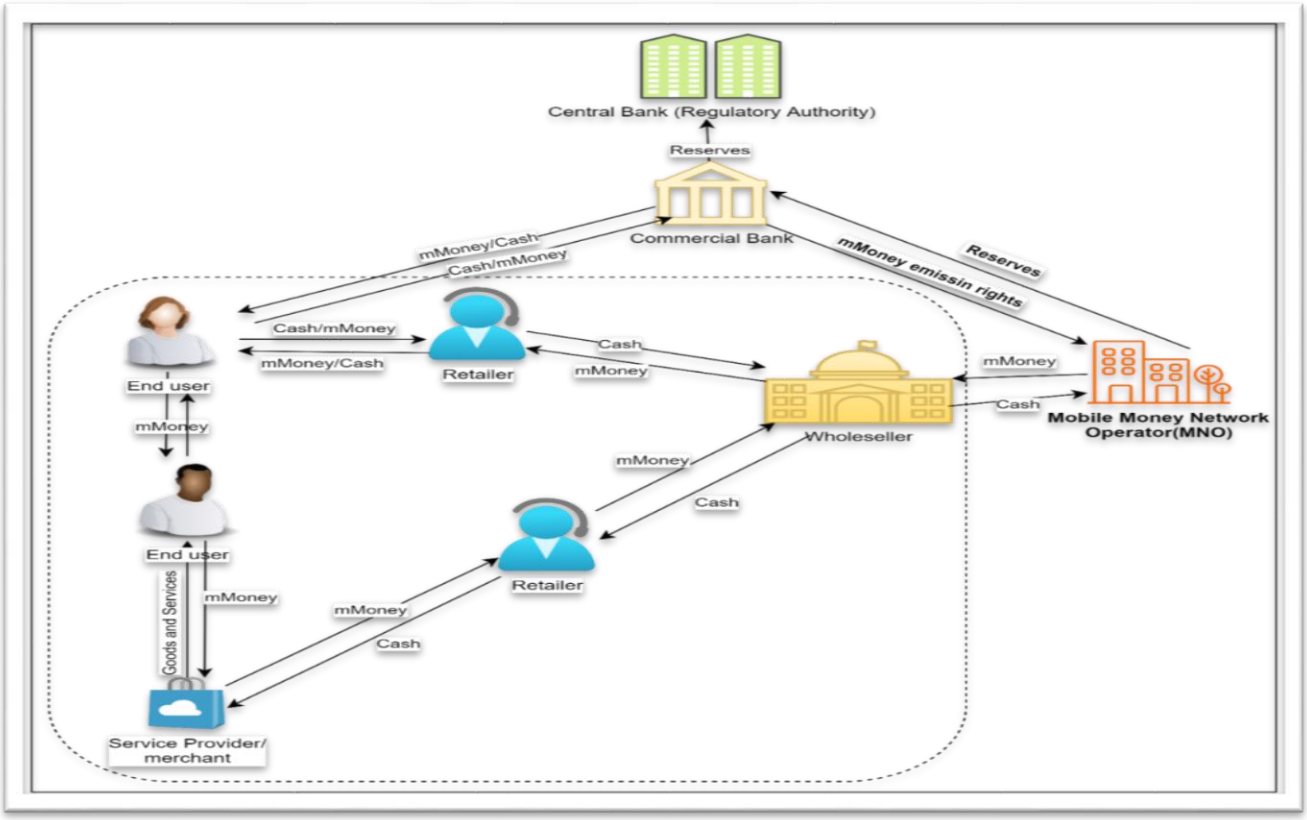

Figure-4: Mobile Money Transfer Ecosystem adapted from [11]

2.2 Background Concepts

- Mobile Money: This is the use of Mobile devices to carry out financial transactions.
- Mobile Money Fraud. This is the use of Mobile money services to carry out illegal financial operations
- Search Based Software Engineering (SBSE) was coined by M. Harman [15], it is a branch of software engineering that seek to apply search-based techniques mostly meta heuristic techniques (Genetic algorithms, simulated annealing, genetic programming, etc.) in resolving difficult software engineering optimization problems that involve competing constraints.
- Meta Heuristic is any algorithm that makes use of prior knowledge of the problem to reduce the search space (preventing exhaustive search) and hence does not guarantee an optimal solution but rather a near optimal solution in a reasonable amount of time. They are of 2 main categories: Population based and single point meta heuristic, (Harman et al., [16]).
- Machine learning it's a branch of computer science that enables us to make computers to be able to perform task without having to explicitly program them. It's divided into 2 broad categories supervised and unsupervised learning
- Hashing: is a widely used computational technique of mapping objects from a larger dimensional space to a much smaller dimensional space. They are very efficient in improving many search and lookup algorithms (Wiem, et al., [17]).
- Semantic encoding refers to transforming the meaning of word, text, sentences or other objects into a representation that can be understood and processed by the computer.
- Feature engineering: When carrying out predictions in machine learning, it is not always the case that all features in our dataset play a positive role in our final predictions. It is therefore necessary to identify and use just those features that play a positive contribution to our prediction.

2.3 Growth and Adoption of Mobile Money Services

As of 2021, more than 1.5 million person-to-person (P2P) transactions were made every hour on average, compared to 68000 in 2012, with an account making on average 3.5 p2p transactions per month. The tables below from State of the industry report on Mobile money for 2022 (Awanis et al., 2021) show the global growth per region (Table-1) and Africa regional growth (Table-2) of mobile money usage, while (Table-3) gives a 2016-2020 BEAC report of mobile money usage in Cameroon, (Awanis et al., [6]).

Table-1: Global mobile money growth per region for 2021

	Live Services	**Registered Accounts**	**Active Accounts**	**Transaction Volume**	**Transaction Value (US$)**
Sub-Saharan Africa	161	605million	183million	36.6billion	697.7billion
East Asia and Pacific	52	328million	64million	6.9billion	141.9billion
South Asia	34	283million	70million	8.9billion	156.3billion
Latin America and the Caribbean	32	49 million	20million	970 million	30billion
Middle East and North Africa	28	59million	5million	242million	13.7billion
Europe and Central Asia	9	22million	5million	294million	6.3billlion
Global Total	***316***	***1.35billion***	***346million***	***53.9billion***	***1 trillion***

Source: GSMA

Table-2: Mobile Money growth in Africa per region for 2022

	Live Services	**Registered Accounts**	**Active Accounts**	**Transaction Volume**	**Transaction Value (US$)**
West Africa	69	237million	58million	9.3billion	239.3billion
North Africa	12	15million	1million	77million	3.7billion
Central Africa	19	60million	19million	2.9billion	50.1billion

Southern Africa	14	13 million	4million	335 million	4.9billion
Eastern Africa	59	296million	102million	24billion	403.4billion

Source: GSMA

Table-3: Mobile Money Usage in Cameroon from 2016 -2020 [7] (BEAC)

	2016	***2017***	***2018***	***2019***	***2020***
Number	49 831 982	210 276 929	415 024 972	615 357 306	806 055 732
Value (XAF)	887 783 935 214	3 412970 418 636	6 333 005 588 739	8 812 226 321 166	12 151 073 705 383
Evolution in amount of electronic money	23 093 868 867	62 416 135 281	93 738 765 071	120 323 278 440	162 194 435 683
Registered Accounts	5 452 730	8 003 252	9 244 064	15 851 849	19 528 723
Active Accounts	1 615 404	3 230 236	4 979 736	7 982 674	8 453 605

Source: BEAC [7]

2.1.1. Regulatory Framework

The regulatory framework of mobile money is still under development as governments and central banks try to catch up with the fast pace of this ecosystem. In 2018 GSMA introduced the Mobile money Regulatory index (MMRI) which is a framework for measuring factors that promote a successful mobile money service. It is based on six enabling factors; **authorization**, **consumer protection**, **transaction limits**, **KYC (Know your customer), agent networks**, **investment** and **Infrastructural environment**.(Awanis et al., [6]) In Cameroon, one of the CEMAC countries, Mobile money is regulated by BEAC. BEAC reviewed the payment regulations in December 2018 to allow non-banks to offer mobile money services directly. The previous regulations required MNOs to partner with financial institutions in order to provide mobile money services. With a National financial inclusion strategy in place, Cameroon is well ahead of its central African sub-regional peers, with a **MMRI** Score of **82.20** (Awanis et al., [6]).

2.4 Mobile Money Fraud Detection and Prediction Techniques

With the Status quo, they arise a need for fraud detection systems that should not only be efficient in predicting these frauds before it affects the users but also one that is highly scalable to meet up with the transaction volume of the ever-growing mobile money ecosystem. Various stress tests need to be conducted to verify the system's ability to withstand sudden changes in volume, its performance under harsh conditions and susceptibility to changes in fraud detection parameters.(Lundin, Kvarnström, and Jonsson, n.d.). Several techniques have been proposed in literature for mobile money fraud detection and prediction (Leskovec et al., [12], Rieke et al., [13], Wiem et al., [17])
There is however, a fundamental flaw in their techniques in that it partially or does not tackle the problem of efficiency and speed in mobile money fraud detection and prediction while taking into consideration the imbalance nature of financial datasets. Furthermore, the classical LSH suffer from premature convergence (stopping criterion), which significantly affects its performance therefore, we have proposed a novel dynamic Locality Sensitive Hashing technique to cope with the weaknesses of classical LSH.

3. Novel Dynamic Locality Sensitive Hashing Methodology

The methodology proposed in this article is an integrated three-stage Novel Dynamic Locality Sensitive Hashing based on Search Based Software Engineering (Figure-5).

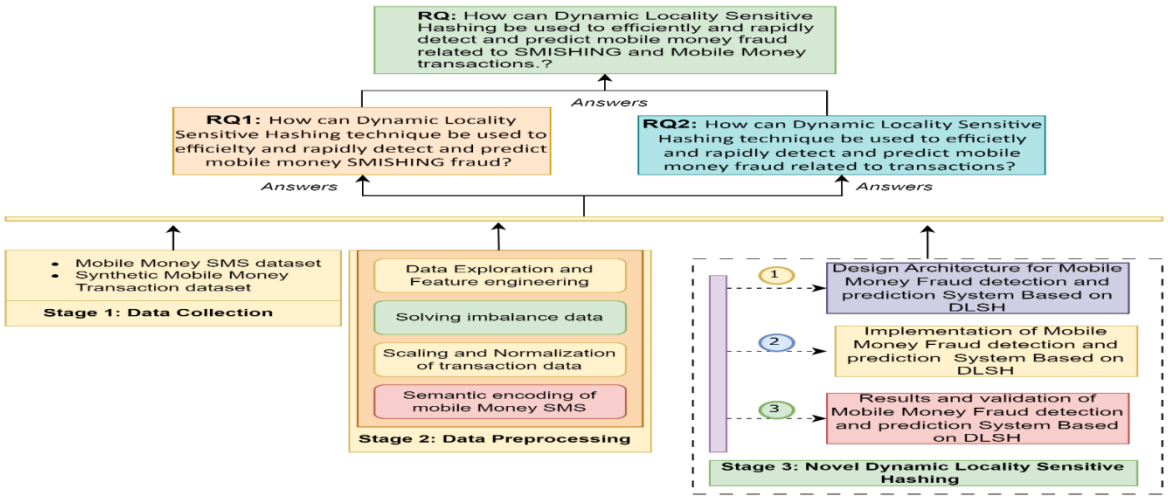

Figure-5: Overview of Research Methodology

Our research methodology can be summarized by the steps shown in Figure-5 above, we start off by:

3.1. Data Collection

We need to obtain reliable Mobile Money datasets. We identified two categories of dataset for our research. Mobile money user's SMS and Transaction dataset. Due to the inherent privacy, security and secretive nature of financial transactions, we were unable to obtain real world dataset. To resolve this challenge, we used synthetic data from Paysim simulator as proposed by Alonso et al., [5]. This data was generated using a real-world data obtained from a mobile money operator operating in over 14 African countries. Though synthetic in nature, it presents the following advantages of not disclosing users' information, it is less costly, time saving, highly scalable, readily available, maintains privacy of service users and enables researchers to test new fraud cases in a controlled environment, including those that have not yet been recorded in the real dataset while preserving the quality and statistical properties of the real-world dataset used.
For mobile Money SMS dataset we obtained real world SMS spam dataset from (Tiago and Almeida [14]) augmented with 100 Mobile Money Spam SMS gotten from colleagues, friends and family.

3.2. Data Preprocessing

3.2.1. Data Exploration

We carried out data exploration to have a better understanding and insights into how the data is represented and also to identify and eliminate some potential null values and data anomalies. The transaction dataset obtained from (Alonso [4]) is made up of the following schema (Table-4). Our SMS data set is made just two rows V1 and V2 which we renamed to **Type** and **Message** respectively (Table-5)

Table-4: Mobile Money Transaction Dataset Schema

Attribute	Description	Datatype
Step	maps a unit of time in the real world. In this case 1 step is 1 hour of time. Total steps 744 (30 days simulation)	*Int64*
Type	CASH-IN, CASH-OUT, DEBIT, PAYMENT and TRANSFER. This are the different possible types of transaction that is capture by our dataset.	*Object*
amount	amount of the transaction in local currency	*Float64*
nameOrig	customer who started the transaction	*Object*
oldbalanceOrg	initial balance before the transaction	*Float64*
newbalanceOrig	new balance after the transaction	*Float64*
nameDest	customer who is the recipient of the transaction	*Object*
oldbalanceDest	initial balance recipient before the transaction. there is not information for customers that start with M (Merchants).	*Float64*
newbalanceDest	new balance recipient after the transaction. Note that there is not information for customers that start with M (Merchants).	*Float64*
isFraud	This is the transactions made by the fraudulent agents inside the simulation	*Int64*
isFlaggedFraud	The business model aims to control massive transfers from one account to another and flags illegal attempts. An illegal attempt in this dataset is an attempt to transfer more than 200.000 in a single transaction.	*Int64*

Table-5: Mobile Money Message Dataset Schema

Attribute	Description	Datatype
Type (VI)	Gives information about the type i.e., Ham or Spam	TEXT
Message (V2)	The actual text message	TEXT

3.2.2. Feature Engineering

Not all attributes of our dataset play an important role toward making predictions, so we carry out feature engineering techniques on our transaction data to identify relevant features. While analyzing the various features we realize the type attribute is a string object so we perform one-hot encoding to convert it to numeric representation. As feature engineering techniques, we deployed Sklearn's ExtraTreesClassifier (see Table-6) feature importance measure and Chi Square score is performed on the transaction dataset and the following results were obtained (Figure-6).

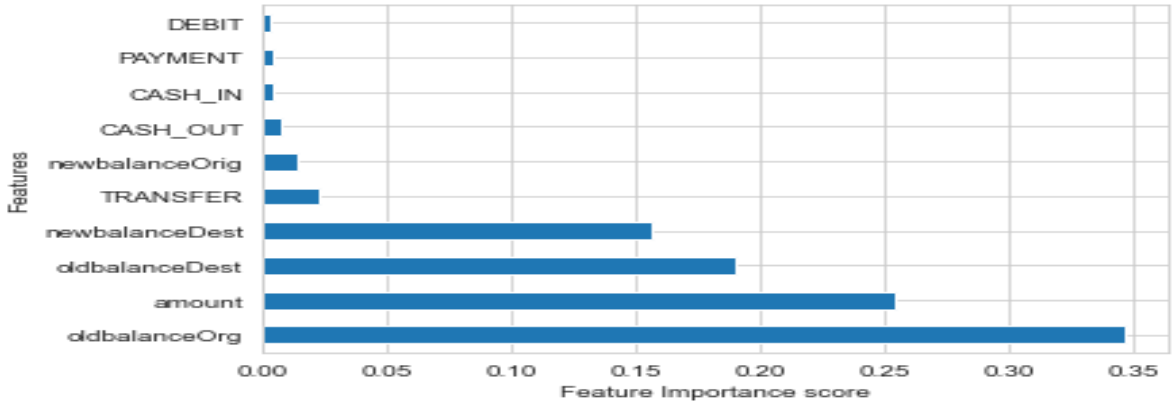

Figure-6: Feature Importance Score of Tree Base Classifier

Table-6: Sklearn Chi2 Feature Scores

Features	***Score***	***p-values***
DEBIT	3.768506e-01	0.539294
CASH_IN	1.375505e+00	0.240868
PAYMENT	2.967699e+00	0.084943
CASH_OUT	3.605765e+00	0.057580
TRANSFER	7.310085e+00	0.006857
Amount	5.566265e+06	0.000000
oldbalanceOrg	1.991225e+06	0.000000
newbalanceOrig	6.265222e+06	0.000000
oldbalanceDest	4.499363e+06	0.000000
newbalanceDest	1.385644e+06	0.000000

We eliminated all the features with a p-value >0.05 since they have no significant contribution to our prediction. This leads to a reduction in our dataset from a $R^{6353307\times11}$to a $R^{532021\times6}$vector. And we are left with the following features, ***TRANSFER, Amount, oldbalanceOrg, newbalanceOrig, oldbalanceDest, newbalanceDest*.**
As for our SMS dataset, we have just two features: the type and the message, so there is no need for feature selection.

3.2.3. Solving the Problem of Imbalance Data

With a total of 532021 transaction instances only 3884 are fraudulent representing 0.73% of the entire dataset. This clearly shows that our dataset suffers from the problem of imbalance data. As for the SMS dataset it is made up of 5650 messages with 847 being spam and representing 15% though being imbalance it acceptable and is not highly skewed like the transaction dataset.To solve the problem of imbalance transaction data we used a data level hybrid approach by performing over sampling using synthetic minority over sampling techniques (SMOTE) which introduces noise and some outlier, we then applied Tomek technique to eliminate the noise and outliers, Resulting in a more balanced dataset.

3.2.4.Scaling and Normalization of Transaction Data

From Observation it becomes evident that the range of values of our features in our transaction is very large and this results in our DLSH taking much more time to converge to a solution. In order to resolve this, we perform feature scaling in which we made the attributes of our transaction data to take on a similar range of values and by so doing we speed up the training and query time our DLSH takes (decrease the time it takes for DLSH to converge to its final solution). Ideally, we want each value of our six features[TRANSFER(U), Amount(V), oldbalanceOrg(W), newbalanceOrig(X), oldbalanceDest(Y), newbalanceDest(Z)] value to fall in the range $0 \leq U, V, W, X, Y, Z \leq 1$.
We adopted Min-Max Normalization technique to scale our transaction data which is evaluated as follows:

$$x_i' = \frac{x_i - \mu(X)}{max(X) - min(X)} \quad \textbf{(3.1)}$$

Where

x_i' is the scaled value of x_i
$\mu(X)$ is the mean value of all values in feature X
$max(X), min(X)$ are the maximum and minimum values of feature X

3.2.5. Semantic Encoding

Since computer systems work only with numbers, we need a mechanism for transforming our Mobile Money SMS messages into numeric representation which captures the semantic meaning of the messages. This is known as semantic encoding, in order to do this encoding, we adopted the Universal sentence encoder approach proposed by Daniel Cer *et al* (Cer et al., 2018) at Google, this model was used due to its accuracy, performance and it supports transfer learning to other Natural Language Processing (NLP) task and most especially its ability to capture contextual meanings compared to word based embedding techniques. This embedding scheme results in a 512-dimensional vector embedding for each message.

$$M_i = [\overrightarrow{M_1}, \overrightarrow{M_2}, \dots, \overrightarrow{M_n}] \quad (3.2)$$

$$M_E = Google_{USE}(M_i) \quad (3.3)$$

Where:

M_i $isthesetofnmobilemoneymessagesformobilemoneyusers$ represented as vectors

$M_E isthesemanticencodingforM_i$

$M_i \in R^{n\times\infty}$ and $M_E \in R^{n\times 512}$

3.3. Dynamic Locality Sensitive Hashing Methodology

This section describes the series of steps needed to implement our novel Dynamic Locality Sensitive Hashing (DLSH) in order to solve the problem of mobile money fraud detection and prediction which is elaborated by the System architectural diagrams (see Figure-7) which depicts the training phased architexcture and (see Figure-8) which depicts the testing phase architecture.

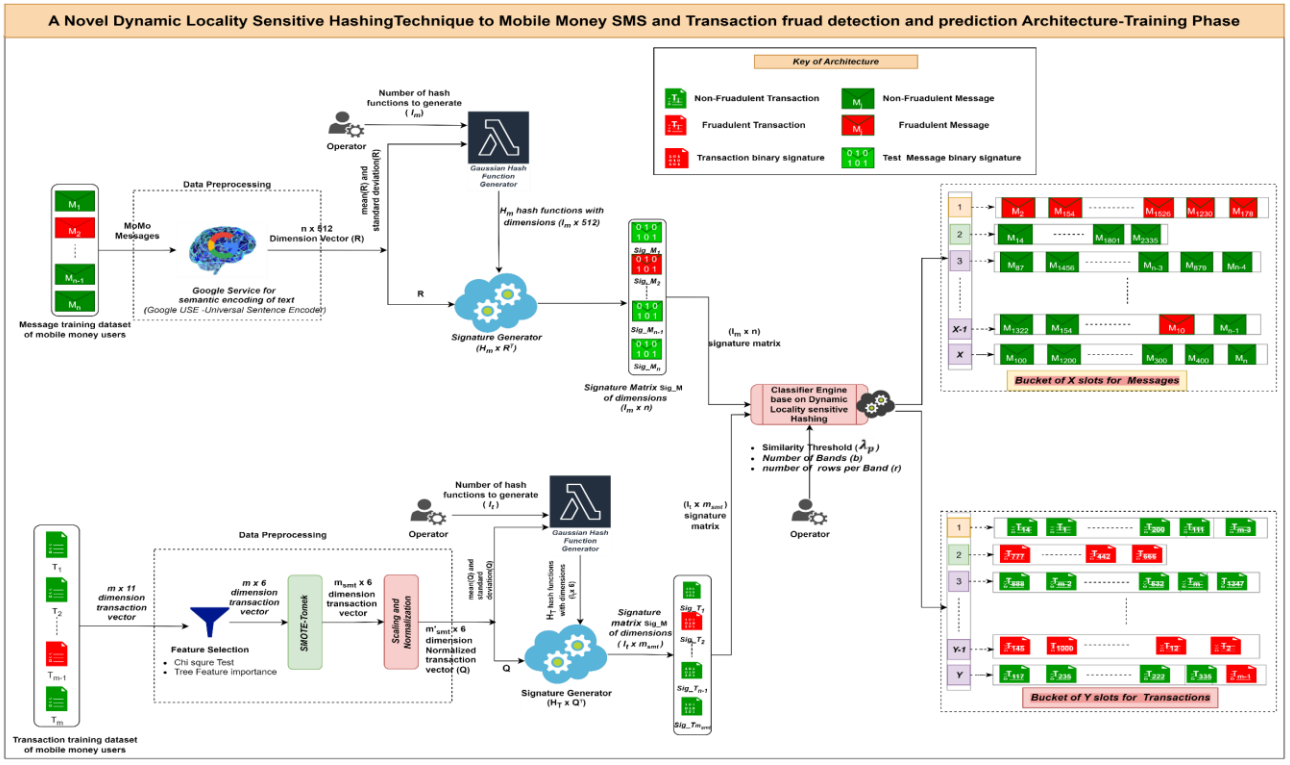

Figure-7: System Architectural View –Training phase

3.3.1. Signature Generation

We start off by constructing signature representation for each message/transaction, in order to do this, we define the notion of a universal family of hash functions H_m , $H_T \in H$where H_m and H_T is the universal family of hash functions for messages and Transactions respectively.

Critical to the signature generation process is the scalar projection (i.e., dot product) operation which we define on the family of hash functions as follows

$$h(\vec{d}) = \vec{d} \cdot \vec{v} \quad (3.4)$$

Where $h \in H$, $\vec{d}$ is the vector representing our data point and $\vec{v}$ is a vector whose dimensions are randomly selected from the gaussian distribution $V \sim N(\mu_D, \sigma_D^2)$. μ_D and σ_D are the mean and standard deviation of the universal set of data points $\vec{D}$ respectively. $\vec{d} \in \vec{D}$ Hence for our case we obtain:

- $h_m(\vec{m}_E) = \vec{m}_E \cdot \vec{x}$ where $\vec{x}$ is a vector, whose components are selected at random from the gaussian distribution $X \sim N(\mu_M, \sigma_M^2)$. With μ_M and σ_M being the mean and standard deviation of the universal set of vectors for encoded messages $\vec{M}_E$ respectively**.**
- $h_t(\vec{t}_{SN}) = \vec{t}_{SN} \cdot \vec{y}$ where $\vec{y}$ is a vector, whose components are selected at random from the gaussian distribution $Y \sim N(\mu_T, \sigma_T^2)$. With μ_T and σ_T being the mean and standard deviation of the universal set of scaled and normalized transactions $\vec{T}_{SN}$ respectively**.**

The signature representation of the data point $\vec{d}$ generated by the projection $\vec{v}$ is obtained by

$$Generate_{Signature}[h(\vec{d})] = \begin{cases} 1, \wedge\, if h(\vec{d}) \geq 0 \\ 0, \wedge\, otherwise \end{cases} \quad (3.5)$$

$$[h(\vec{d}) = h_m(\vec{m}_E)] \vee [h(\vec{d}) = h_t(\vec{t}_{SN})] \quad (3.6)$$

For the projection operator to serve its intended purpose, it is required that close points get the same signature value i.e., 0 or 1, this implies that:

- $\forall i, j \in R^n$ where $i$and j are points that are close to each other in an n-ddimensional space, there is a high probability P_1 that they get the same signature value for a given projection h

$$P_H[h(i) = h(j)] \geq P_1 for d_{cosine}(i,j) \leq R_1 \quad (3.7)$$

- $\forall i, j \in R^n$ where $i$and j are points that are far apart in an n-dimensional space, there is a low probability P_2 (i.e. $P_2 << P_1$) that they get the same signature value for a given projection h

$$P_H[h(i) = h(j)] \leq P_2 for d_{cosine}(i,j) \geq R_2 \quad (3.8)$$

$d_{cosine}(i,j)$ is the cosine distance between points $i$and j

$$d_{cosine}(i,j) = 1 - \frac{i \cdot j}{\|i\|\|j\|} \quad (3.9)$$

$$CR_1 = R_2 \quad (3.10)$$

Where C is a constant. C >1

We can amplify the probability ration between P_1 and P_2 by performing K dot product operations since $\left(\frac{P_1}{P_2}\right)^n > \left(\frac{P_1}{P_2}\right)$ this leads to a **n-bit** signature representation for each message and transaction.

The algorithm to generate signature and it's time complexity are given below:

Algorithm	***Generate_Signature***
Input:	*An array **D** of transactions or messages, An array **H** of hash functions.*
Output:	***Sig_D** a 2D binary signature representation for **D***
1:	*Sig_D ← empty array list [[]]*
2:	*i←0*
3:	*for i=0 to length(D) do*
4:	*for i=0 to length(H) do*
5:	*temp← D[i]*H[i]*
6:	*if temp ≥ 0 do*
7:	*temp ← 1*
8:	*Else*
9:	*temp ← 0*
10:	*end if*

11:	*Sig_D[i] ← temp*
12:	*j ← j+1*
13:	*end for*
14:	*i ← i+1*
15:	*end for*
16	*return Sig_D*

The signature generation algorithm takes an array D of transaction data or encoded SMS and a family of hash functions H. and for every hash function in H it carries out a dot product multiplication with each data item in D. and if the resulting value is ≥ 0, then a value of 1 is generated for that particular point otherwise a 0 is generated. The process is performed for all hash functions in H this results in an array Sig_D which is a bit representation for D.

Complexity of ***Generate_Signature: O(\|D\|*\|H\|)***

3.3.2. Banding Technique for Signatures

We proceed to the banding stage where we divide our signature representations to b bands of r rows each. The values of b is selected such that $b * r = n$ where ***n*** is the number of bits that represent a signature.

- For large number of bi.e smaller r the higher the opportunity for two (2) transactions signature **Sig_T₁** and **Sig_T₂** or messages signature **Sig_M₁** and **Sig_M₂** to match for a given band and hence be hashed into the same bucket during the bucketing stage even if they are not very similar(This is the case if we want to consider the similarity threshold to be small).
- For small number of b i.e larger r it becomes hard for two of the transaction signature or message signature to harsh to the same bucket for a given band and they are few bands which give them the opportunity to do so. Thus, a small number of bands leads to a high threshold of similarity.

Similarity is the percentage of signature values in which two (2) transaction signatures or message signatures agree in.

3.3.3. Identifying Candidates by Hashing of Bands to Buckets

We proceed to perform locality sensitive hashing on the various bands of the signature representations into buckets. Buckets are simple hash table (HT) data structures that stores values in key value pairs **(**$<k:V>$**).** Where k is and integer and v is a list storing ID values(integers)of transactions or messages. Each band of r -bits is converted to its equivalent decimal representation and the decimal value correspond to the hash table (HT) key k for that band. Hence for a band B of r rows, the size of the hash table is given by 2^r**.** If a band B_i for any signature hashes to a given keyK_i then the **ID** of the transaction or message is stored in list V_i **.** Any Transactions or messages whose IDs falls in the same bucket are considered as candidate pairs.

Pairs of messages/transactions are candidate pair if they have the same key for at least one band i.e. two (2) transactions/messages x and y are candidates if $K_x \cap K_y \notin \emptyset$ where $K_x \wedge K_y$ are the set of all keys for all bands of x and y respectively.

The algorithm to put transaction and message signatures into buckets and it's time complexity are given below

Algorithm:	***DLSH_Bucketing***
Input:	***Sig_D*** a 2D binary signature representation for ***D,*** *number of buckets* ***b*** *and number of rows per bucket* ***r***
Output:	***Returns a bucket***
1:	*Sig_ID ← 1*
2:	*bucket←empty harsh_table{}*
3:	***Comment****: assert that the signature can be divisible into b bands*
4:	*Assert |Sig_D| % b==0*
5:	***for*** *i=0 to* 2^{r-1} ***do***
6:	***Comment****: Creating empty list and assigning to various bucket keys*
7:	*bucket[i] ←new_empty_list[ ]*
8:	*i ← i+1*
9:	***endfor***
10:	***for*** *i=1 to length(Sig_D)* ***do***
11:	***Comment****: |Sig_D| is the number of signatures in the dataset*
12:	***for j=1*** *to length (Sig_D[i]) do*
13:	*hash ←BinaryToInteger (Sig_D [j : j + r])*
14:	*bucket[hash] ← Sig_ID*
15:	*j ← j + r*
16:	***end for***

17:	*Sig_ID ← Sig_ID +1*
18:	*i ← i+1*
19:	***end for***
20:	*return bucket*

Complexity of *DLSH_Bucketing*: ***O(2^r) + O(|Sig_D| * (r*b))***

3.3.4. Architecture of Mobile Money Fraud Detection and Prediction System Based on Dynamic Locality Sensitive Hashing (Testing phase)

This phase involves using DLSH to make predictions (L_X) on sample message/transaction X based on candidate labels. The Testing architecture in figure below elaborates the activities of this phase (Figure-8)

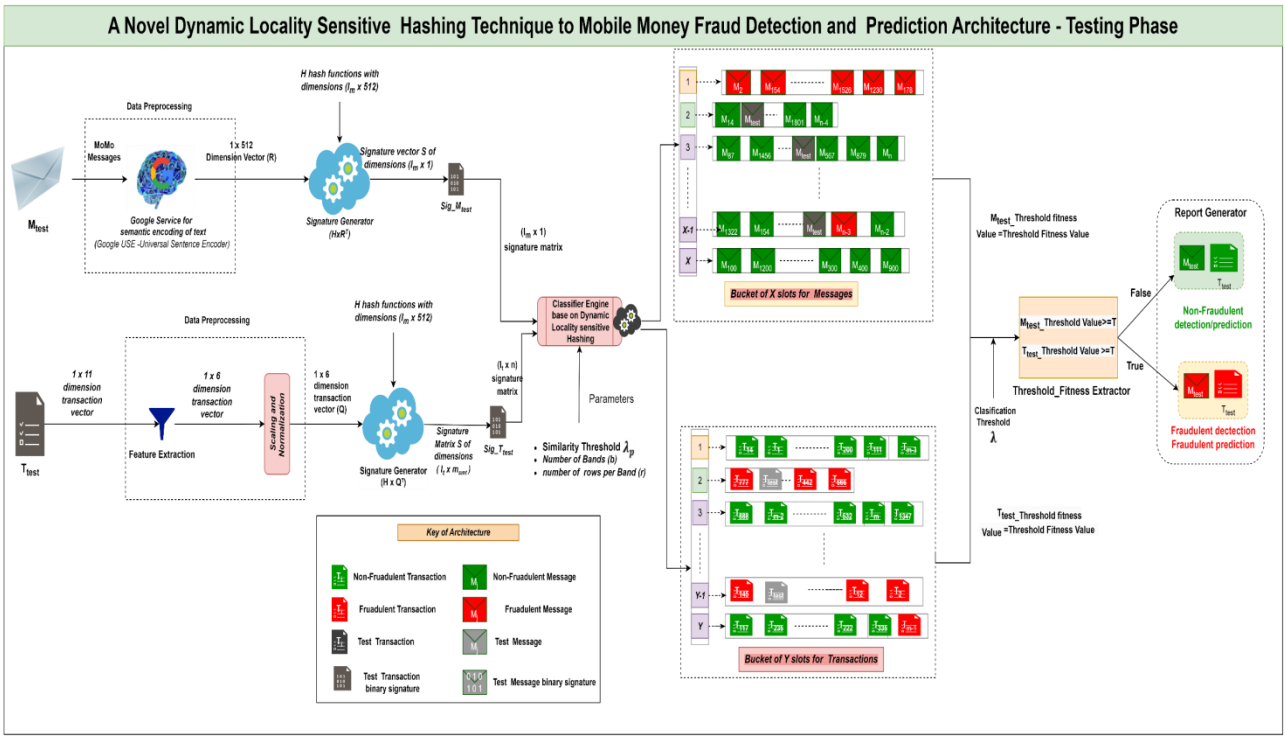

Figure-8: Architectural Design of an Integrated Mobile Money Fraud Detection Based on DLSH – Testing phase

Given a sample message or transaction instance X,

- if X is a transaction, we Select just the features as highlighted in section 3.3.1 and performing scaling and normalization using the same values for the computation as done in the training phase.
- If X is a message, we perform semantic encoding as in section 3.3.4.
- We proceed to generate signature Sig_X for X based on the same set of hash fitness functions $h_m, h_t \in h$ used in section 3.4.1 of the training phase.
- The same banding technique and values in section 3.4.2 is applied to the signature Sig_X
- We proceed to threshold fitness evaluation which is a 2-stage computational process:

Stage 1: Threshold Fitness Value

$$C(x) = Q(x, HT) \quad (3.10)$$

$$ThresholdFitnessvalue = \sum_{1}^{n} \frac{L_{cx}}{n_{cx}} \quad (3.11)$$

Where:

- $C(x)$ are the candidates of transaction/message x
- HT is a hash table of messages or transactions
- Q is a query function that takes in a value of x and a hash table HT and returns the candidates of x from HT.

- L_{cx} are the labels for $C(x), L_{cx}\epsilon[0,1]$; 0 for non-fraudulent and 1 for fraudulent labels.
- $ThresholdFitnessvalue$ is the prediction score derived by averaging all the labels of the candidate L_{cx}
- n_{cx} is the number of candidates of x.

Stage 2: Threshold Fitness Extractor

We proceed to define a threshold value λ such that**:**

$$ThresholdFitnessExtractor = \begin{cases} 1,\wedge\, if ThresholdFitnessvalue \geq \lambda \\ 0,\wedge\, otherwise \end{cases} \quad (3.12)$$

The DLSH testing (Query algorithm) is given by:

Algorithm :	***DLSH_Query***
Input Parameters:	***q*** *data to query* ***HT*** *hash table of messages/transactions* ***H*** *hash functions used in signature generation algorithm* ***b*** *number of bands* ***r*** *number of rows per band* λ_p*similarity threshold* λ classification threshold
Output:	Returns ***1*** *if* ***q*** *is fraudulent and* ***0*** *otherwise*

```
1.  q_keys ← empty hash Table{ }, candidates ← empty array list[ ]
2.  candidateLabels← empty hash Table{ }
3.  Comment: generating signature for q
4.  Sig_q ←Generate_signature (q, H))
5.  q_keys .append(Sig_q))
6.  Assert |Sig_q| % b==0
7.  for i=1 to |Sig_q| do
8.      Comment: Convert a band of the signature into integer and store it in q_keys
9.      q_keys.append(BinaryToInteger(Sig_q[i : i + r]))
10.     i ← i+(r+1)
11. end for
12.  for key in q_keys do
13.     Comment:  using the keys extract candidates from hash table HT
14.     candidates.append (HT[key])
15.  end for
16.  for candidate in candidates do
17.     candidateLabels.append(extractLabelFromCandidate(candidate))
18.  end for
19. ThresholdFitnessvalue ←Average(candidateLabels)
20. if ThresholdFitnessvalue ≥ λ
21.      return 1
22. Else
23.      return 0
24. end if
```

The DLSH Query algorithm takes as input the query data (q) which could be a semantically encoded Mobile money SMS or a scaled and normalized transaction data and a set of configuration parameters (HT,H,b,r, λ_p, λ). We proceed to convert $q$to its corresponding signature representation using the generate signature algorithm we use the parameter settings b and r to divide the signatures of b band of r rows each. A BinaryToInteger function is then used to convert each signature band into a key value and all the keys for q are stored in a list. For each of these keys, we go to the hash table HT and extract the candidates and put them in an array list Candidates. We then use a helper function extractLabelFromCandidates to get the labels for each candidate in Candidates array list. A Threshold fitness values is calculated for q by averaging the candidateLabels and if this value is $\geq \lambda$ (classification threshold), the algorithm returns 1 to signify q is fraudulent otherwise it will return 0 to signify q is not fraudulent.

Complexity of ***DLSH_Query*** = ***O(****|Sig_q|****) + O(****|q_keys|****) + O(****|candidates |****)***

$$= O(b)+O(b)+O(n)$$
$$= O(2b+n)$$
$$= O(n)$$

but b is a constant denoting the number of bands hence the complexity of the algorithm is ***O(n)*** *where n is* the total number of transactions or messages stored in the hash bucket.

4. RESULTS AND DISCUSSION

This section presents an analysis of DLSH experiment applied to Mobile Money Messages and Transactions dataset.

4.1 Analysis of Dynamic Locality Sensitive Approach

Let $b$denote the number of bands of $r$rows each and consider a pair of messages or transactions with cosine similarity s, the probability that these messages or transactions become candidates can be calculated as follows:

a) The Probability of their signatures agreeing in a single row is s
b) The Probability of signatures agreeing in all rows r of one particular band is S^r
c) The probability of signatures disagreeing in at least one row of a particular band is $1 - S^r$
d) The probability that their signature disagrees in at least one row of each of the band is $(1 - S^r)^b$
e) The probability that the signatures agree in all rows of at least one band and hence become candidates is $1 - (1 - S^r)^b$

DLSH S-Curve Analysis

Ideally, we want our DLSH to behave like an ideal Step function(S-Curve) based on selected values of r and b (Figure-9 *and* Figure-10)

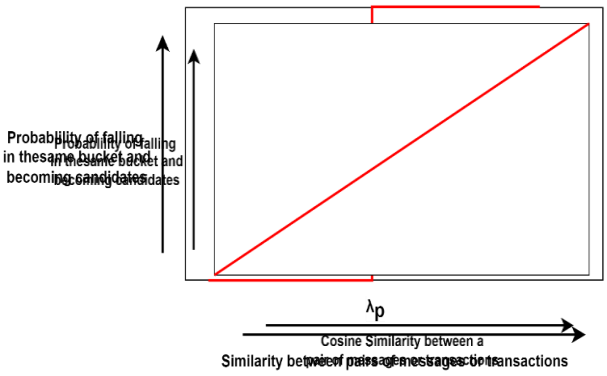

Figure-9 : Ideal S-Curve for a pair of transaction/messages

Where λ_p is the similarity threshold between pairs of transactions or messages. Transactions or Messages with $s > \lambda_p$ are guaranteed to be candidates. An approximation to λ_p as a function of b and r as proposed by(Leskovec, Rajaraman, Ventures, and Ullman, 2010) is**:** $\lambda_P = \left(\frac{1}{b}\right)^{\left(\frac{1}{r}\right)}$

Let's define a function for the probability of signatures becoming candidate for signatures divided into b bands of r rows as

$$P_b^r = (1 - (1 - s^r)^b) \quad (4.1)$$

4.1. Experimentation and Results of DLSH with Real World Dataset

In the experimentation phase of this research, we carried out two (2) sets of experiments. The experimentation environment was an Asus GL553VD i7 Model, 8GB RAM, 8 2.8GHz processors computer. The methodology described in section 3 was implemented using python programming language, Jupyter notebook 6.5.2 on a Dataspell IDE with the help of libraries such as, Numpy, pandas, Google USE library, SKlearn, Scipy seaborn and Matplotlib. The goal of the experiment was to provide answers to the research question raised in this article.

RQ: How can a novel dynamic locality sensitive hashing approach be used to efficiently and rapidly detect and predict mobile money fraud related to SMS and transactions?

And in order to answer this research questions we designed two (2) experiments that answer the sub research questions as follows

4.1.1. **Experiment 1: DLSH with Mobile Money Messages**

This experiment was conducted with 5650 real datasets obtained from (Tiago A. Almeida, 2011) augmented with 100 Mobile Money Spam SMS gotten from colleagues, friends and family. The dataset was split in a ratio of 9:1 into training and test dataset respectively. The training phase as highlighted in section 3.4 was applied to the training set while the testing phase was applied to the testing set. The following was investigated from the experiment based on our research questions:

RQ1: How can we use a novel dynamic locality sensitive hashing techniques to efficiently and rapidly detect and predict mobile money SMS fraud?

The following table depicts the results of application of DLSH on some real-world Mobile Money SMS in our dataset (Table-7*)*

Table-7: Application of DLSH on Sample Mobile Money Messages

ID	Message (M)	Encoded Message	signature Message	λ	Bucket ID(s)	Candidate ID(s)	Threshold fitness value	Fitness Extraction	Label
344	MTN is updating all user Information and require you to call 0243238543	0.225045 0.17478848 ... 0.22507 0.2250725	[[0] [0] [0] [0] [0] [1] [1] [0] ... [1] [0] [1] [0]]	0.8	[117, 3439, 3961, 1136,...,1415, 27001, 2069]	[5344, 5345, 5347 ....5341,5 , 7854,4,8]	0.871	1	1
02	Send money for Free to ALL NETWORKS via Vodafone Cash. Dial *110# and select option 1 to transfer money for free.	-4.265-02 7.748 … -4.93e-02 - 6.449e-02	[[1] [0] [0] [0] [1] ….. [1] [0] [1] [0] [1]]	0.8	[2300843089, 3893603326, ……. 1249143732, 2110570037]	[4337, 4339, ….. 5571, 5570]	0.62	0	0

To investigate the efficiency of DLSH which is expressed as a probability defined in equation 4.1 the following results were obtained from the experiment (Table-8*)*

Table-8: Probability for a Pair of Messages to Become Candidate Pairs for a Given Cosine Similarity(S) on Different DLSH Parameters b and r

Cosine Similarity (S)	P_{512}^{1} (λ_P = 0.00195)	P_{256}^{2} (λ_P = 0.0625)	P_{128}^{4} (λ_P = 0.29730)	P_{64}^{8} (λ_P = 0.5946)	P_{32}^{16} (λ_P = 0.8052)	P_{16}^{32} (λ_P = 0.9170)	P_{8}^{64} (λ_P = 0.9680)	P_{4}^{128} (λ_P = 0.9892)	P_{2}^{256} (λ_P = 0.9972)	P_{1}^{512} (λ_P = 1.0)
0	0.0000000000	0.00000000000	0.00000000000	0.00000000000	0.000000000000	0.00000000000	0.00000000000	0.00000000000	0.00000000000	0.00000000000
0.1	1.0000000000	0.92368501609	0.01271906031	0.00000064000	0.000000000000	0.00000000000	0.00000000000	0.00000000000	0.00000000000	0.00000000000
0.2	1.0000000000	0.99997106419	0.18532336796	0.00016382679	0.000000000210	0.00000000000	0.00000000000	0.00000000000	0.00000000000	0.00000000000
0.3	1.0000000000	0.99999999997	0.64690631246	0.00419037354	0.000000137749	0.00000000000	0.00000000000	0.00000000000	0.00000000000	0.00000000000
0.4	1.0000000000	1.00000000000	0.96382810704	0.04108878568	0.000013743804	0.00000000000	0.00000000000	0.00000000000	0.00000000000	0.00000000000
0.5	1.0000000000	1.00000000000	0.99974158163	0.22158039064	0.000488165784	0.00000000373	0.00000000000	0.00000000000	0.00000000000	0.00000000000
0.6	1.0000000000	1.00000000000	0.99999998077	0.66178861106	0.008988187881	0.00000127339	0.00000000000	0.00000000000	0.00000000000	0.00000000000
0.7	1.0000000000	1.00000000000	1.00000000000	0.97763124934	0.101045159430	0.00017669379	0.00000000098	0.00000000000	0.00000000000	0.00000000000
0.8	1.0000000000	1.00000000000	1.00000000000	0.99999213854	0.598938077569	0.01260145857	0.00000502167	0.00000000000	0.00000000000	0.00000000000
0.9	1.0000000000	1.00000000000	1.00000000000	1.00000000000	0.998581293746	0.42824413600	0.00939331694	0.00000556033	0.00000000000	0.00000000000
1	1.0000000000	1.00000000000	1.00000000000	1.00000000000	1.000000000000	1.00000000000	1.00000000000	1.00000000000	1.00000000000	1.00000000000

To investigate the efficiency of DLSH in detecting and predicting fraudulent messages through the ability of finding similar pairs of transaction, we obtained the following S-curve plot from our experiment with various DLSH parameter settings for **b** at various candidate points (denoted with orange dots) see Figure-11

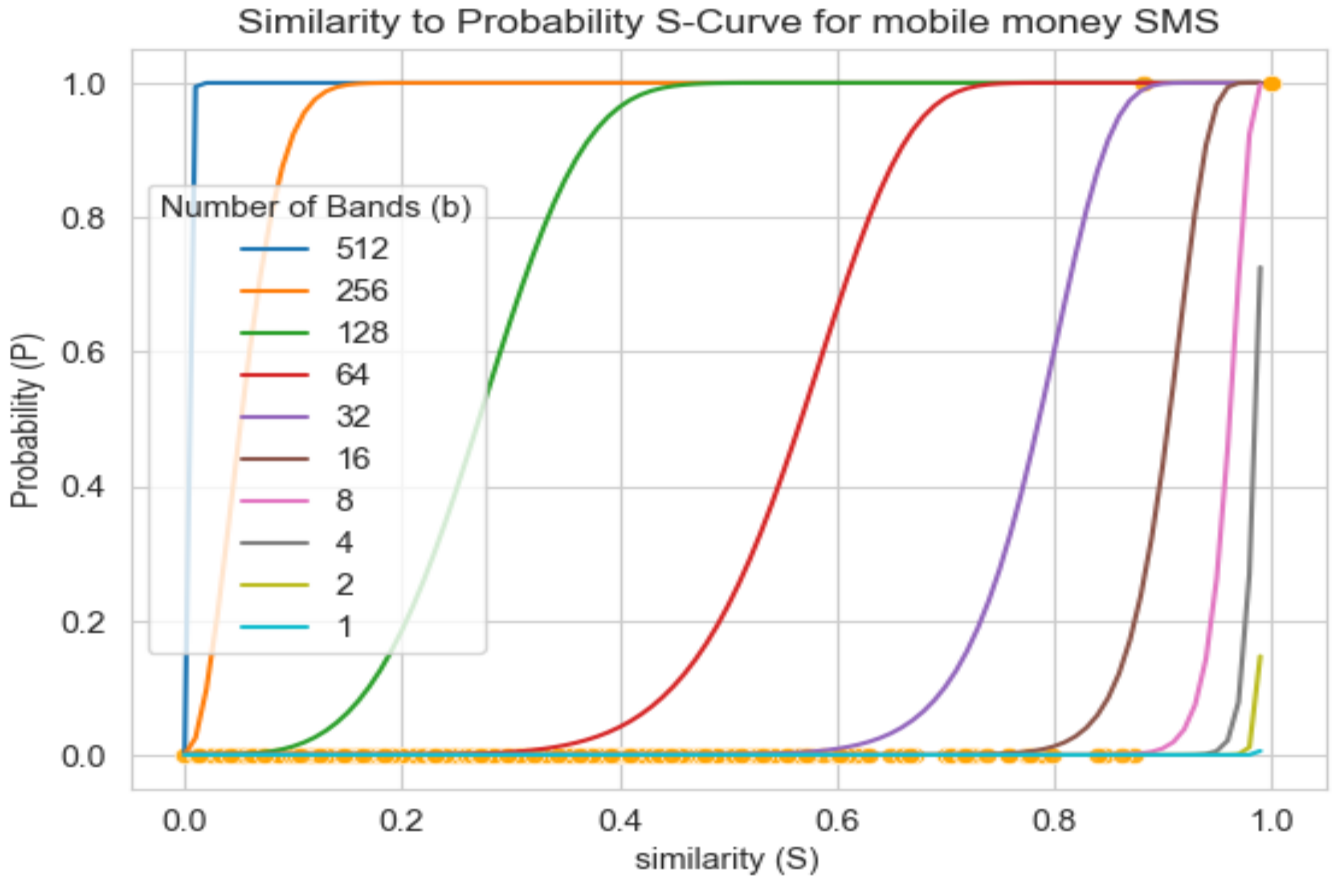

Figure-11: Similarity-Probability S-Curve for Mobile Money Messages

To investigate the running time(rapidity) of DLSH with various parameter setting compared with a naïve algorithm is presented in Table-9.

Table-9: Running Time of DLSH with Naive Algorithm for Various Message Dataset Size

Dataset size	b=1	b=2	b=4	b=8	b=16	b=32	b=64	b=128	b=256	b=512	Naïve algorithm
100	102	104	108	116	132	164	228	356	612	1124	4999.5
500	502	504	508	516	532	564	628	756	1012	1524	124999.5
1000	1002	1004	1008	1016	1032	1064	1128	1256	1512	2024	499999.5
1500	1502	1504	1508	1516	1532	1564	1628	1756	2012	2524	1124999.5
2000	2002	2004	2008	2016	2032	2064	2128	2256	2512	3024	1999999.5
2500	2502	2504	2508	2516	2532	2564	2628	2756	3012	3524	3124999.5
3000	3002	3004	3008	3016	3032	3064	3128	3256	3512	4024	4499999.5
3500	3502	3504	3508	3516	3532	3564	3628	3756	4012	4524	6124999.5
4000	4002	4004	4008	4016	4032	4064	4128	4256	4512	5024	7999999.5
4500	4502	4504	4508	4516	4532	4564	4628	4756	5012	5524	10124999

											.5
5085	5087	5089	5093	5101	5117	5149	5213	5341	5597	6109	12928612

The line graph for the time complexity for various DLSH Band settings and various dataset size is show below (Figure-12).

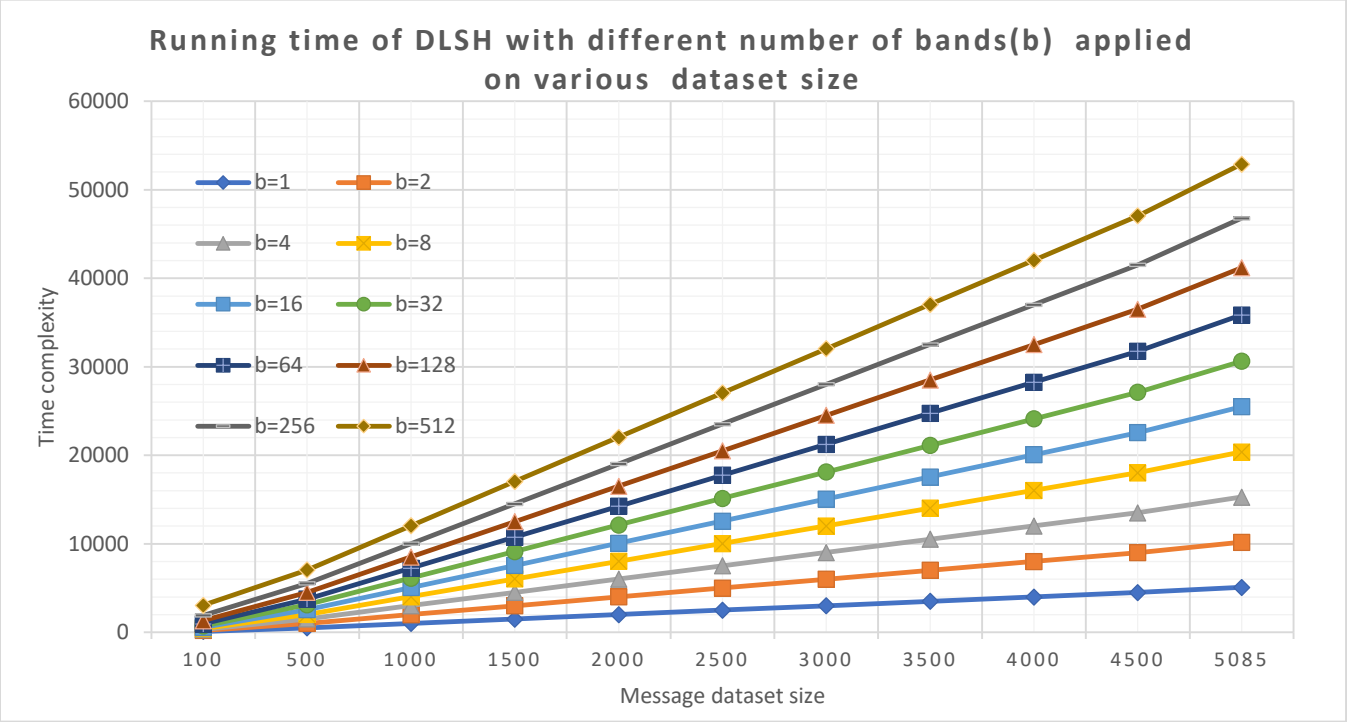

Figure 12: Running Time of DLSH on Messages with Different Bands(b)

We then investigated further by comparing the time complexity of DLSH with different settings for b and the naïve based approach. The following graph was obtained(Figure-13 and Figure -14).

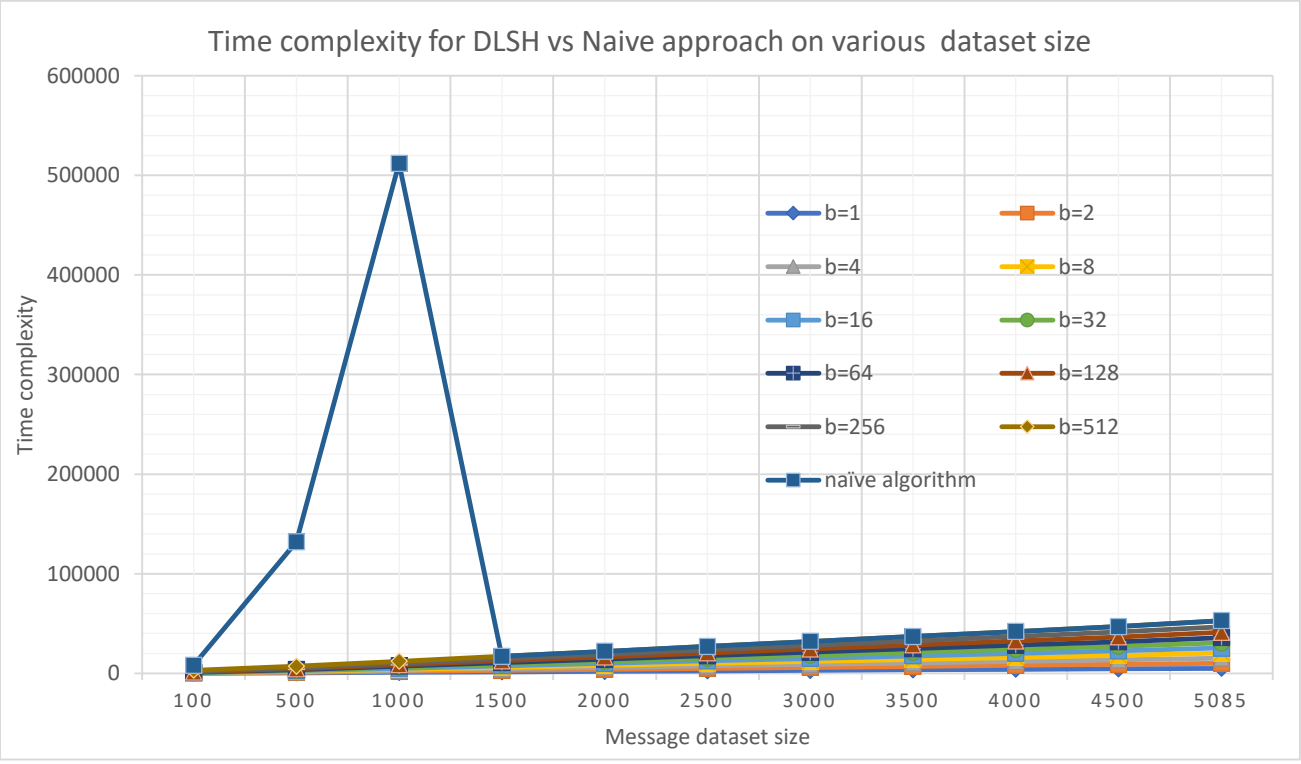

Figure-13: Running Time for DLSH vs Naive Approach for Messages

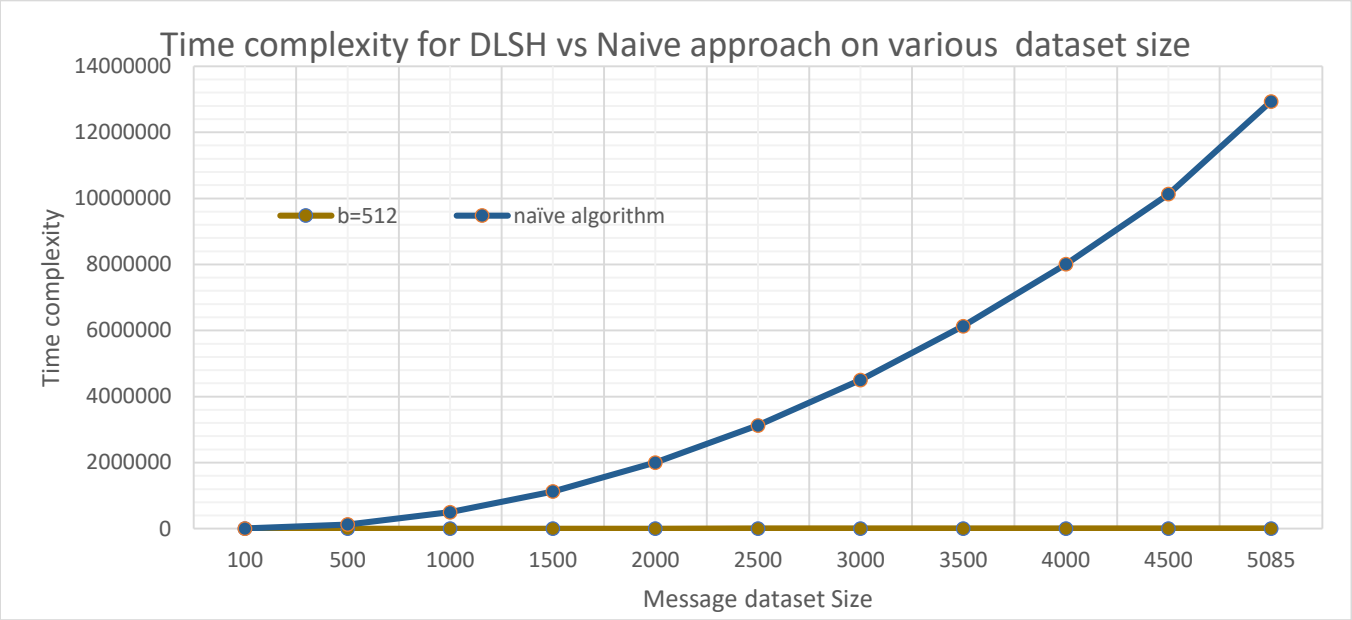

Figure-14: Running Time for DLSH vs Naive Approach for Messages

4.1.2. Experiment 2: DLSH with Mobile Money Transaction

The second experiment was conducted to provide answers to our second research question:

RQ2: How can we use a novel dynamic locality sensitive hashing techniques to efficiently and rapidly detect and predict mobile money transaction fraud?

This experiment was conducted with synthetic data obtained from (Alonso [5])) made up of 532021 transactions. We proceed to carry out data preprocessing as describe by section 3.3 where we explored the data to remove null values, proceed to carryout feature selection and realizing the skewness of the data, we applied SMOTE-Tomek in order to resolve this problem. We end up with 600000 transactions which we divided in the ratio 8:2 for training and testing respectively. To investigate the efficiency of DLSH in detecting Fraudulent transactions, we investigate the probability of a transaction being similar to other pair of transactions and end up being classified as fraudulent or non-fraudulent. The following (Table-10*)* was obtained for various bands, rows and similarity threshold.

Table-10: Probability for a Pair of Transactions to Become Candidate Pairs for a Given Cosine Similarity(S) on Different DLSH Parameters b and r

cosine similarity	P^{1}_{1113} ($\lambda_P = 0.00089$)	P^{3}_{371} ($\lambda_P = 0.1391$)	P^{7}_{159} ($\lambda_P = 0.4847$)	P^{21}_{53} ($\lambda_P = 0.8277$)	P^{53}_{21} ($\lambda_P = 0.9441$)	P^{159}_{7} ($\lambda_P = 0.9878$)	P^{371}_{3} ($\lambda_P = 0.9970$)	P^{1113}_{1} ($\lambda_P = 1.00$)
0	0.0000000000	0.00000000000	0.00000000000	0.00000000000	0.000000000000	0.00000000000	0.00000000000	0.00000000000
0.1	1.0000000000	0.31008413510	0.00001589987	0.00000000000	0.000000000000	0.00000000000	0.00000000000	0.00000000000
0.2	1.0000000000	0.94920390003	0.00203314338	0.00000000000	0.000000000000	0.00000000000	0.00000000000	0.00000000000
0.3	1.0000000000	0.99996110769	0.03417932922	0.00000000055	0.000000000000	0.00000000000	0.00000000000	0.00000000000
0.4	1.0000000000	0.99999999998	0.22950278562	0.00000023310	0.000000000000	0.00000000000	0.00000000000	0.00000000000
0.5	1.0000000000	1.00000000000	0.71265313850	0.00002527206	0.000000000000	0.00000000000	0.00000000000	0.00000000000
0.6	1.0000000000	1.00000000000	0.98905058818	0.00116199550	0.000000000037	0.00000000000	0.00000000000	0.00000000000
0.7	1.0000000000	1.00000000000	0.99999883767	0.02917708501	0.000000129543	0.00000000000	0.00000000000	0.00000000000
0.8	1.0000000000	1.00000000000	1.00000000000	0.38805143687	0.000153446459	0.00000000000	0.00000000000	0.00000000000
0.9	1.0000000000	1.00000000000	1.00000000000	0.99784874530	0.076004181711	0.00000037124	0.00000000000	0.00000000000
1	1.0000000000	1.00000000000	1.00000000000	1.00000000000	1.000000000000	1.00000000000	1.00000000000	1.00000000000

The corresponding s-curve for mobile money transactions is shown below with green dots showing candidate points for given similarity values (Figure-15*)*.

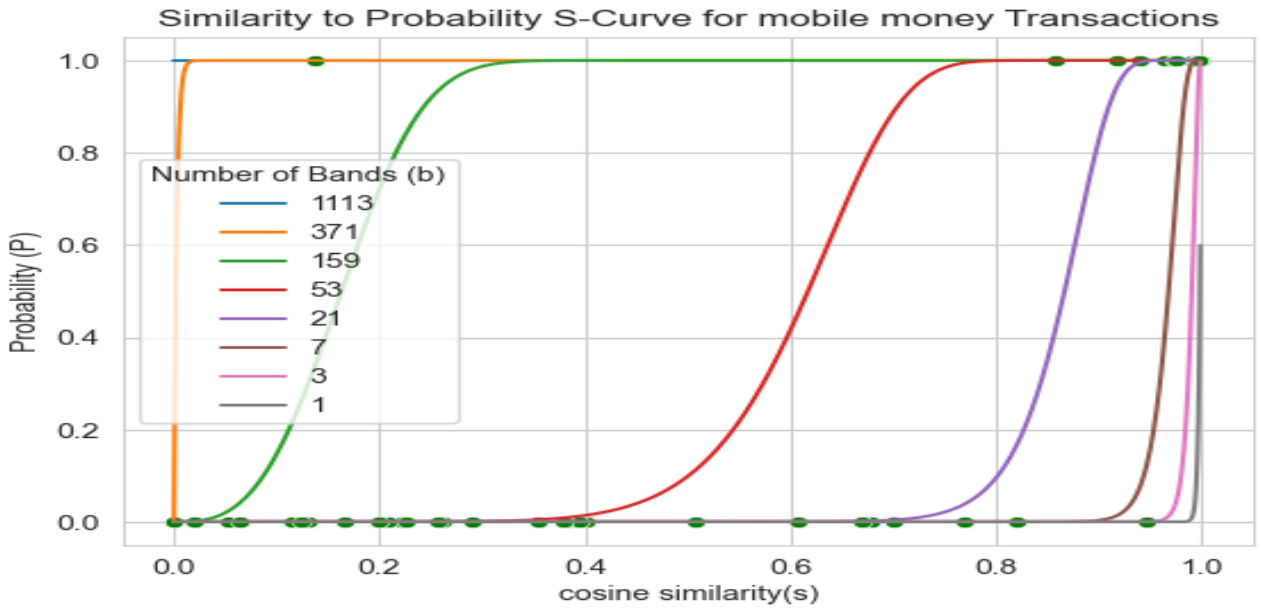

Figure-15: Similarity-Probability S-curve for Mobile Money Transactions

To further investigate the running time of DLSH on transaction fraud detection, we obtained the following table for various DLSH parameter settings (Table-11).

Table-11: Running Time of DLSH with Naive Algorithm for Various Transaction Dataset Size

Dataset Size	b=1	b=3	b=7	b=21	b=53	b=159	b=371	b=1113	naïve algorithm
5000	5002	5006	5014	5042	5106	5318	5742	7226	12499999.5
10000	10002	10006	10014	10042	10106	10318	10742	12226	49999999.5
50000	50002	50006	50014	50042	50106	50318	50742	52226	1250000000
100000	100002	100006	100014	100042	100106	100318	100742	102226	5000000000
200000	200002	200006	200014	200042	200106	200318	200742	202226	20000000000
300000	300002	300006	300014	300042	300106	300318	300742	302226	45000000000
375000	375002	375006	375014	375042	375106	375318	375742	377226	70312500000
450000	450002	450006	450014	450042	450106	450318	450742	452226	1.0125E+11
478819	490002	490006	490014	490042	490106	490318	490742	492226	1.2005E+11

The corresponding line plot obtained is shown below (Figure-16)

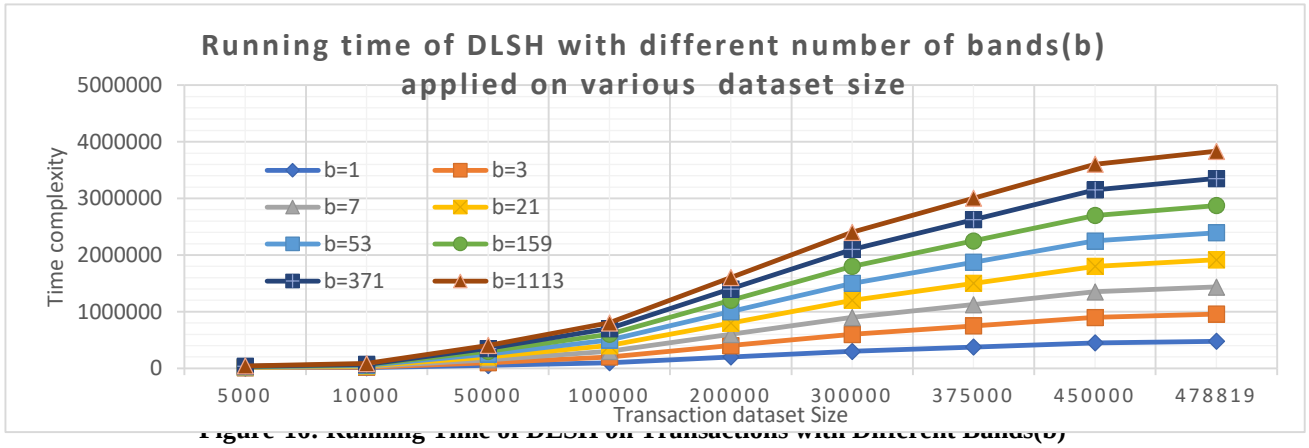

Figure-16: Running Time of DLSH on Transactions with Different Bands(b)

A comparison of DLSH with different b settings and the Naïve approach yields the following curve (Figure-17)

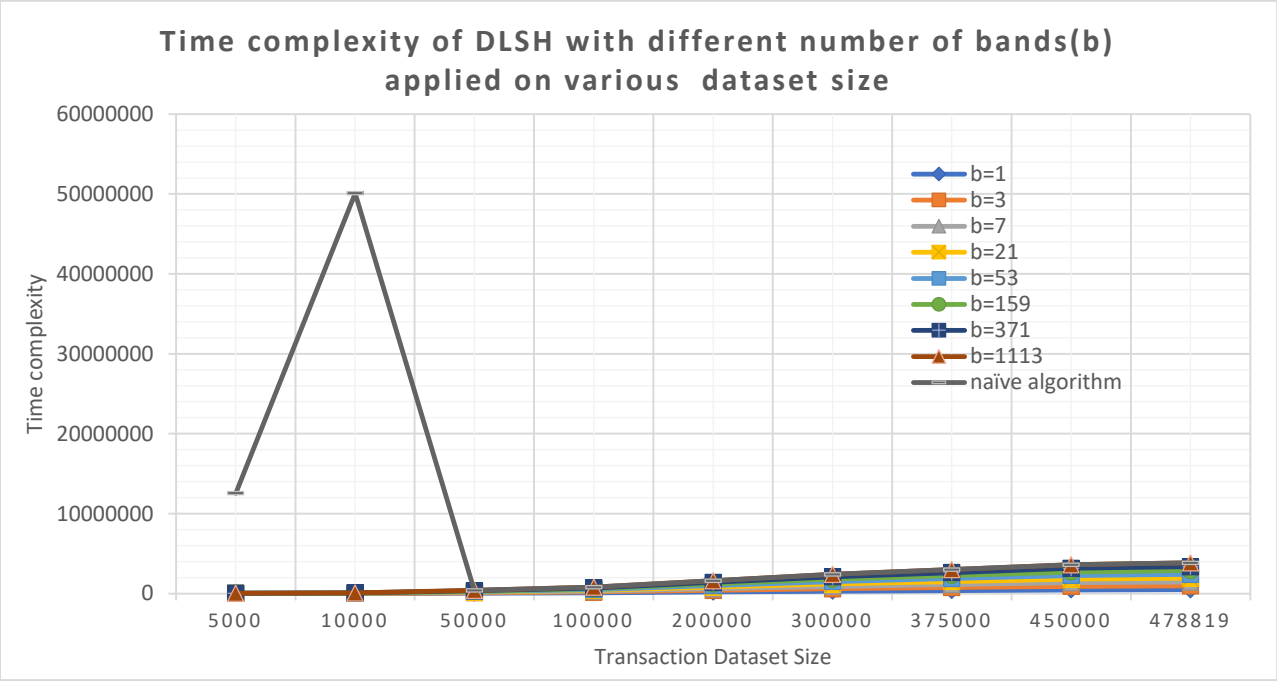

Figure-17: Running Time of DLSH vs Naive Approach for Transactions

4.2. Discussion of Results

Efficiency of DLSH

From the experiment to verify the efficiency of DLSH in detecting fraudulent Transaction and Messages, we can observe that at a high value of **b,** the higher the probability of a pair of transaction or message to be hashed into the same bucket and hence be candidates even with a low cosine similarity value. Inversely, the smaller the number of bands b, the smaller the probability of any pair of transaction or message even with a high similarity value to be hashed into the same bucket and become candidates. Hence an optimal parameter setting is needed in which transactions with high cosine similarity are hashed into the same bucket and those with low similarity values are hashed into separate buckets.

The efficiency of DLSH is calculated using the following formula:

$$\text{Efficiency} = \frac{numberofcorrectpredictions}{totalnumberofpredictions} \quad (4.1)$$

The question of which DLSH setting is optimal? is highly based on the properties of the dataset, as we can see from Figure-15 the optimal parameter for DLSH applied to messages was obtained **b=32, r= 16**, $\lambda_p = 0.805$ when trained with 5175 messages and tested with 575 messages, 567 messages were predicted correctly out of the 5175 messages giving a prediction efficiency of 0.986 (98.6%). (see Figure-18).

As seen in Figure-1,7 parameter for optimal DLSH was obtained when **b=53, r= 21**, $\lambda_p = 0.8277$ with a training set of 480,000 and tested with 120,000 transactions. From the total testing set, 118,800 transactions were predicted correctly giving an accuracy of 0.99 (99.0%) (see Figure-18).

```
In 243   1  print("Mobile Money Effciency Scores")
         2  print(f"Mobile Money SMS: {efficiency_SMS}")
         3  print(f"Mobile Money Transaction: {efficiency_Transaction}")

         ⌄    Mobile Money Effciency Scores
              Mobile Money SMS: 0.98608
              Mobile Money Transaction: 0.99
```

Figure-18: Data Spell Screenshot of Efficiency Results

Running Time(Rapidity) of DLSH

From both experiments it can be observed that the running time increases as the size of the dataset grows, also, as the number of bands used for DLSH increases, the higher the running time. This is depicted in Figure-12 and Figure-13 comparing the running time of DLSH with the naïve approach (See Figure-14), with 500 messages the naïve approach requires 124999 comparison operations while the worst performing DLSH(b=512) performs 1524 comparison operations (Figure-13) shows the pattern (Figure-14) shows a similar pattern for transaction data with the naïve approach performing 12,499,999 operations for 5000 transactions while DLSH(b=1113) performs 7226 operations.
Using the computer described in the illustrative example that performs 1000 comparison operations per second, for 500 messages DLSH(b=512) would be performed in approximately 1.5 seconds while the naïve approach will execute in 2.08Minutes. For 5000 transaction data the naïve approach performs the comparison operations in 3.47Hours, while DLSH(b=1113) perform just 7226 operations in 2.0Minutes (Figure-15).

4.3. Comparison of DLSH with Different Approaches

After we had implemented our approach, we did a comparative analysis with other state of the art techniques (Table-12).

Table-12: Comparison of DLSH with State-of-the-art Approaches for Mobile Money Fraud Detection and Prediction

Approaches	% Efficiency	Speed	Scalable	Detection and prediction ability	Computational Cost
Case Based Reasoning [2]	98% prediction efficiency	Fast	NO	-Transaction	Very High
Cross-case Analysis Approach [8]	- Support Vector Machine: 99.91% - Naïve Bayes algorithm: 99.65% - Gradient boosted Decision tree: 89.9%	Fast	YES	Transaction	High
A Process Behavior Analysis Approach [5]	Unknown	Slow	NO	Transaction	High
Dynamic Locality Sensitive hashing (Our Approach)	- SMS: 98.6% - Transaction: 99%	Very Fast	YES	-SMS -Transaction	Low

Table-12 clearly shows our approach is superior in speed, scalability, detection and prediction abilities and the computational cost required. But the cross-case analysis approach using Support vector machine is the most efficient with a 99.91% prediction accuracy on transaction data.
From the results obtain we were able to provide answers to our research questions and attain our objectives of being able to use DLSH a Search Based Software Engineering approach to efficiently and rapidly identify fraudulent messages and transactions all in an effort to reduce fraud across mobile money network.

5. CONCLUSION AND RECOMMENDATION

we are going to summarize what we have been able to achieve throughout this entire research and also discuss on the challenges encountered and give recommendations for future studies.

5.1 Summary of Findings

Throughout this research we focused on fraud research on the mobile money sector which can be taggled by technological means. we explore how we can apply the state-of-the-art technique of locality sensitive Hashing of nearest neighbour search to the problem of detecting fraud in mobile money messages and transactions. The approach shows encouraging results for both Mobile money messages and transaction datasets.

5.2 Challenges Encountered

Despite these outstanding results, the research was not all rosy as we encountered the following challenges:

1) Difficulty in obtaining data for the research due to privacy concerns. During this research the greatest challenge was that of obtaining real world dataset both SMS and Transaction. This is due to the sensitive nature of financial data.
2) We wanted to translate this work to an actual industrial context but the opportunity was not granted due to the lack of real-world data mentioned above.
3) Exploring the vast domain of fraud detection and Finance was challenging
4) Mobile money being a new and promising field, it was very difficult to find information necessary for the research.

5.3 Conclusion

Conclusively, we have been able to successfully design and develop a Mobile fraud detection system using locality sensitive hashing whcih is a Search Based Software Engineering approach, while realizing promising results on efficiency, speed and scalability. The results showed a 98.6% efficiency for SMS fraud data detection and prediction and 99% for transaction data while yielding an running time for both dataset where n is the dataset size. These results are very encouraging and will go a long way to reducing the amount of fraud on mobile money networks and hence an increase in customer trust and adoption of mobile money services.

4.1. **Recommendation and Further Studies**

For future perspective, we recommend the following:

1) Given the promising results obtained, we recommend that this work be applied on real industrial data in order to minimize fraud by employing more robust search base software engineering techniques for optimal results.
2) Explore means of apply DLSH to taggling Mobile fraud related to fraudulent calls using advanced AI techniques
3) Explore possibilities of extending this research to tackle process related fraud such as money laundry within the mobile money sector
4) Adapt the Architecture proposed in the research to a distributed environment such as a Map Reduce computational model to further speed up the process of fraud detection and prediction
5) Explore means of adapting the proposed Locality sensitive hashing technique in handling fraud related problems in other domains such as bank transactions etc.

REFERENCES

[1] Debray, A., Kwon, H., and Gill. R. (2014). *Mobile Money Opportunities for Mobile Operators*.
[2] Adedoyin, A. (2018). Predicting Fraud in Mobile Money Transfer. Retrieved from PhD thesis, computer Science Department, University of Brighton :
[3] Adedoyin, A., Kapetanakis, S., Samakovitis, G., and Petridis, M. (2017). Predicting fraud in mobile money transfer using case-based reasoning. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10630 LNAI, 325–337. Retrieved from https://doi.org/10.1007/978-3-319-71078-5_28
[4] Alonso Lopez-Rojas, E., Axelsson, S., and Baca, D. (2017). *Analysis of Fraud Controls Using the PaySim Financial Simulator*.
[5] Alonso Lopez-Rojas EalaX, E., Axelsson, S., Alonso Lopez-Rojas, E., and Elmir, A. (2016). *PAYSIM: A Financial Mobile Money Simulator for Fraud Detection.* Retrieved from https://www.researchgate.net/publication/313138956
[6] Awanis, A., Lowe, C., Andersson-Manjang, S. K., and Lindsey, D. (2021). *State of the Industry Report on Mobile Money - 2021*. Retrieved from www.gsma.com/mobilemoney
[7] BEAC. (n.d.). Payment-by-electronic-money-services-in-CEMAC-EN 2020.
[8] Botchey, F. E., Qin, Z., and Hughes-Lartey, K. (2020). Mobile money fraud prediction: A cross-case analysis on the efficiency of support vector machines, gradient boosted decision trees, and Naïve Bayes algorithms. *Information (Switzerland)*, 11(8). Retrieved from https://doi.org/10.3390/INFO11080383 Leskovec, Rieke Wiem [1] Debray, A., Kwon, H., and Gill. R. (2014). *Mobile Money Opportunities for Mobile Operators*.
[2] Adedoyin, A. (2018). Predicting Fraud in Mobile Money Transfer. Retrieved from PhD thesis, computer Science Department, University of Brighton :
[3] Adedoyin, A., Kapetanakis, S., Samakovitis, G., and Petridis, M. (2017). Predicting fraud in mobile money transfer using case-based reasoning. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10630 LNAI, 325–337. Retrieved from https://doi.org/10.1007/978-3-319-71078-5_28
[4] Alonso Lopez-Rojas, E., Axelsson, S., and Baca, D. (2017). *Analysis of Fraud Controls Using the PaySim Financial Simulator*.
[5] Alonso Lopez-Rojas EalaX, E., Axelsson, S., Alonso Lopez-Rojas, E., and Elmir, A. (2016). *PAYSIM: A Financial Mobile Money Simulator for Fraud Detection.* Retrieved from https://www.researchgate.net/publication/313138956
[6] Awanis, A., Lowe, C., Andersson-Manjang, S. K., and Lindsey, D. (2021). *State of the Industry Report on Mobile Money - 2021*. Retrieved from www.gsma.com/mobilemoney
[7] BEAC. (n.d.). Payment-by-electronic-money-services-in-CEMAC-EN 2020.
[8] Botchey, F. E., Qin, Z., and Hughes-Lartey, K. (2020). Mobile money fraud prediction: A cross-case analysis on the efficiency of support vector machines, gradient boosted decision trees, and Naïve Bayes algorithms. *Information (Switzerland)*, 11(8). Retrieved from https://doi.org/10.3390/INFO11080383
Leskovec, Rieke Wiem
[9] Cer, D., Yang, Y., Kong, S., Hua, N., Limtiaco, N., John, R. St., … Kurzweil, R. (2018). Universal Sentence Encoder. Retrieved from http://arxiv.org/abs/1803.11175
[10] Chadha, S., Kipkemboi, K., and Muthiora, B. (2021). *The Mobile Money Regulatory Index 2021: Regional and Country Profiles 1 The Mobile Money Regulatory Index 2021 REGIONAL and COUNTRY PROFILES*.
[11] Nkoy, J. and Mohammed. A. (2021). Mobile Money SMS Fraud Detection. *Stanford CS 229 Final Project Report*.
[12] Leskovec, J., Rajaraman, A., Ventures, R. and Ullman, J. D. (2010). *Mining of Massive Datasets*.
[13] Rieke, R., Zhdanova, M., Repp, J., Giot, R., and Gaber, C. (2013). *Fraud Detection in Mobile Payments Utilizing Process Behavior Analysis*. Retrieved from https://hal.archives-ouvertes.fr/hal-00841002.
[14] Tiago A. and Almeida, J. M. G. A. Y. (2011). SMS Spam Collection Dataset. Retrieved 16 November 2022 from https://doi.org/https://doi.org/10.1145/2034691.2034742.
[15] Harman, M. and Jones, B. F. "Search-based software engineering," Inf. Softw. Technol., vol. 43, no. 14, pp. 833–839, 2001.
[16] M. Harman, S. A. Mansouri, and Y. Zhang, "Search-based software engineering: Trends, techniques and applications," ACM Comput. Surv., vol. 45, no. 1, p. 11, 2012.
[17] M. M. K. Wiem, D. Kalyanmoy and O. C.Mel, "High Dimensional Search-based Software Engineering: Finding Tradeoffs Among 15 Objectives for Automating Software Refactoring Using NSGA-III," ACM Comput. Surv., vol. I4, no. 7, 2024.