ATHABASCA UNIVERSITY

IDENTIFYING MALICIOUS VOIP USAGE USING COMPUTATIONAL INTELLIGENCE

BY

JASON MCKELLAR

A THESIS
SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTERS OF SCIENCE IN COMPUTER INFORMATION SYSTEMS

SCHOOL OF COMPUTING AND INFORMATION SYSTEMS,
FACULTY OF SCIENCE AND TECHNOLOGY
ATHABASCA UNIVERSITY

OCTOBER, 2018

Athabasca University
FACULTY OF GRADUATE STUDIES

The future of learning.

Approval of Thesis

The undersigned certify that they have read the thesis entitled

IDENTIFYING MALICIOUS VOIP USAGE USING COMPUTATIONAL INTELLIGENCE

Submitted by

Jason McKellar

In partial fulfillment of the requirements for the degree of

Master of Science in Information Systems

The thesis examination committee certifies that the thesis
and the oral examination is approved

Supervisor:
Dr. Mahmoud Abaza
Athabasca University

Committee Member:
Dr. Ching Tan
Athabasca University

External Examiner:
Dr. Ebrahim Bagheri
Ryerson University

October 22, 2018

Abstract

VoIP user accounts are a prime target for hackers to compromise for profit. VoIP accounts are targets of the same types of attacks as any other Internet account that is authorized with a username and password. Unlike many other Internet accounts VoIP has a direct monetary cost to the user being compromised. Toll-fraud perpetrated using a compromised VoIP account can accrue expensive toll-charges that either the user or the service provider are liable to pay for. This paper discusses the prior research in detecting unauthorized usage on VoIP accounts. The researched methods are based on machine learning techniques. A new technique of using a Recurrent Neural Network for detecting unauthorized usage periods on a VoIP account is developed and demonstrated. The technique uses a Long-Short Term Memory style of Recurrent Neural Network to achieve over a 99% accuracy when testing against calls tagged as occurring during a toll-fraud event.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter I

INTRODUCTION

This research describes the significant problem of toll-fraud against VoIP providers. Chapter 2 will review how VoIP works — specifically with the internet telephony service provider (ITSP) that will be providing the call data. A literature review will discuss the methods of toll-fraud detection that have been researched. The patterns of calls and fraud events at the ITSP is analyzed and discussed. The Design chapter explains the process for ingesting data and configuring an algorithm to detect toll-fraud. The discussion chapter will present the algorithm parameters and the results of testing the algorithm on a dataset of calls tagged as occurring during a fraud event.

1.1 Significance of the Problem

Voice-over-IP (VoIP) provides telephony access anywhere there is an Internet connection. VoIP user accounts are a prime target for malicious entities to compromise for profit. Once a hacker has access to an account many calls can be made to high-toll destinations. A VoIP service provider was even found to be routing all of their calls through hacked VoIP accounts in order to bypass any telephone fees they would normally pay. *[12]* The mechanisms that allow VoIP to be exploited and the motivation behind those exploits will be discussed in this section. This fraudulent usage impacts both Internet telephone service providers (ITSP) and customers of VoIP services.

1.1.1 What is VoIP

VoIP is gaining greater popularity as the method to exchange telephone calls. The FCC reports that VoIP usage was growing at a rate of 16% per year from 2010 to 2013 *[11]*. They give the total VoIP business usage as 8.9 million subscribers — about 6.6% of total phone lines — at the end of 2013. Adoption in Canada is lower with 0.8 million subscribers — about 3.9% of total phone lines — in 2013 *[6]*.

The traditional publicly switched telephone network (PSTN) uses a set of wires that physically connects a customer location with the telephone company. This physical circuit of telephone subscriber cannot be used remotely as it requires a physical connection. Voice-over-IP (VoIP) allows making telephone calls over an Internet connection. A VoIP subscriber can connect to the VoIP provider and make calls through any Internet connection. Instead of a physical connection to identify a user VoIP accounts use a username to identify the user and a password to authenticate. Phone services often include a phone-in interface to voicemail and other simple phone settings that can be changed with numeric choices. Being an Internet connected service VoIP providers usually offer a web interface for updating settings such as call forwarding, checking voicemail, calling contacts, and other extended functionality. Unlike the traditional phone accounts a VoIP subscriber can make multiple concurrent calls. This allows features such as placing multiple calls on hold and making calls from multiple locations. To audit calls and provide billing information the VoIP system outputs call detail records (CDR). These are structured records containing information about every call made on the system. The structure and protocols of VoIP make it possible for providers to easily add features such as complex call-forwarding rules, voicemail, conferencing, and other features above the common telephone service.

1.1.2 Toll-Fraud

Toll-fraud occurs when someone makes toll calls using a VoIP account that does not belong to them. The cost of the toll-charges becomes the responsibility of the ITSP or VoIP subscriber. Fraud can also be described as any telecommunications-related activity committed

with no intention of paying *[17]*.

Making Unauthorized Calls

To make calls on an account that does not belong to them the attacker must find credentials to a VoIP account and a way to make calls. The SIP username and password can be cracked by guessing or brute-forced if the password is simple enough. This allows the unauthorized user access to make calls directly from a softphone or SIP phone with those credentials. The SIP username and password could be found indirectly by having access to the configuration of an authorized users softphone or SIP phone. Many SIP phones have web portals that use a publicly available default password. If this password is not changed it is trivial for the attacker to login and view the SIP username and password. If the attacker can access the authorized users computer they may have access to SIP credentials saved as part of a softphone configuration.

An attacker does not require the SIP credentials to make unauthorized calls. Logging into a VoIP providers web portal will allow making click-to-dial calls or change the users forwarding settings. The forwarding settings can be updated to forward incoming calls to the desired destination. The attacker can call the hijacked VoIP account and be forwarded to the destination they setup. The forwarding settings can also be changed through the voice portal which authenticates with a numeric PIN. These credentials can also be guessed, brute-forced, or discovered by hacking the users computer or SIP phone.

The VoIP attacker does not need to manually attempt to guess or brute-force the account credentials. They can use automated tools to scan VoIP providers and attempt to determine passwords for each account. The automated tools may also attempt to crack passwords of well-known SIP phones and softphones. They could even lease time on a botnet to scan many providers at once.

Malicious Destinations

There are multiple reasons an attacker would want to make an unauthorized call. Calls to high-toll destinations are an obvious reason to hijack a VoIP account. The attacker can then call those destinations without paying for the call. If this was the only use-case the number of calls and amount of the charges for a single hacked account would be relatively small considering how many calls a single person could make in a day. The attacker may offer this hacked account to others to also use to make free high-toll calls. VoIP accounts are not limited to a single call at a time, so many people may use the same hacked account concurrently.

There are more efficient ways for VoIP attackers to profit. Hijacked VoIP accounts can be pooled together and used as a rudimentary switching platform to route calls through. Long distance calling cards can be created and sold. A buyer calls the number on the calling card and then dials their long distance destination. What happens is that when dialing the calling card number they are routed to the switching platform which accepts the destination digits. The call then gets automatically routed through an available hijacked VoIP account. The attacker only incurs the cost of their switching platform and not the toll charges for the calls. This type of scheme can also be used for to setup a VoIP provider service. The malicious VoIP provider sells VoIP accounts as a legitimate service would, but all of the VoIP calls are routed through other VoIP providers hijacked accounts. An example of this type of toll-fraud that was successfully prosecuted is that of Edward Pena in 2010 reported by as *[12]*. Pena ran a VoIP service provider in Miami, US that was routing it's customers calls through hijacked VoIP accounts. The hijacked accounts, which came from over 15 different VoIP providers, had accrued more than 10 million minutes of fraudulent calls. This cost the affected service providers an estimated $1.4 million. The FBI press release indicates that one of the victim providers had more than 500,000 calls routed through hijacked accounts *[10]*. This case shows the extent of financial damage caused by a single malicious actor. The FBI press release reports that Pena faces up to 25 years in prison for his actions. Most VoIP fraud does not initiate from a country with robust telecommunication laws such as the US. This makes it practically infeasible to pursue and

punish VoIP attacks in most cases.

Another use-case for hacked accounts is to patronize premium-rate services that are run by the VoIP attacker. A premium number (such as a 1-900 number) is setup by the attacker. Any calls made to the premium number will charge the calling number a set amount per minute. The fees are paid to the VoIP attacker making it advantageous for them to call their own premium rate number with hijacked VoIP accounts. An example of this mass fraud is a VoIP attacker that was on the FBI's cyber most wanted list *[9]*. The International Business Times reports that Noor Aziz Uddin was cracking VoIP accounts and using them to call premium rate services he owned. The VoIP attackers made over $50 million during four years they were in business *[23]*. This case shows how a VoIP attacker can target victims globally without discrimination as the FBI wanted poster lists at least eight countries in which VoIP providers had accounts hijacked.

These mass-calling schemes are most effective for the VoIP attacker when they have access to many VoIP accounts. An account could be locked at any moment, but if they have enough hijacked accounts they will always have one available for use with their calling card or illegitimate VoIP provider. The number and length of calls made to premium rate numbers directly correspond to the profits made by the attacker.

Impact

Toll-fraud does not discriminate between small and large VoIP providers. A small service provider pays the same toll costs as a large provider (if not more due to economies of scale) and can be disproportionately effected by even a single toll-fraud case. In general terms *[5]* estimates that global VoIP fraud in 2013 cost service providers $46.3 billion. Besides the financial burden of fraud there are other costs associated. Reputation, customer satisfaction, lost opportunities, and passing fraud costs into the price of services are given by *[26]* as other consequences of VoIP fraud. *[26]* estimates that 3%-5% of VoIP providers annual revenue is lost to fraud. A single toll-fraud event with a single VoIP account can accrue costs that reach thousands of dollars in charges within days.

The reported impact on VoIP providers is great, but is also under-reported. VoIP providers may not report fraud incidents in an attempt to hide details of fraud from potential attackers and to protect their reputation *[18]*.

1.2 Research Goals and Contribution

This research aims to determine whether a recurrent neural network (RNN) can be applied to CDRs can be used to identify compromised VoIP accounts or malicious usage of a VoIP account. Additionally to determine how quickly fraud usage can be identified with an RNN.

Toll charges can accrue very quickly on a compromised VoIP account. An important goal of this research is to test how soon after an account is compromised machine learning and computational intelligence techniques can determine that it is compromised. The optimal outcome would be finding a technique that can confidently identify unauthorized usage immediately after fraudulent usage begins.

1.3 Research Limitations and Scope

Each VoIP provider will have an inherently subjective data set. Each prior research study has their own proprietary data set that was used in performance testing. Although the performance is researched and documented it cannot be directly compared to the performance of the result of this research. Thus directly comparing performance between prior research and this new research is not possible and is out of scope, although it does provide an opportunities for the future which will be discussed in the conclusion.

1.4 Causes for Opportunity

Current research into detection of compromised VoIP accounts is a myriad of naive techniques. The literature review will detail research that has been performed using statistical, rule-based, and machine learning techniques for detecting compromised VoIP accounts. The potential solutions section will discuss how this research can improve on the current state.

Machine learning and computational intelligence techniques have been rising in popularity recently. New techniques and computer hardware improvements mean that algorithms that may have been unfeasible in the past are now available to be researched thoroughly. Recent advancements have shown the new potential in machine learning. IBM has created Watson; a cognitive computing system that beat Jeopardy champions *[22]*. Google has been using and improving artificial intelligence techniques in image classification and speech recognition tasks *[14]*.

Chapter II

LITERATURE REVIEW

The research on detecting VoIP fraud is varied in certain aspects while being narrow in others. Detecting VoIP fraud is the main reason for the research being discussed, but other VoIP-related research using similar techniques are also analyzed. The data which is input into VoIP fraud detection systems will be discussed and compared. To compare the data used the attributes taken from call-detail records will be listed to see how CDRs are approached by different researchers. The final sections of this literature review discusses the techniques used in the researched first looking at techniques that do not use machine learning and finally techniques that include machine learning algorithms.

2.1 VoIP Review

Data will be gathered from a telecommunications company that provides VoIP service. The features of their VoIP platform and organization of VoIP entities are an aspect to consider when performing the research. This VoIP service primarily serves business users. This results in a unique organization of VoIP users.

2.1.1 VoIP Platform

The VoIP platform is the SIP-based BroadWorks made by Broadsoft. Each VoIP user corresponds to a single phone user. A VoIP user can have multiple registrations. This allows the user to login from multiple locations at the same time such as having a SIP phone on their desk, a softphone on their mobile phone, and an ATA at home for calling from their

business line. Allowing multiple registrations is also required to allow correctly restarting a VoIP client. If a SIP phone restarts it will attempt to make a new registration resulting in two registrations until the previous registration times out.

The VoIP service is organized into three entities. An enterprise contains groups and each group contains users. The enterprise corresponds to a business customer while users correspond to the individual phone users in that company. Groups are for organizing subsections of the company, but are usually limited to a single group per enterprise. This structure allows for features such as extension dialing for users in the same company. A user in the same company may call another user with administrator defined digits. Intra-enterprise calls are always free of charge, because they are originating and terminating on the same VoIP platform.

A VoIP user may have a direct inward dialing (DID) number that can receive calls from the publicly switched telephone network (PSTN). Alternatively the user may have a private extension that is only accessible from within the same enterprise. Extension-only users receive calls as transfers from a user that has a DID. A popular setup uses an auto-attendant on the company's main telephone number. The auto-attendant allows dialing an internal users extension or searching for a user by name. Extension-only users can make external calls which appear as originating from the companies main telephone number.

2.1.2 VoIP User Access

A VoIP user can login using any client such as a SIP-phone, ATA, or softphone as long as they know the SIP credentials which consist of a username and password. A convention for a username is to use the DID of the VoIP user. If the user is extension-only then the DID of the enterprises main number is used followed by the digits of the extension. This makes the username easy to obtain, but extremely convenient for users and management.

Once the VoIP account is registered calls can be made to any destination. The only action necessary is for the VoIP client to dial the destination number. This can also be automated as in the case of using hijacked VoIP accounts to call premium rate destinations. Directly

using a VoIP account is not the only way to make calls. By altering the configuration of the account calls can be made by using features of the VoIP platform.

Call-forwarding can be configured to send incoming calls to a desired destination. A VoIP attacker with access to configure call-forwarding can route calls to their malicious destination without making a direct phone call with a registered VoIP client. Complex call-forwarding rules can hide the unauthorized configuration by only forwarding specific incoming numbers or limiting the time of day that forwarding is applied.

Conference out-dialing is another method that can be used to make external calls. The VoIP attacker only needs to dial into an open conference and then call out to their desired destination. A conference call with only two participants is practically the same as a call between two participants, but the conference may take additional resources within the VoIP platform.

These features can be configured without knowing the SIP account credentials and can be used without registering a SIP client with the VoIP platform. The VoIP platform has a web portal that users can use to configure their account. The web portal allows changing forwarding settings and creating new conferences. The password for the web portal can be different from the SIP password, but the username is the same. It's accessible from anywhere on the web giving a second endpoint for a VoIP attacker to attempt to usurp.

The voice portal can also be used to change forwarding settings. The voice portal is access by calling the voicemail number and pressing a key. The caller is then prompted to enter their four digit pin number. Once entered the forwarding settings can be updated. This portal has limited functionality, but the password complexity is less than the web portal, because it is limited to a numerical PIN.

The commonality between these methods of accessing an account is that they can be used to originate a call from the VoIP platform and therefore be a target for toll-fraud.

2.1.3 Call-Detail Records

Call-detail records (CDRs) are generated by the VoIP platform for every call made between two endpoints. CDR files are files that contain multiple CDRs. A new file is output every fifteen minutes containing the CDRs for all calls in the last fifteen minute interval. The data and layout of the records are unique to the VoIP platform, but usually contain a similar set of data that is necessary for billing and auditing of calls. The BroadWorks platform used in this research outputs the records as XML files. The relevant CDR fields are described described as follows *[3]*.

- direction — Incoming or outgoing.
- calledNumber — Phone number that was called. This will contain a dialable number.
- startTime — The time when the call was sent or received from the system.
- answerIndicator — Indicates whether the call was answered. Values can be Yes, No, Yes-PostRedirection. The last status means the call was answered after being redirected such as forwarding or voicemail.
- answerTime — Time when the call was answered.
- releaseTime — Time when the call was ended.
- terminationCause — Reason the call ended. Possible reasons include:
  - Normal.
  - User alerted, no answer — Used when the system ends the call because it was ringing too long.
  - Network disconnect – Use when the system ends the call because its duration exceeded a configured value for long calls.
- callCategory — Call category determined by the VoIP system. These are only reported for outgoing calls. Possible categories include:
  - private — Used for intra-group calls

  - local — Local calls.
  - national — National (long distance) calls.
  - internat — International calls.
  - intralat — Intra-lata calls.
  - emerg — Emergency call (eg 911).

- networkCallType — BroadWorks can set a call type based on dial plan rules. There are a set of built-in call types, but more can be added. This is an internal flag for Broadworks, but may have relevant data for call flagging.
- chargeIndicator — Indicates whether the call is chargeable. Values can be: Yes, No, or No Indication.
- releasingParty — Indicated whether the local or remote party ended the call.
- userId — Identifies the VoIP user associated with the call.
- groupId — Identifies the group (and therefore company) of the VoIP user.

2.1.4 VoIP Provider Statistics

The ITSP providing call data for analysis handles approximately 8 million calls per year. Calls are performed by over 1000 VoIP customers with approximately 10,000 active users. They provide services to commercial and residential customers. Commercial customers include various types of business including small business, large enterprises, and call centres. Customers may also engage in activities such as automated calling and many international calls.

2.2 Motivations for Research

There are multiple issues related to VoIP calls that result in similar processing and analysis as toll-fraud detection. Call volume forecasting is used to check whether network capacity can meet future demand. Machine learning techniques that are similar to fraud-detection techniques are used by Mastorocostas to predict future call volumes *[24]*. Forecasting is also used to detect anomalies in expected call volumes to ensure that service is not affected if traffic increases more than expected *[25]*. These call forecasting techniques are relevant to fraud-detection, because they could be used as an indication of fraud when actual call volumes diverge from forecast volumes. A specific type of telephone fraud has been used a the basis for research that yields techniques that may be useful for VoIP toll-fraud detection. Detecting abnormal calling behaviour to find SIM box fraud is one motivation for the research discussed in this literature review *[8]*. This type of fraud is specific to mobile networks, but the techniques used could be applicable to any telephone fraud that is based on calling behaviour. Researchers have focused their efforts in some cases to improve aspects of the algorithms that are used for fraud detection. Extracting the most relevant features from CDRs can be used to improve the inputs to machine learning algorithms *[7]*. The majority of research focuses directly on detecting toll-fraud. Toll-fraud detection research falls into two categories in the research. The first category is fraud committed by internal users of the system. These could be customers that commit fraud against others or customers that try to open a new account after falling into arrears *[1]*. Internal fraud can be caused when a user uses another persons access code to make toll calls *[16]*. The second category of research is toll-fraud committed by external actors. This is the most studied area of research where toll-fraud is defined simply as “illegitimate usage of communication network”*[30]* or as “any activity by which telecommunications service is obtained without intention of paying”*[17]*. All of the analyzed research contributes to a solution for detecting toll-fraud within VoIP systems, but their motivations for that research comes from varied problems.

2.3 Input Data

Systems for detecting toll-fraud require data that is input into the system. The most common method for creating input data is to use attributes of calling behaviour and aggregating the values over a time period. The aggregations may take the form of a mean, sum, moving average, or other statistical summaries. These aggregations of information are referred to as signatures, profiles, or vectors while describing the same basic structure.

2.3.1 Signatures

The are multiple challenges included in using signatures of calling behaviour. The time periods for aggregations of values need to be chosen. The method for updating signatures and when to update with new behaviours needs to be determined *[26]*. The call behaviour signature for an account before any calls are made is another challenge with defining and using signatures. It is difficult to determine what to include in a behaviour signature *[26]*. If there are too many attributes used in the signature it may overwhelm the detection system or include irrelevant information.

Signature Organization

Signatures are organized into relatively few similar structures. The most predominant being a historical and a current signature of behaviour for each VoIP user. The historical signature can be compared against the current signature to determine if they are significantly different *[4]*. Similar to historical and current behaviour is using long-term (extending months into the past) and short-term (extending hours or days into the past) signatures *[21]*. The signature lengths can be configurable to ensure best results. The long-term and short-term signatures are also compared for differences. The techniques for determining if there is a significant difference will be discussed later in this literature review. A historical signature can also be compared against new calls directly without creating a current behaviour signature *[19]*. Signatures may not include historical behaviour at all. Signatures

of current behaviour can be used as the input to detection systems *[18]*. Hilas also uses multiple current behaviour signatures for each user.

New Behaviour

New behaviour must be added to signatures or signatures must be re-created to include the most recent behaviour. A common method to update signatures is to use a weighting function on attribute values *[26]*. An exponentially weighted moving average (EWMA) can be used along with a smoothing factor. Each account has a specific smoothing factor that is based on the average calling rate of the account. The issue being solved is that a smoothing factor that includes 30 days of historical information in the EWMA for a low volume account may only include one day of historical information for a high volume account. If the calling rate is higher the smoothing factor is lower to include more historical information in the results. To calculate the smoothing factor Becker multiplies the current smoothing factor by $1/\log_2(r)$ where *r* is the calls per week for the account *[1]*.

Initial Signature

One method to provide an initial signature is to create a set of base signatures. After the account has made two calls one of the signatures is selected based on the calling attributes. If there has not been any calls made from an account in over a month a new signature is selected based on the next two calls *[1]*. Another option is to simply wait until there are enough calls to build the initial signature. A drawback to this option is that a historical signature may take a long time to include enough calls. This could cause fraudulent call behaviour to be included in the signature. The initial signature may have a minimum number of required calls that must be included in the signature before it can be used for comparison. In the research by Hilas a minimum of eight calls must be present in the signature before it can be considered to be a distinct signature *[19]*.

Signature Structure

This section summarizes the CDR attributes used by each research paper.

Becker

- calls per hour
- distribution of calls by day of week
- distribution of calls by hour of day (calculated separately for weekday and weekend)
- distribution of call duration
- locations called
- most frequent countries called
- most frequent numbers called
- number of calls that are toll calls

[1]

Burge

- user identifier
- location of mobile phone
- chargeable duration
- type of call (international, national, local)
- dialed number
- call start time

[4]

De Lutiis

- call durations

[32]

Elmi

- subscriber identifier
- total calls made by subscriber
- total unique numbers called in one day
- total duration of calls in one day
- total calls made by subscriber during the night
- total duration of calls during the night
- total calls received in one day
- ratio of unique called numbers to total calls
- average call duration

[8]

Hilas (2005)

- number of calls to local destinations
- duration of local calls
- number of calls to mobile destinations
- duration of mobile destination calls
- number of calls to national destinations
- duration of calls to national destinations
- number of calls to international destinations
- duration of calls to international destinations

[19]

Hilas (2008)

- Accumulated weekly behaviour with seven fields:
  - mean and standard deviation calls per week
  - mean and standard deviation of duration of calls per week
  - maximum number of calls
  - maximum duration of one call
  - maximum cost of one call
- Detailed daily behaviour. Number of calls and their duration per day categorized by:
  - destination (national, international, mobile).
  - time of day (working hours, afternoon, night).
- Accumulated daily behaviour:
  - number of calls and duration categorized by national, international, or mobile

Detailed daily behaviour and accumulated daily behaviour signatures are also calculated per week to make five total signatures.

[17]

Hilas (2009, 2015)

- Weekly user behaviour signature contains features:
  - mean call count
  - standard deviation of call count
  - mean duration
  - standard deviation of duration
  - max call count
  - max duration
  - max cost

- Daily user behaviour signature contains features:
  - call count
  - duration
  - toll charges
  - max duration
  - max units

[16], *[18]*

Hoffstadt

- Short-term (Hourly/Daily):
  - count of toll calls
  - call durations
- Long-term profile (weekly and monthly) are summaries of short-term profile using trimmed mean, median absolute deviation, or moving average.

[21]

Taniguchi

- Signature for Artificial Neural Network:
  - average number of calls per day
  - standard deviation and the number of calls made per day
  - maximum call duration
  - maximum number of calls per day
- Signature for gaussian mixture model
  - daily number of calls for office hours, evening, night
  - duration of calls for office hours, evening, night

- grouped by international and national calls

[30]

2.3.2 Other Input Types

There are relatively few inputs that are not based on statistical summary signatures. Graph-based signatures that are distinct from the discussed statistical summary signatures have been used to track the calls between VoIP users and identify patterns. Entropy has been measured and used as a means to monitor changes in the VoIP platform that need further analysis.

Graph-based Signature

AT&T went further in their research on signature-based fraud detection by developing graph signatures *[1]*. These are much different than the statistical summary signatures. Calls were mapped onto a graph with telephone numbers as nodes and calls between numbers as edges. Sub-graphs were extracted by looking at the most called numbers for a node and numbers that most called a node. This sub-graph is called the community of interest signature. It proved to be useful because users that were using AT&T telephone services for fraud purposes had similar communities of interest. This lead the researchers to catch an account that was used as the endpoint for a fraudulent toll-free number. Customers calling the toll-free number would end up being forwarded to a premium rate service. Once the fraudulent service account was shutdown comparisons to its community of interest led to AT&T catching another fraudulent service account. They also use this to catch users that have become delinquent on their bill and attempted to open a new account under a new name. The false identities calling graph will match closely to their previous delinquent account. The call graph technique is useful for catching fraud within the users of the provider.

Entropy

The entropy of the VoIP system and the individual VoIP users can be used as inputs to toll-fraud detection. De Lutiis calculates the entropy of the VoIP system and monitors for changes in the calculated entropy *[32]*. Entropy is calculated for each VoIP account and the system entropy is the average account entropy. The entropy calculation is based on the durations of the calls in which the user is involved over a time period. When an accounts entropy differs from its past entropy further analysis is performed on the users behaviour.

The calculation for entropy for a given VoIP account is:

$$1 - \sum (\frac{D_{ij}}{T})^2$$

Where D_{ij} is the duration of calls between the current user *i* and a destination *j* that occur in the time period *T* *[32]*.

During a trial of the system the historical entropy was gathered over a month period *[32]*. The historical set uses the mean and the standard deviation for the previous month. Each day of the week had its current entropy compared to entropy for that day of the week in the historical entropy. If the current entropy falls more than one standard deviation from the historical mean for that time of day then it sends an alert. They found a good indication of the system working when the current entropy was calculated on a national holiday when the call volume was much lower. Due to the difference from the historical entropy an alert was triggered.

2.4 Non-Machine Learning Techniques

Toll-fraud detection techniques that do not use machine learning algorithms come two main varieties. The most common is based on rules that are applied to statistics from the VoIP platform. Measuring the distance between the statistical summary signatures discussed earlier is another means for toll-fraud detection. Forecasting expected call volumes

using a poisson distribution can be used as a technique to determine if there are anomalies that could indicate toll-fraud.

2.4.1 Rule-based

Burge describes a rule-based approach as *absolute analysis*. The attributes of call data are compared against static values to detect fraud. This may be good at detecting fraud that is so extreme that a high threshold value will not identify many false positives *[4]*.

AT&T has a long history of being a telephone service provider. Their early work on a fraud detection framework included building rules based on threshold values *[1]*. Initially these used a single threshold value of the number of minutes used by a customer. If the threshold was broken within an hour an alert was generated. The problem with this is it was not generalizable to all customers. A small customer and large customer will have vastly different normal number of minutes. Burge also notes that rules are only good at detecting fraud that is so extreme that a high threshold value will not identify false positives *[4]*. This can result in many false negatives when the fraud is not extreme or for levels that are only applicable to large customers. There are many other factors to threshold definitions that will affect their effectiveness such as time of day *[26]*. AT&T resolved this by creating more threshold rules that were based on different values. This allowed finer-tuned thresholds according to customer size, time of day, day of week, and location called. This lead to having 30 000 thresholds defined and still alerting on too many false positives *[1]*.

Manually configuring the rules is a problem for rule-based solutions. Rules must be developed that can generalize to the entire user base, but also be specific enough to catch fraud. New fraud techniques are developed and the rules need to be updated to target those techniques as well *[26]*. The human resources required for managing rules can become burdensome to the VoIP provider. Having the over 30 000 different thresholds as AT&T had would quickly become overwhelming and require a new strategy for fraud detection.

There are cases where rules can be successfully combined with other detection methods.

A simple rule based on location can check if a users location between calls is further than could possibly be traveled in that time. If the distance is further than possible, then it is an indication that fraud could be occurring *[21]*. At the very least it is an indication that multiple people are using that account which could be against the VoIP providers terms of service.

Rules may also be used to identify fraud patterns. They can compare sequences of SIP messages to known fraud patterns to identify attacks such as if one IP address sends registration attempts to multiple users in a short period of time *[21]*. Hoffstadt also identifies the applicability of a blacklist that denies calls based on blacklisted values of call attributes. This could be used to block calls based on the source IP address or destination phone number.

False positives when detecting fraud are likely to happen when using a rule-based strategy. Rules identify changes in usage or match known fraud patterns, but this behavioral change could be normal to the user *[26]*. The strict nature of rules makes them easy to define for a specific user, but generalizing the rule to fit 100% of the VoIP users is unlikely. Even a small percentage of false positives can affect a significant number of users.

2.4.2 Poisson Distribution

A poisson distribution can be used to forecast call volumes. Mata uses the number of active calls with a five minute interval Poisson distribution *[25]*. Two different measurements of active calls is used by Mata. The first is the current number of active calls. The second is the number of calls which do not end in the time interval. This is used to forecast the expected call volume. If the number of active calls differs from the expected value then it can indicate a problem that needs to be investigated. This research is not ideal for detecting toll-fraud. The active calls during the night were excluded, because the large difference in day and night volumes were not suitable for Poisson distribution. As described in the research this method is useful for detecting a possible overload of the VoIP system when the call volumes increase more than expected.

2.4.3 Signature Distance

AT&T compares incoming calls directly with signature to identify fraud *[1]*. When a call is made it is compared against both the VoIP accounts signature and a fraud signature. If it is closer to the fraud signature then a fraud score is incremented on the VoIP account. Once the fraud score reaches a critical level an alert will be generated. The fraud score of each account decrements back to zero over time. If the signature is closer to the accounts signature then its behaviour is incorporated into that account signature. Unfortunately the measurement to compare similarity between the signatures is not given by Becker.

In contrast to a fraud score most signature-distance techniques is a threshold that acts on the distance between incoming calls (either directly or their a short-term signature) and the accounts historical signature. Hoffstadt builds short-term and long-term profiles. The distance between short-term and long-term profiles is calculated. If the distance is too high it is used as an indication of fraud. The distance measurements used are z-score, Median Absolute Deviation, and Hellinger distance *[21]*. Burge also uses the Hellinger distance to compare signatures, but compares sequences of signatures *[4]*. Burge compares the sequences by summing the Hellinger distance of each pair of signatures in the two sequences. Hilas measures the similarity between signatures with a variation of r-contiguous bits algorithm *[19]*. The algorithm is adapted to allow comparing sequences of signatures and to consider durations to be equal if they are within a certain range of each other. Comparing the sequence of signatures can allow treating the set of signatures as a probability distribution and allows applying decay calculations in the distributions *[4]*.

To determine whether the signature comparisons would detect variations in behaviour Hilas compared signatures of users. When comparing the set of signatures between two users the comparison should determine that they differ. Hilas found that accounts could be determined to be 80% different after at least eight signatures could be used in the test sequences *[19]*.

2.5 Machine Learning Techniques

Machine learning algorithms are popular for toll-fraud detection. They most often use summary signatures as inputs. Artificial Neural Networks are the most researched algorithm used with various configurations. Feature extraction methods have been researched to provide better inputs to support vector machines. Clustering algorithms are used for developing rules to identify fraud and for grouping similar calling behaviour such as fraud behaviour. Some less researched techniques include gaussian mixture model and bayesian belief networks.

2.5.1 Artificial Neural Network

Artificial Neural Networks (ANNs) have been the focus of much of the toll-fraud detection research. They have many advantages over other detection techniques. The types of Neural Networks used will be discussed along with weight tuning, training, and testing used for the ANNs.

Advantages

There are multiple advantages to using Neural Networks. They are especially suitable for noisy data *[26]*. This is useful for VoIP calls because the calls may not have clear cuts between fraudulent and non-fraudulent behaviour. Neural networks also have the advantage of detecting fraudulent behaviour that has not been seen before *[26]*. This is an improvement over rule-based strategies because they are based on defining rules that detect known fraud behaviours, while the Neural Network can adapt to any changes in behaviour. Artificial neural networks and machine learning techniques in general do not require knowing the underlying distribution of variables *[26]*. This is and advantage over techniques such as the Poisson distribution forecasting discussed above.

Types of Artificial Neural Networks

The majority of research into using Artificial Neural Networks for toll-fraud detection uses Feed-Forward Neural Networks. The only notable exception is a Recurrent Fuzzy Neural Network that is used for forecasting. Mastorocostas uses a Recurrent Fuzzy Neural Network with the goal of providing forecast information for planning expansion and providing insight into future call volumes *[24]*. This may also be useful in a toll-fraud detection system by verifying whether the forecast call volume matches actual call volumes and investigating discrepancies. Self-organizing maps (SOM) are an ANN technique that has been discussed as a possible approach to fraud detection, but no research has been found that uses SOM for fraud detection *[26] [21]*.

Inputs

The inputs into the ANN consists of the signatures described above. The values in the signatures can be transformed to be more suitable for ANN input. Hilas transforms the variable values to make the average value in the set equal to zero and the variance maximum one *[17]*. Hilas also applies principal component analysis (PCA) to the distribution of signatures before using them as input.

Training

Back-propagation algorithms are the preferred method for training ANN. Elmi uses back-propagation with momentum and learning rate terms that affect the rate at which the weights change during training *[8]*. The resilient back-propagation algorithm is used by Hilas and a variation of the same algorithm is used by Mastorocostas *[17] [24]*. Burge uses Levenberg–Marquardt algorithm for back-propagation training of the ANN *[4]*. Both Elmi and Burge use the sum squared error for the minimization target for their back-propagation algorithms. Burge also uses cross-validation during training to optimize the weights of the ANN *[4]*. As an alternative to back-propagation Taniguchi uses quasi-newton optimization *[30]*.

Optimization

Cross-validation is one technique that is used to optimize the parameters used for the ANN. Elmi uses 10-fold cross-validation to test the performance of multiple ANNs *[8]*. Verifying the parameters including the number of hidden layers, learning rate, and momentum of the Neural Network resulted in 240 different ANNs that were compared using 10-fold cross validation. The best performing ANN had a 98.71% accuracy. This ANN had two hidden layers of five nodes, learning rate of 0.6, and momentum term of 0.3 *[8]*.

Evaluation

The receiver operating characteristic (ROC) curve is a popular choice for evaluating the performance of Artificial Neural Networks. It is used by Elmi, Taniguchi, and Hilas to test their results *[30] [17] [8]*. The root mean squared error (RSME) is also used by multiple researchers. Mastorocostas uses RSME along with mean absolute percentage error and Theil's U-statistic for evaluation *[24]*. Elmi uses RSME, ROC curve, as well as precision and recall for evaluation.

2.5.2 Support Vector Machine

Although Support vector machines (SVM) have not appeared directly as a toll-fraud means, there is research into providing feature extraction methods for VoIP call data that will provide optimal inputs to SVM that detects fraud. Bivariate statistics (BS), information gain (IG), principal component analysis (PCA) are popular feature extraction methods. These are compared to a new feature extraction method GPCA (described as a combination of PCA and IG) developed by Donga to extract features from call-detail records for use as inputs to SVM *[7]*. The results indicate that when the results of the new feature extraction algorithm are used as the inputs to SVM for fraud detection it performs better than the other three algorithms. The results from the SVM are evaluated based on sensitivity and specificity.

2.5.3 Clustering

Clustering is an unsupervised machine learning technique that can be used to group calling behaviour into clusters of fraud and non-fraud behaviour. There are various clustering algorithms that have been used to classify VoIP calling behaviour when looking for indications of toll-fraud.

Hierarchical agglomerative clustering technique was used by Hilas to confirm that the clusters formed match the classifications from an Artificial Neural Network. Two distance calculations are used for the clustering: euclidean distance and correlation between objects. The dendrograms for the clustering shows that correlation between objects provides clusters that more distinctly identify fraud. Of the top four clusters three contained only fraud cases which were 70% of the total fraud cases *[17]*.

K-means clustering was also analyzed by Hilas with a comparison between using euclidean distance and correlation as a distance measurement. K-means using correlation was accurate in 84.8% of cases. *[18]*.

Decision trees can be used to cluster the data and provide rules on when attributes will put the data into a cluster. Hilas used C4.5 decision trees as part of an expert system developed to detect fraud *[16]*. The first tree developed uses weekly summarized call information. The attributes mean duration, standard deviation of duration, and mean call count as inputs into a decision tree resulted in two rules being developed based on the six leaves of the tree. Hilas also developed a decision tree based on the daily call behaviour. This tree resulted in 10 leaves using three attributes of the data. Two rules were extracted from the decision tree to be used in a rule-based strategy. Although this leads to a rule-based strategy decision trees help alleviate the drawbacks by providing an automated method for determining rules. Decision trees also provide an advantage because the results are easily interpreted by looking at the branching points to develop rules *[26]*.

2.6 Conclusion

Much of the research into toll-fraud detection uses statistical summaries to provide inputs to algorithms. Unfortunately the time period used for current behaviour summaries may require hours or days of usage before the signature is available for fraud detection. The algorithms found used a minimal number of CDR attributes. Modern VoIP platforms provide fine-grained detail into calling behaviour, but this detailed level of information is not being used to its potential. This may have to do with historical processing constraints associated with large amounts of data and machine learning algorithms. Overall the research indicates that multiple methods should be used to generate multiple indications of fraud. Using multiple indicators of fraud is concretely broken down by Rehabi into three classifications *[26]*:

- Primary indicator — indicates fraud without needing other indications.
- Secondary indicator — provides information that can indicate fraud, but by itself cannot prove fraud.
- Tertiary indicator — provides supporting information to the other indications.

These classifications can be used to build a case around a pattern of behaviour to identify it as fraudulent or non-fraudulent VoIP usage.

Along with multiple methods for indicating fraud Becker emphasizes the role of humans in the fraud detection process. The application of expert knowledge is important when verifying alerts as fraud events. The tools that AT&T develop are used to support the people that investigate the fraud cases as opposed to being the only point of detection. Visualization of calling behaviour and fraud enables the human verifiers to easily see patterns that can be identified as fraud by a person where the detection algorithm may only have enough information to give indications of fraud *[1]*.

Chapter III

RESEARCH METHODOLOGY AND DESIGN

This section describes the design of the data pipeline. The first design step is gathering and processing the data from the database. Next is defining the data vector structure that is to be passed into the algorithm. And finally designing the algorithm itself.

3.1 Processing Data

Ingesting CDRs includes multiple steps that are combined as an input pipeline. The CDRs proceed through the steps in the pipelining including parsing XML records, sanitizing values, anonymizing records, and storing them in an accessible data structure.

3.1.1 Gathering Data

The CDR files generated from the ITSP's VoIP platform have been archived since March 2013. Call data between March 2011 and March 2013 is stored in a relational database after having been parsed from CDR files by the ITSP. Calls before March 2011 were in a legacy system that did not record any call information. That system was the target of large scale toll-fraud, however no records are available to analyze that fraud event.

The call data before March 2013 is missing some information that would have been included in the CDR files. The ITSP excluded calls that were: unanswered, internal to a customer such as calls from a private extension to another private extension, or calls

where the destination could not be determined based on either the calling or called number depending on whether the call was inbound or outbound. The CDRs during this time period are not available.

The call data after March 2013 is stored in XML files. The VoIP platform uploads these XML files to the archive server using FTP whenever a new CDR file is complete which is defined as every 15 minutes. The calls that ended within the last 15 minutes are included in the file. The files were downloaded as-is from the archive site to the workstation where parsing would occur. At the time of downloading the latest call in the archives is from December 24th, 2015.

3.1.2 Data Storage Setup

A relational database (Postgresql) is setup to store the parsed call records. This database will be used to store the call data before anonymization. Only after the data is anonymized can it be exported from the ITSP. Tables were created to store calls, destinations, toll rates, and service invocations. The tables are:

- calls — Stores parsed call records. This table contains the majory of columns related to a call.
- destinations — Stores the destination lookup data provided by the ITSP. This contains the country code, dialed digit prefix, destination name, and toll rate. Each call row is related to a destination row.
- rates — Stores the ITSP toll rates. Each rate is associated with multiple destinations.
- countries — Stores a list of countries that includes basic information such as the country code. Each destination is associated with a country.
- service_invocations — Contains the services that were used for a specific call. Each service invocation is associated with a single call.

Indexes were created on the tables to provide faster querying of data based on the most used columns such as user ID, answer time (for ordering), and destination. After storing

the call data the calls table is 40GB in size.

3.1.3 Parsing and Sanitizing Call Data

Pre March 2013 data was exported from the relational database into a CSV file. That CSV file was then imported into the relational database that would be used during tagging and sanitizing values.

The call data prior to March 2013 was exported from its current location and then imported into the new database.

An software application “CDR Parser” was created to process the CDR files containing call data for calls after March 2013. The CDR files are XML documents that follow Broadworks specification for their CDRs.

Each file contains many call records. The CDR Parser reads each call element from the XML document iteratively. The CDR files may be very large if many calls are made in the duration that the CDR file covers. A file that is several gigabytes in size may consume too much memory if read in its entirety during processing. Instead the XML elements for calls are read in sequence. After each call is processed the previous call is removed from memory.

The CDR processing involves several steps:

- XML Element for call is read.
- Call records that are not “normal” are skipped. This includes call records generated during an ongoing call. This may happen if a call duration is especially long or if the call fails over to another VoIP server in the cluster.
- Phone numbers and dates within the CDR are normalized to a standard representation. The dates will be stored in the database as a timezone-aware timestamp.
- The destination for the external number is queried.

- The call record is saved as row in the database. Every field from the CDR is stored in the database row.

Finding the destination for the external number takes additional processing. The external number is the calling number for inbound calls and the called number for outbound calls. The destination can be categorized based on the phone number format into the following categories:

- Internal call between private extensions. Eg 2000.
- International. Eg 011-1-555-555-0100.
- North american. Eg 1-555-555-0100.
- North american local. Eg 555-555-0100.
- North american local to NPA of NANPA calling plan. Eg 555-0100.
- Three digit special number. Eg 411.
- Star code. EG *98.

The ITSP has a database of destinations that is used to lookup the destination for a particular phone number. Knowing the category of destination increases the effectiveness of querying the destination. The destination database includes the country code and a digit prefix. Each country can (and does) have multiple digit prefixes assigned to it. The digit prefix is matched to the beginning digits in the destination phone number. The longest digit prefix that matches the start of the destination phone number is used for the destination.

The destination database uses “1” as the country code for US and Canada. Other North American countries use “1” plus the NPA for their country code. For example the Bahamas uses the country code “1242”. The digit prefix for NANPA numbers includes the NPA. Destinations in Canada may have the country code “1” and the digit prefix “705”. This is the case for all North American countries even if the NPA is part of the country code. The Bahamas has the country code “1242” and digit prefixes that begin with “1242”.

The destinations are generally fine grained. The digit prefix may identify geographical areas, service providers, mobile service, or any other organization that the country has designed. A small country such as the Bahamas has 76 digit prefixes in the destination database.

An example of an SQL query used to find the destination for an international phone number is:

```
SELECT destination_id
FROM destinations d
WHERE destination_id = (
    SELECT destination_id
    FROM destinations
    WHERE char_length(digit_prefix) = (
        SELECT max(char_length(digit_prefix))
        FROM destinations
        WHERE substring(
          international_destination_number
          FROM 1 FOR char_length(digit_prefix)
        ) = digit_prefix
        AND country_code != '1'
    )
    AND SUBSTRING(
      international_destination_number
      FROM 1 FOR char_length(digit_prefix)
    ) = digit_prefix
    AND country_code != '1'
```

If the destination cannot be found for outbound calls the CDR Parser raises a fatal error and the processing stops. This is important because the service provider must know the destination of a call in order to bill it accordingly. If the destination cannot be found for inbound calls the call will be saved with no destination. This may occur when an inbound call is from a private number.

Additional fields were generated during parsing to make querying more efficient. The duration was calculated by taking the difference between the answer time and the release time of the call. The toll charge was calculated by taking the duration rounded up to the nearest minute multiplied by the toll rate per minute. The country code and NPA of the user was stored. The duration of time between the start time of a call and the start time of the previous call by the same user is stored as a new field “start_time_delta”.

3.1.4 Tagging Data

Fraud is defined by fraud events that have a start time, end time, and a user ID. These events are used to indicate that a sequence of calls contain fraud rather then marking individual calls as fraud. The Recurrent Neural Network to be used takes sequences of calls and a tag for the sequence as input. The tag is related to the entire sequence. Any call for the user that has a start time within the fraud event boundaries is tagged as being made during a fraud event.

To determine past fraud events I searched my own emails in-depth as I have been notified of every fraud event at the ITSP. Several fraud events were identified along with the time period and users affected. The fraud events are described in greater detail in the analysis section.

After the calls were parsed and stored in a database the calls rows that were made during a fraud event were tagged as such by adding a column to the calls table. The data in the table is not anonymized yet making the tagging easy to apply against the specific users calls.

3.1.5 Anonymizing

The call records in the table need to be anonymized to prevent identifying particular customers or users of the ITSP’s VoIP service. Before anonymizing generic information was gathered to give context to the fraud events. For example a customer may have been iden-

tified as a call centre with many users or an international company that may call many international destinations. Those descriptions do not identify customers because the ITSP has many customers that meet those definitions.

The CDRs include identifying information such as customer IDs, user IDs, customer telephone numbers, users' full names. The destination and source telephone numbers may also identify users. The full list of CDR fields with potentially identifying information is:

- Service provider (equivalent to a Customer ID)
- User phone number
- Group phone number
- Calling phone number
- Calling presentation indicator
- Received calling phone number
- Dialed digits
- Called phone number
- User ID
- Other party name
- Other party presentation indicator
- Group name
- Original called phone number
- Redirecting phone number
- Conference owner
- Conference owner phone number

Telephone numbers present a challenge in anonymization because they can be used to identify a person, but also must be available to find multiple calls to a destination and

to determine where the destination is. To provide these two functions two values will be generated from each source or destination phone number. The entire phone number will be hashed into an opaque value. This will allow analyzing multiple calls to/from the same phone number. The second value will be the destination of the call. This is a reference to the destination table provided by the ITSP.

The destinations are not fine-grained enough to identify the user or external party. A few examples of destinations are:

Table 3.1: Example destinations.

Country Code	Digit Prefix	Destination Name
1	705	Ontario - Northern
1	800	Toll-Free Calling
44	447467	United Kingdom - Vodafone Mobile
44	447403	United Kingdom - H3G Mobile
57	5759125	Colombia - Baranquilla
57	5759124	Colombia - Baranquilla

There is also a class of phone numbers in the NANP that does not follow geographic constraints. The most recognizable are toll-free numbers such as those beginning with 1-800. These numbers follow the same method, because the destinations lookup includes special NANPA phone numbers such as 1-800 numbers and 3-digit numbers such as 911.

All identifying fields will be converted to an opaque value using a one-way hash function. This allows identifying calls made by the same user or to the same destination without identifying the user or callee.

During CDR parsing the user's country code and NPA was stored separately from their phone number. This allows identifying the location of the user without identifying them.

Table 3.2: Example field anonymizations.

Field	Raw Value	Anonymized Value(s)	Anonymization Method
Customer ID	1000	b6de3e94...	Hash function
User ID	4165551234@example.com	01be53f2...	Hash function
User phone number	+14165550000	1416, 7cedfdfc...	Truncate after NPA, hash function
External phone number	+14165551111	f6675c0e...	Hash function
External Phone Number (non-NANPA)	+4403069990000	f6bf76d8...	Hash function
External Phone Number (special NANPA)	+19005550000	889fa4bf...	Hash function

After anonymization the original values of the data are removed completely from the database.

3.2 Data Design

3.2.1 Splitting Data

The data must be split into training, testing, and validation sets. There are important constraints to consider when making these splits. The Recurrent Neural Network algorithm requires the call data to be in chronological order. Therefore the data should be ordered by the time the call was made. Each data set requires a mix of fraud and non-fraud calls. To ensure that both fraud-tagged an non-tagged calls are included only calls from users that have been the victim of a fraud event are included.

All calls that are not during a fraud event are included in each dataset. Calls tagged as occurring during a fraud event are split between training, validation, and testing datasets.

The training set will contain 3/4 of the calls made during each fraud event. Validation and testing will each contain 1/8 of the calls during each fraud event. The tagged calls appear in only one dataset.

The following diagram shows how a fraud event for a user is split up into three datasets. This is done for each user and each fraud event.

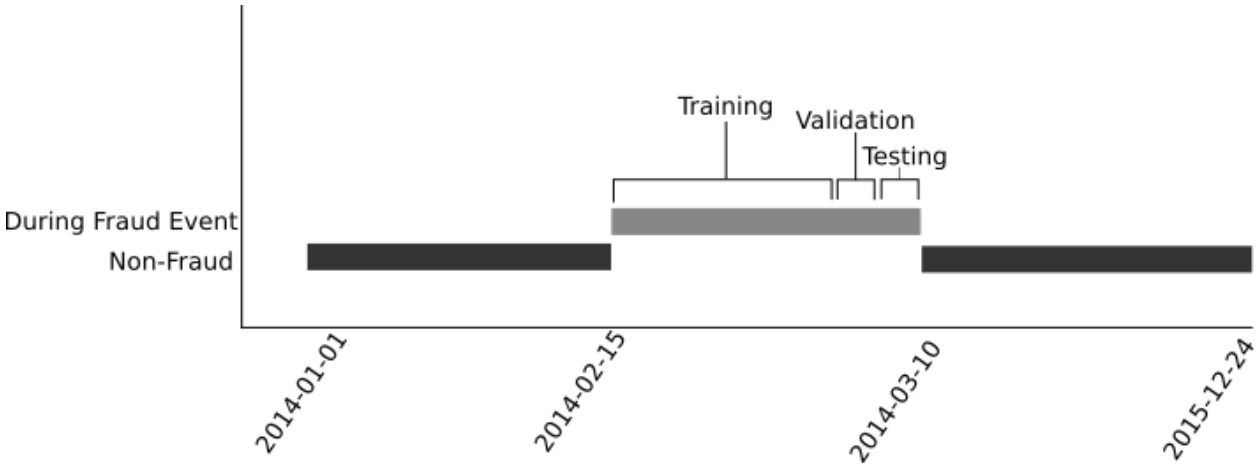

Fig. 3.1: Diagram showing how fraud events are split between training, validation, and testing data sets.

An example of the SQL query used to extract the dataset of tagged calls can be seen here:

```
SELECT
  ca.*
FROM calls ca Fraud
  LEFT OUTER JOIN fraud f
    ON    ca.start_time between f.start_time
      AND (f.start_time + (f.end_time - f.start_time)/8 * 6)
      AND ca.user_id = f.user_id
WHERE
  ca.during_fraud_event = True
```

The start time of a call is present for each call. This is used for ordering the calls in chronological order. Now a group of calls may be missing from the users sequential call

history, because it is in another dataset. This means there could be a large gap between the start time of sequential calls. Therefore the start time must not be used as a feature when training the Recurrent Neural Network algorithm. When the calls were parsed and ingested into the relational database a start time delta field was saved on the call record. This gives the difference in start times between two sequential calls made by the same user. This start time delta value along with the chronological ordering of the calls and a categorization of the time of day obviates the need to use the start time as a feature.

3.2.2 Input Data

The input data comes from the fields defined on the calls data stored in the relational database. This includes all fields from the CDRs as well as some calculated fields.

After retrieving all of the calls in the training dataset the fields without unique values are removed. The following fields are left in the input call records:

- User Country/NPA — Contains the User's country code and NPA.
- Direction — Specifies whether the call is incoming or outgoing.
- Answer indicater — Flag indicating if the call was answered.
- Start time delta — The time between sequential calls.
- Start time category — A categorization of the call start time. The categories are:
  - Morning — Between 0:00 and 8:00.
  - Day — Between 8:00 and 18:00
  - Evening — Between 18:00 and 22:00
  - Night – Between 22:00 and 0:00
- Termination cause — Reason for the call end.
- Call category — Call category from CDR.
- Network call type — Call type from CDR.

- Charge indicator — CDR field indicating the call has a toll charge.
- Releasing party — Flag indicating whether the user or the other party ended the call.
- Redirecting reason — If the call was transferred this is the reason for the transfer.
- Country code — Country code of external party.
- Rate — Per minute rate.
- Duration — Duration of call.
- Charge — Total charge of the call. Note that this is not the same as rate * duration, because the charge uses the duration rounded to the highest minute in its calculation.

Vocabularies

Each feature needs to be converted to a value amenable to matrix multiplication. Values containing text data will be converted to a vocabulary formatted as a one-hot vector. Numeric values will be standardized as the number of standard deviations away from mean.

The vocabularies are codified by assigning an index of a vector to each distinct word value. For example the direction could either be “Incoming” or “Outgoing”. Incoming would be assigned the index 0 and outgoing the index 1 in a two-element vector. For each call record the element at the words index will be set to 1 and all other values set to 0.

The following diagram shows a one-hot vector where the “1” value is in the index position for “Outgoing”.

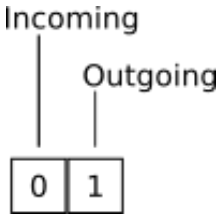

Fig. 3.2: One-hot vector showing “Outgoing” as set.

If there were more possible values the vector would be longer. The one-hot vector for

country codes would include an index for each country option resulting in a length of around 200 elements. Only values actually in the dataset need to be included in the vocabulary.

Continuous Variables

The continuous variables in the input data are: start time delta, rate, duration, and charge. The start time delta and the duration are converted from a time interval to a number of seconds. Each of the variables is then processed using the same standardization method.

The variable is converted to a value representing it's difference from the mean as a multiple of standard deviations.

$$X = \frac{x - \mu}{\sigma}$$

This brings the variable values into a space where a variable with high numbers (such as duration or start time delta) does not have a greater impact than variables with small numbers such as rate. Using this standardization allows high values to still impact the data set. Fraud events often result in high charges and therefore using a normalization technique that results in a variable between 0 and 1 may squash all but the very highest values into a very small space.

Input Vector

A new vector is created by concatenating the vectors for each vocabulary. The standardized variable are then appended to the vector. The following table shows the number of elements each feature takes within the input vector.

Table 3.3: Input vector fields.

Vector Length	Field Name	Type
3	Direction	Vocabulary
4	Answer indicator	Vocabulary
5	Start time category	Vocabulary
12	Termination cause	Vocabulary
9	Call category	Vocabulary
15	Network call type	Vocabulary
3	Charge indicator	Vocabulary
4	Releasing party	Vocabulary
240	Country code	Vocabulary
1489	Destination name	Vocabulary
1	Start time delta	Standardized
1	Rate	Standardized
1	Duration	Standardized
1	Charge	Standardized

This vector is generated for each call in the dataset. The vectors may be processed to remove superfluous attributes. Any element in the vector that has the same value for all calls is removed. For example if no one calls a specific destination in the vocabulary then the element for that item is removed. This reduces the length of the vector making calculations faster. The list of attributes removed during training must be saved so that the same attributes can removed when validating, testing, and using the algorithm.

During parameter testing the exact vector presentation will be adjusted. Section 9.1 will discuss the full and final input vector format.

Input Sequences

The input vectors are arranged into sequences. Each sequence contains 10 call vectors. These are in chronological order. The first sequence contains the first 10 calls. The second second will contain 10 calls starting after three calls. This overlap ensures that most combinations of chronological ordering are passed into the algorithm. The following diagram shows the first five sequences that will be passed to an algorithm. The calls are number in chronological order. Each call is an input vector as defined above.

Call 01	Call 02	Call 03	Call 04	Call 05	Call 06	Call 07	Call 08	Call 09	Call 10
Call 04	Call 05	Call 06	Call 07	Call 08	Call 09	Call 10	Call 11	Call 12	Call 13
Call 07	Call 08	Call 09	Call 10	Call 11	Call 12	Call 13	Call 14	Call 15	Call 16
Call 10	Call 11	Call 12	Call 13	Call 14	Call 15	Call 16	Call 17	Call 18	Call 19
Call 13	Call 14	Call 15	Call 16	Call 17	Call 18	Call 19	Call 20	Call 21	Call 22

Fig. 3.3: First five input sequences showing the calls within them.

These sequences are generated for the entire dataset of calls. Each sequence has a corresponding flag indicating if the sequence contains calls made during a fraud event for that user. The tag information is stored in a separate sequence where the index number of the tag is the index number of the sequence it is tagging.

The number of input vectors per sequence and the number of elements to stagger the overlap are configurable. These parameters are to be tested in various combinations to optimize the algorithm.

3.3 Algorithm Design

Recurrent Neural Networks (RNN) are applied to label a sequence of input data. *[15]* The sequence of input vectors for a given VoIP user account will provide the necessary input to a Recurrent Neural Network. The context of the call vector within a sequence of the users call history may reveal more information about the status of the user account. Using entire sequences of call history provides this context to the algorithm. The entire input sequence will be classified as being part of a fraud event or not being part of a fraud event. This will enable identifying a period of time as containing unauthorized fraud activity.

There are variations of the Recurrent Neural Network algorithm. The Long Short-Term Memory (LSTM) algorithm is a specific RNN algorithm that has proven itself for time-series events. It solves some issues seen in previous RNNs when the time between events may contain delays. *[15]*

The unpredictable times of calls may be a good match for LSTM. RNNs can be difficult to train because they are not acyclic like feed-forward Neural Networks. Small changes within an input in the sequence can blow up to have a greater effect on the output when it loops back through the algorithm nodes. *[29]* Truncated backpropagation can be used to effectively train the RNN and take into account the timesteps between the input sequences events. *[29]*

3.3.1 LSTM Design

LSTM uses gates to determine which information to keep and to forget on each timestep. The “add” (or “input”) gate determines which information is added from the current input and previous hidden state. The “forget” gate determines which information to drop from the previous state. The “output” gate determines which information is output from the current state. *[20]*, *[13]*

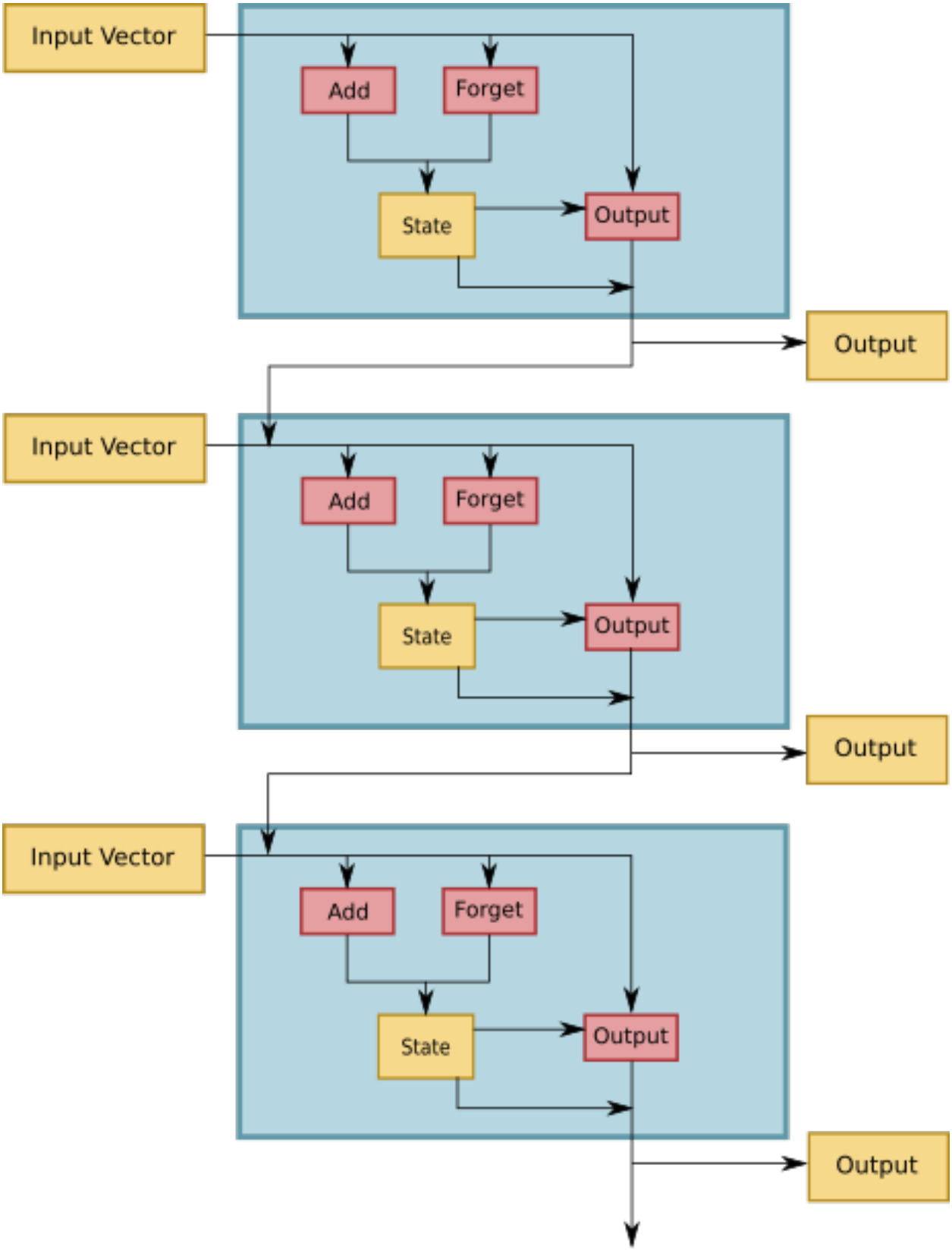

Fig. 3.4: Diagram of LSTM gates used in sequence.

An LSTM node is defined by a several variables and functions to create the variables.

X_t is the input at the current timestep.

W_i is the trained input gate weight for X_t.

h_{t-1} is the hidden state from the previous timestep.

U_i is the trained input gate weight for h_{t-1}.

i_t is the result of the input gate.

W_f is the trained forget gate weight for X_t.

U_f is the trained forget gate weight for h_{t-1}.

f_t is the result of the forget gate.

W_c is the trained context weight for X_t.

U_c is the trained context weight for h_{t-1}.

$\widetilde{C}_t$ is the potential context for the current timestep.

C_{t-1} is the context state at the previous timestep.

C_t is the new context state for the current timestep.

V_o is the trained output weight for C_t.

O_t is the result of the output gate.

h_t is the hidden state at the current timestep.

$$i_t = \sigma(X_t W_i + h_{t-1} U_i)$$

$$f_t = \sigma(X_t W_f + h_{t-1} U_f)$$

$$\widetilde{C}_t = tanh(X_t W_c + h_{t-1} U_c)$$

$$C_t = i_t * \widetilde{C}_t + f_t * C_{t-1}$$

$$o_t = \sigma(X_t W_o + h_{t-1} U_o + C_t V_o)$$

$$h_t = o_t * tanh(C_t)$$

The hidden state h_t and the context C_t are used in the next timestep of the node.

The flow of the variables through the node is visualized in the following diagram. The input X_t and hidden state h_{t-1} are used to create the add and forget gates. The potential state is also created from the same two variables. These are combined with the previous state to create the next current state in the node. The output O_t is created from the input X_t, the previous hidden state h_{t-1} and the current context C_t. The current hidden state is created from the output O_t and the current context C_t. The hidden state and the current context will be used in the next iteration of the node.

The next diagram shows the formulas for generating each variable.

Each variable and gate combination has its own trained weight. As can be seen in the application of W_i, U_i for the input gate; W_f, U_f for the forget gate; W_c, U_c for the new potential context; W_o, U_o, and V_o for the output gate.

i_t is applied to the new potential context $\widetilde{C}_t$ to determine which information to add to the new context C_t while f_t is applied to the previous context C_{t-1} to decide which information to forget.

The variables for the input gate, forget gate, and potential context depend on only the input at the current timestep and the previous hidden state. Therefore these variables can be calculated in parallel when the algorithm runs. The algorithm will be executed on a modern GPU that provides extremely efficient parallel processing.

3.3.2 Training

Data from the training set will be used to train the algorithm. Each sequence of 10 input vectors will have a corresponding binary label which indicates whether the sequence contains calls that were made during a fraud event occurring on the user account. The training data will be iterated over many times (epochs) while training the LSTM. Training will attempt to minimize the categorical crossentropy error *[31]*.

$$(-\sum \log \hat{y})/N$$

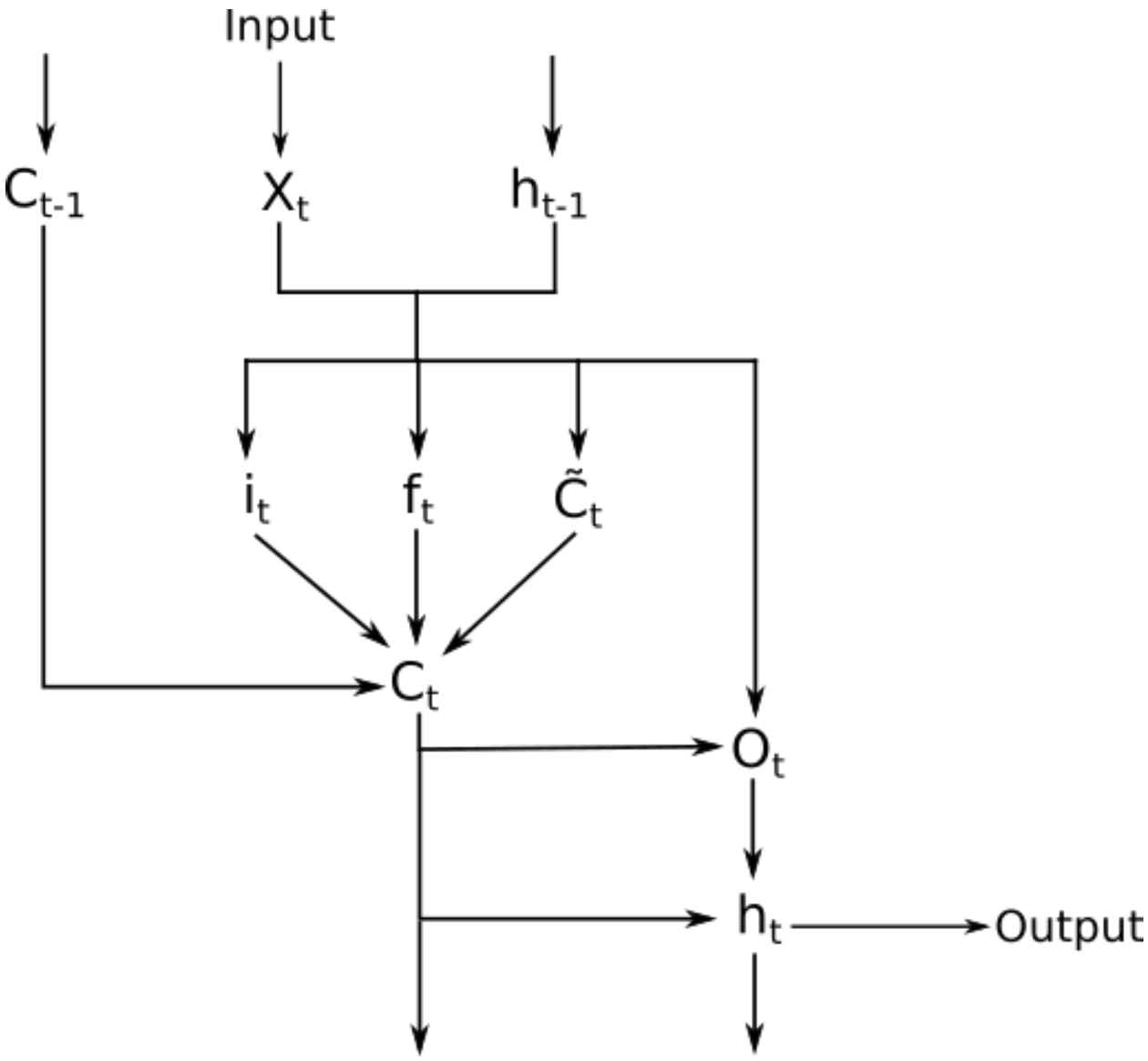

Fig. 3.5: Diagram of variables used in a LSTM node.

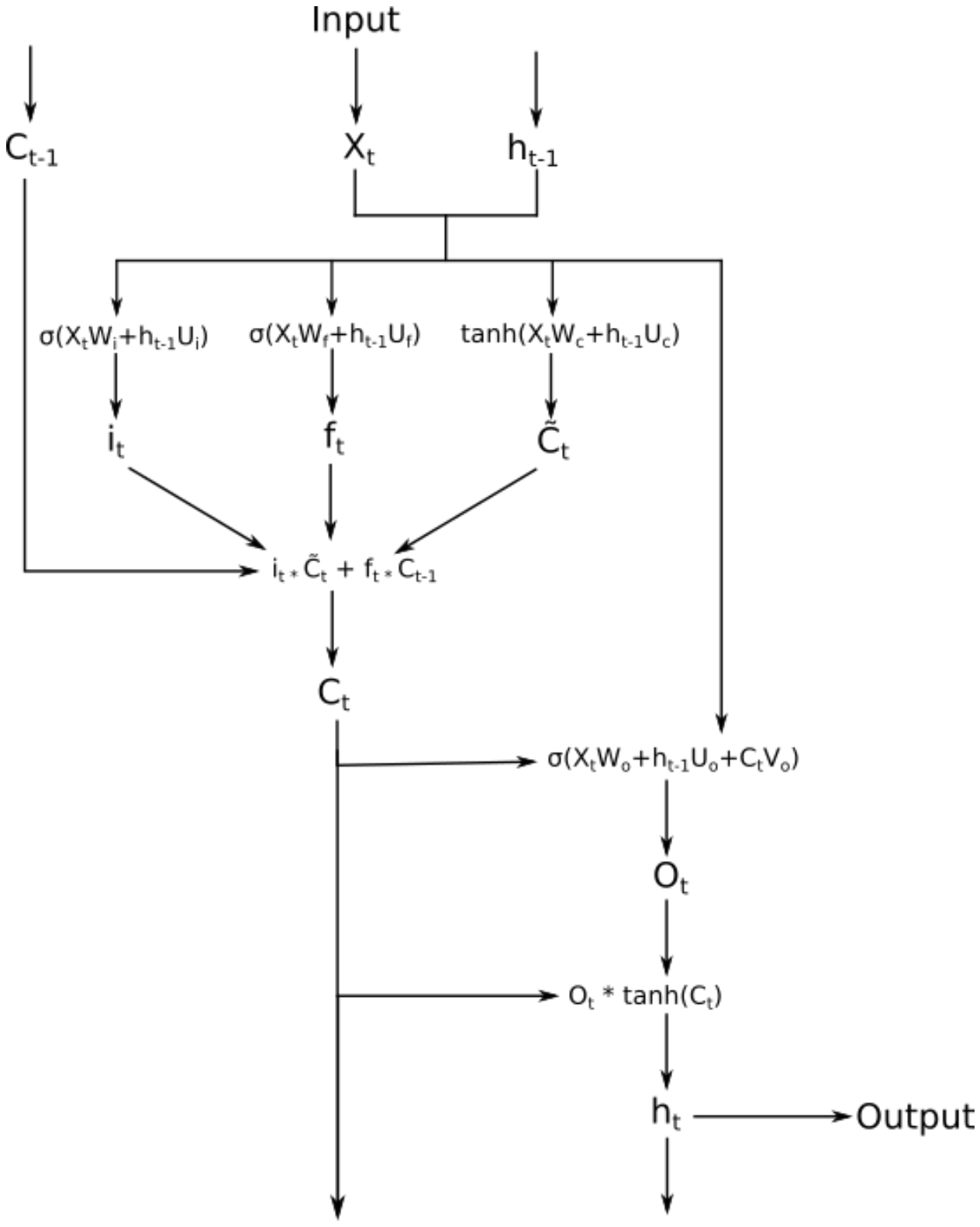

Fig. 3.6: Diagram of functions used in an LSTM node.

y is the distribution of values from the data, $\hat{y}$ is the distribution of predicted values, and N is the number of samples.

Below is an example function to calculate cross entropy.

```
def crossentropy(x, y):
    N = np.sum(len(y_i) for y_i in y)
    loss = 0
    for i in np.arange(len(y)):
        # Predict values
        o, s = self.forward_propagation(x[i])
        # Distribution of correct predictions
        correct_predictions = o[np.arange(len(y[i])), y[i]]
        # Calculate loss and update running total
        loss += -1 * np.sum(np.log(correct_predictions))
    loss = loss / N
    return loss
```

RMSprop is used as an optimizer to determine the updates to the weights during training. It dynamically adjusts the learning rate based on the running average of the recent gradients. *[27]*

3.3.3 Execution

The LSTM will be defined and run using Theano. Theano is a Python library that can be used to define graphs of operations. These are compiled to be performed efficiently on either CPU or GPU. The library provides up to 44x faster performance compared to similar library alternatives. *[2]*

Chapter IV

VOIP DATA ANALYSIS

Analyzing the call data from the service provider has yielded information about how the service is used and abused. The patterns of VoIP usage for customers is described followed by a discussion of known fraud events that have occurred.

4.1 Call Patterns

The entire set of calls from the service provider was analyzed to study and chart the call patterns for legitimate (non-tagged) calls and calls tagged as occurring during a fraud event.

4.1.1 Calls per Hour

The following diagram shows a heatmap diagram of calls per hour for calls that are not tagged as fraud. The hour is based on UTC time while the service provider is in the Eastern timezone. This heatmap clearly shows the calls are mostly made during business hours of 9am to 5pm. Calls outside of weekday business hours are almost non-existent. The values for this heatmap are all calls made by this service provider.

The next diagram shows a heatmap of average calls per hour made per user. The calls per hour for each user was calculated for each combination of hour of the day and day of the week. This diagram takes into account the period over which the user was active. The calls per hour was calculated for each hour of the day and day of the week for each hour that that the user was active (from when their account was created to when it was disabled).

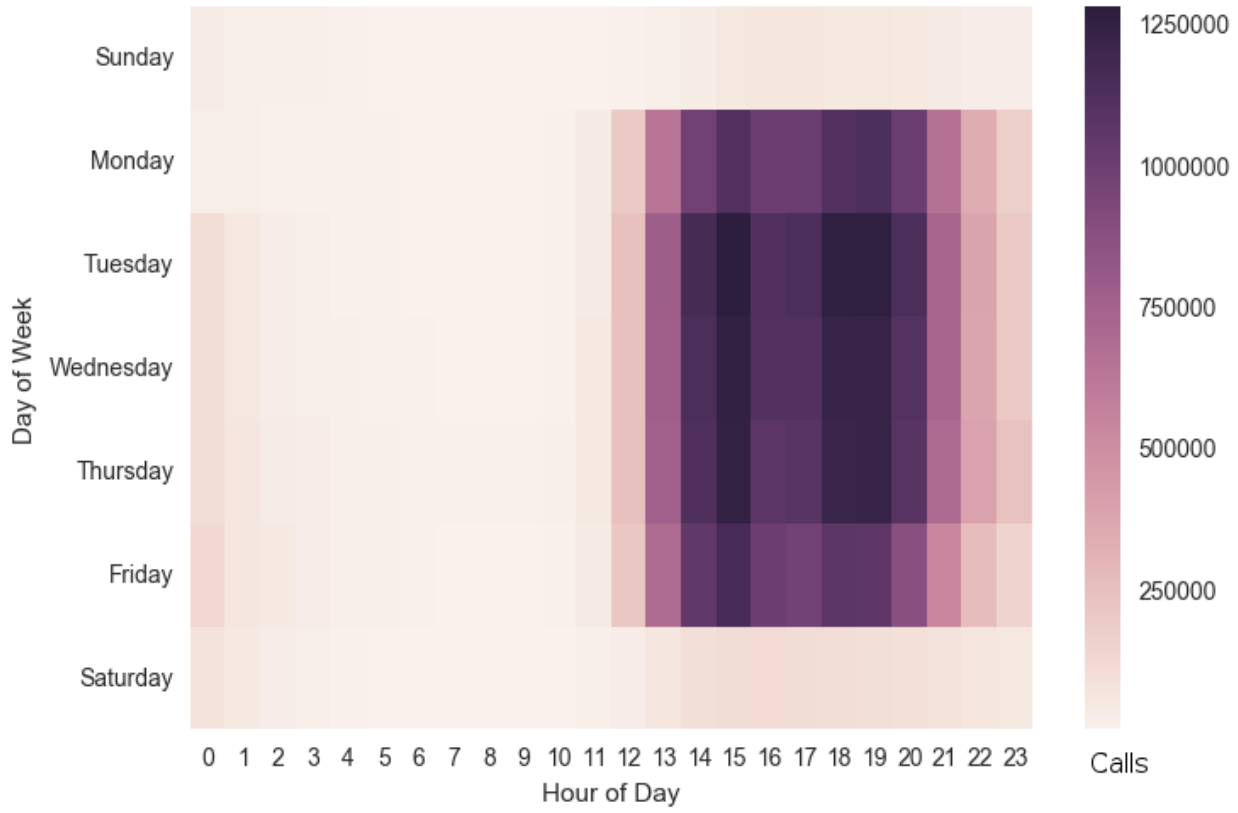

Fig. 4.1: Heatmap of non-fraud calls per hour and day of week.

This results in the average calls per hour for a user during each combination of hour of day and day of week. This diagram shows the same pattern as the total calls heatmap.

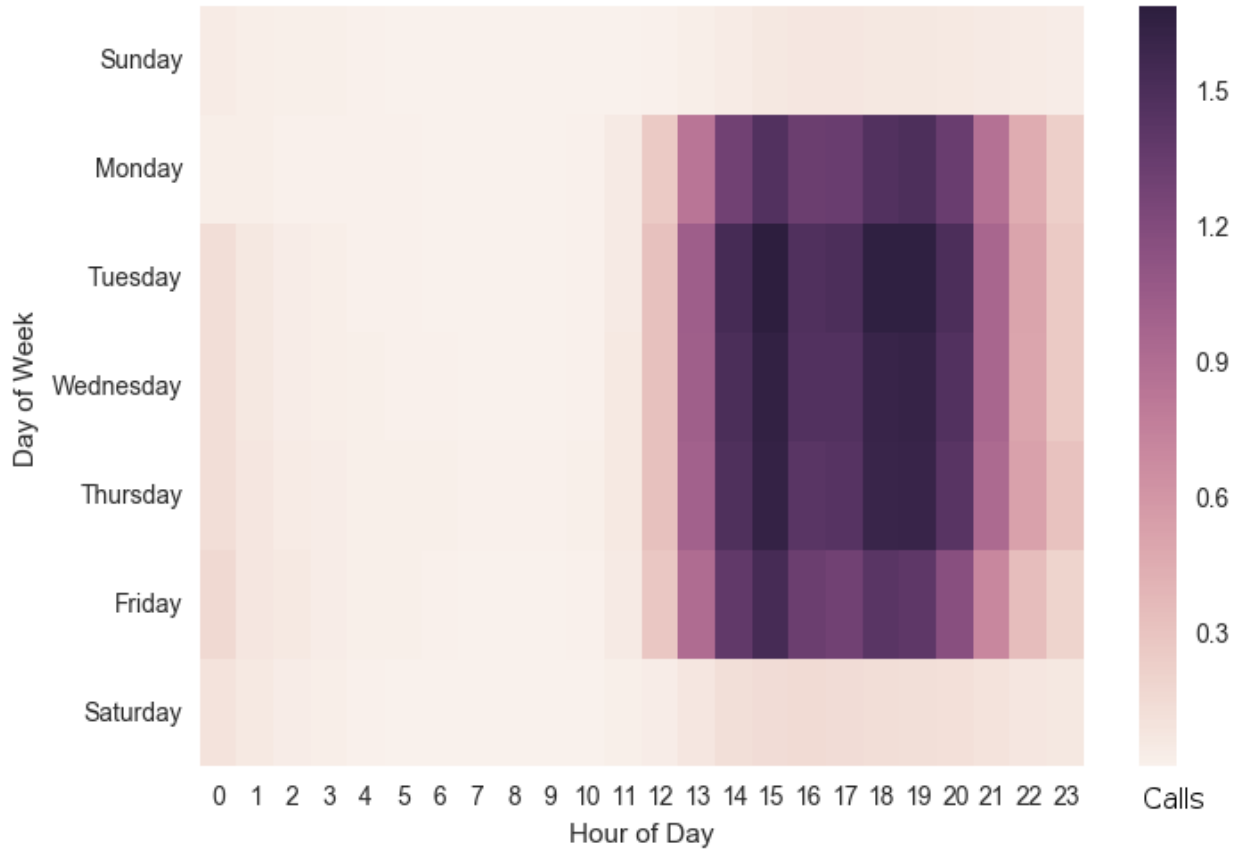

Fig. 4.2: Heatmap of average non-fraud calls per hour and day of week.

This diagram shows the heatmap of calls tagged as fraud. This is similar to the per user diagram in Fig. 4.1, but only for tagged calls. The heatmap is shaded slightly darker during the same business hours as the legitimate usage. This is caused by the real legimate usage that is happening alongside unauthorized usage. Fraud is tagged as a time period when the account was being used to send unauthorized calls rather than tagging individual calls as fraud. The total range of values is much smaller than Fig. 4.1, because there are simply less fraud calls than non-fraud.

Most of the cells that are much darker are outside of normal business hours and are easily distinguished. This shows that the nature of fraud events is a burst of many calls in a short time period. The shaded cells do not appear to have a pattern to them which could mean

that fraud events in general do not have a time preference. It's likely that each dark shaded cell is from a separate event or cross-section of fraud events.

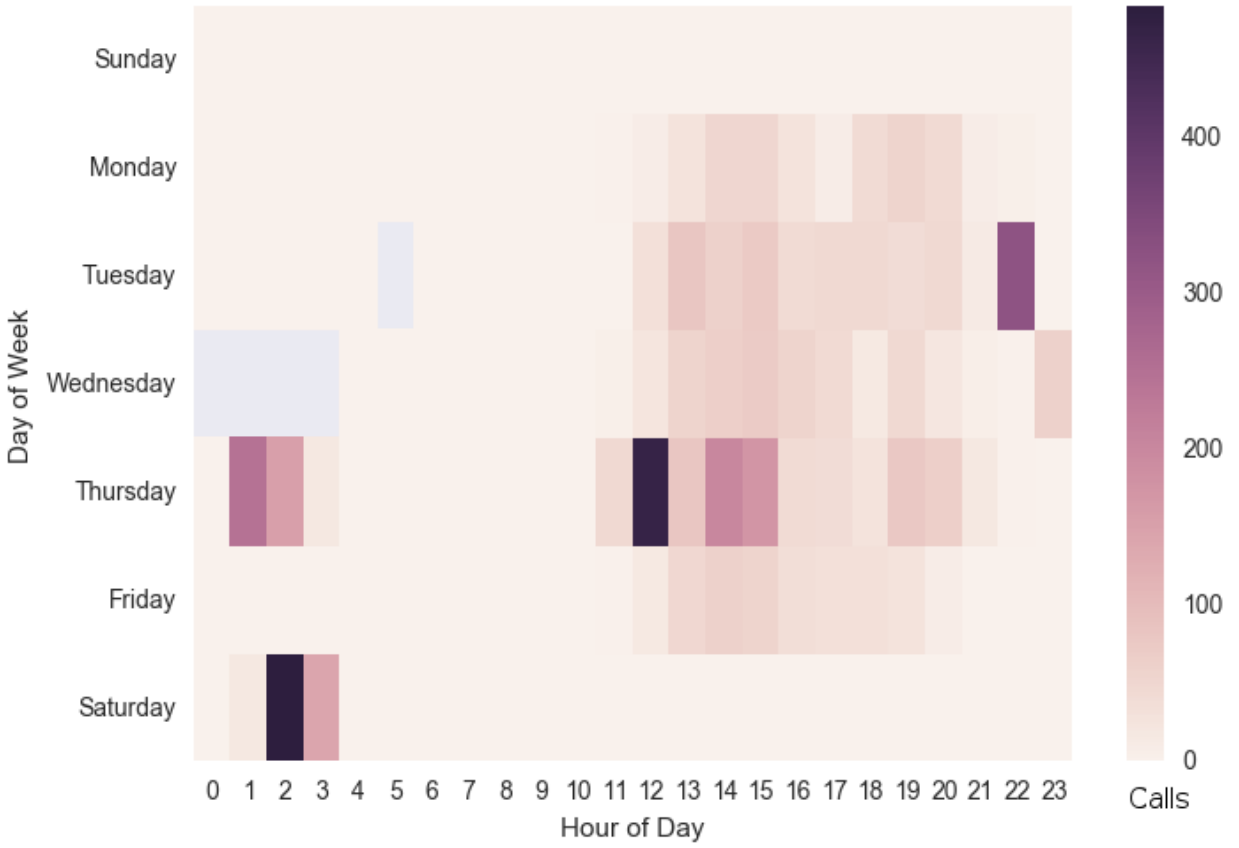

Fig. 4.3: Heatmap of fraud calls per hour and day of week.

This diagram shows the heatmap of the average calls per hour tagged as fraud. This is the same design as in Fig. 4.2, but only for tagged calls. This diagram shows some cells from the above heatmaps as darker and some as lighter. This heatmap uses per user calls per hour value resulting in lighter cells when multiple users were part of a fraud event. Note that the range of values in this diagram is from 0 to 150 calls per hour while the legitimate usage varies from 0 to 1.5 calls per hour. Fraud events result in a very high number of calls on a user account in a short time period.

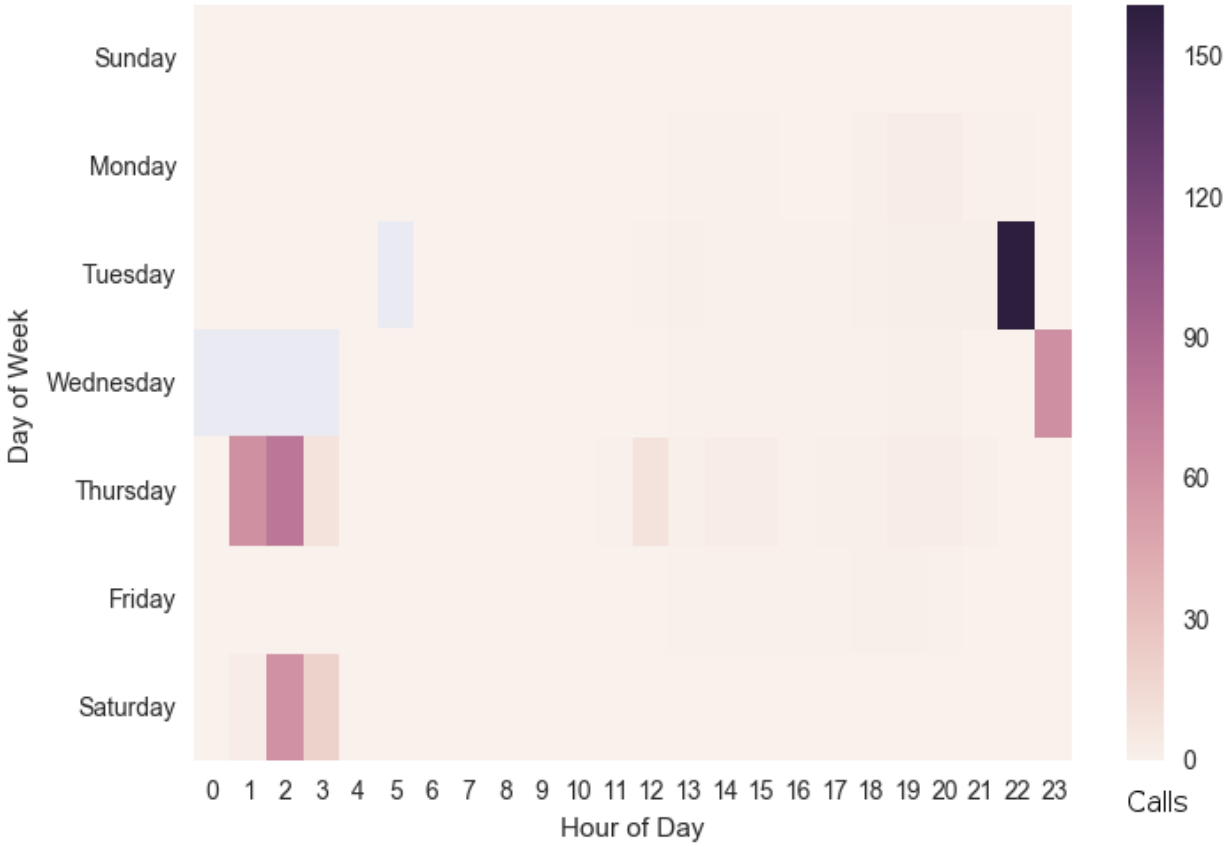

Fig. 4.4: Heatmap of average fraud calls per hour and day of week.

4.1.2 Charges

The following diagram is a histogram of the charges per toll call for non-fraud calls. Over 75% percent of all legitimate toll calls have a charge of $0.05 or less.

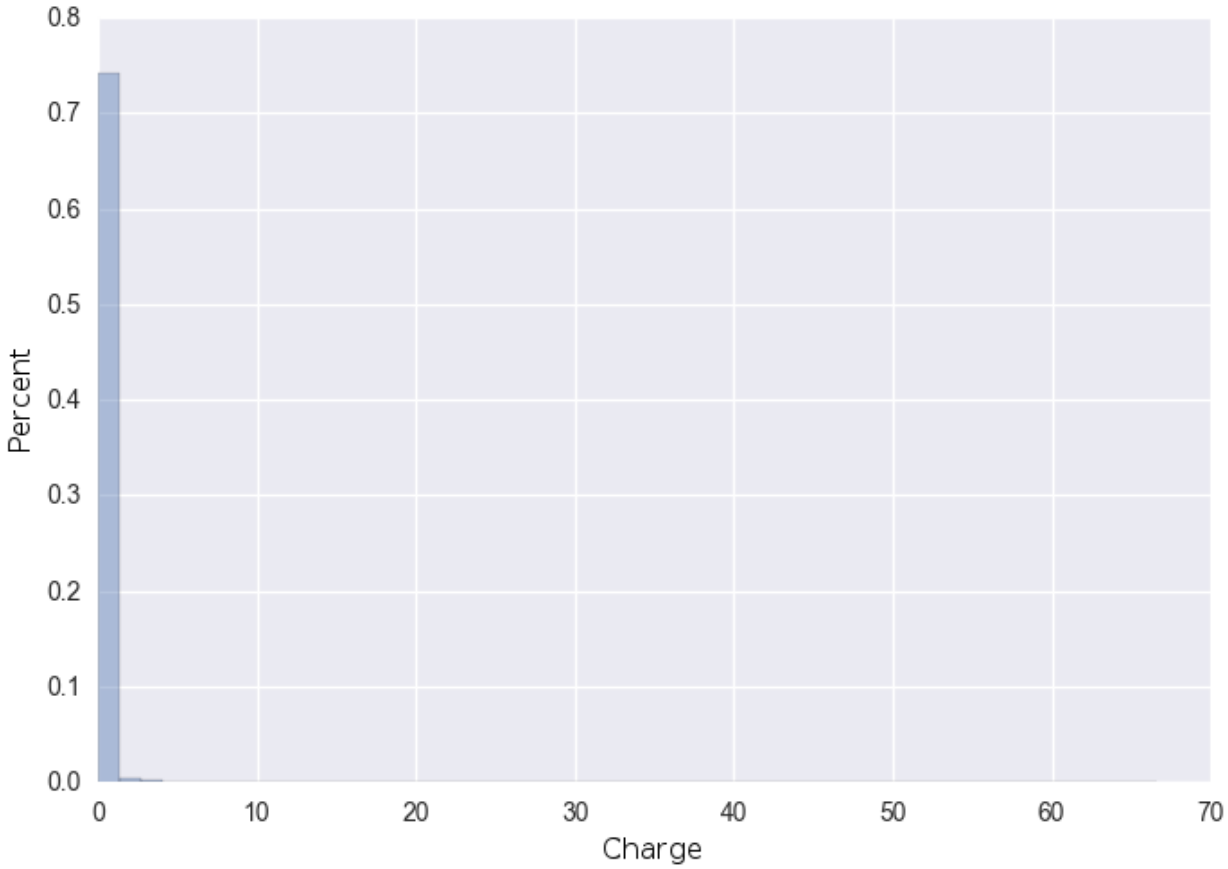

Fig. 4.5: Histogram of the charge per call for legitimate toll calls.

The next diagram shows the histogram of charges per toll call where the total charge is greater than $1.00. This histogram is also heavily weighted towards lower total charges. There is a clear pattern of low toll charges for legitimate calls. The distribution is clearly skewed right.

The following diagram is a histogram of the charges per toll call for calls tagged as fraud. There are more buckets that are clearly high and at the same number of calls than in the legitimate call histogram.

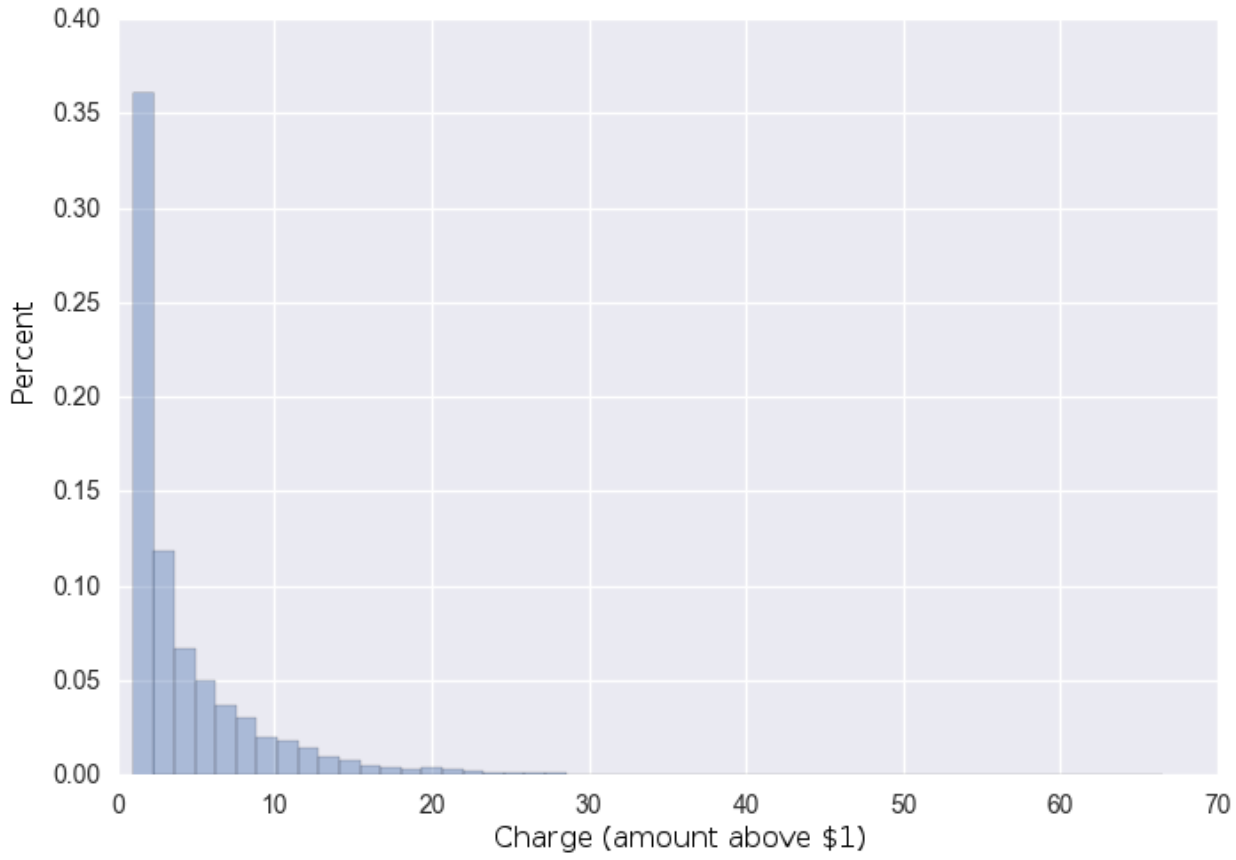

Fig. 4.6: Histogram of charge per call for legitimate toll calls where the charge is over $1.00.

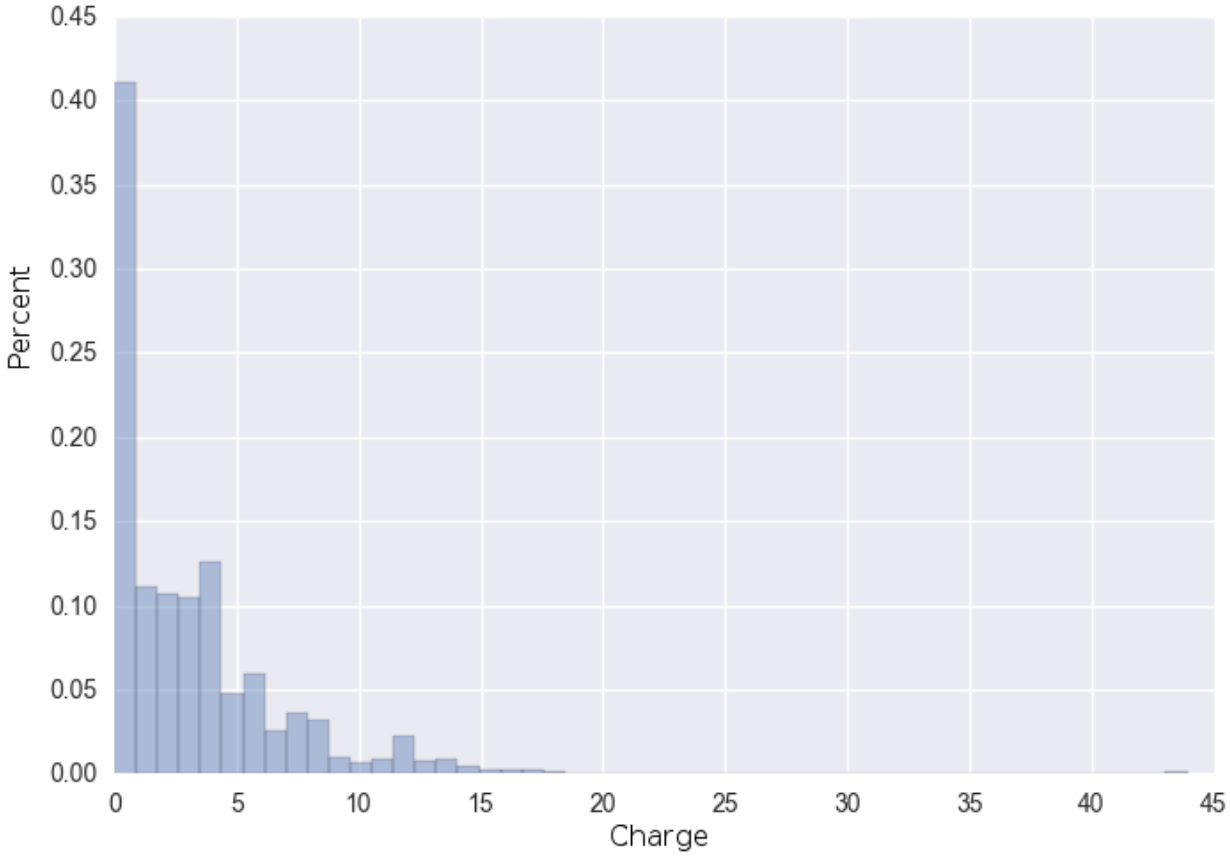

Fig. 4.7: Histogram of charge per call for fraud toll calls.

4.1.3 Duration

The vast majority of legitimate calls have a short duration as seen in the following histogram of call durations for legitimate toll calls.

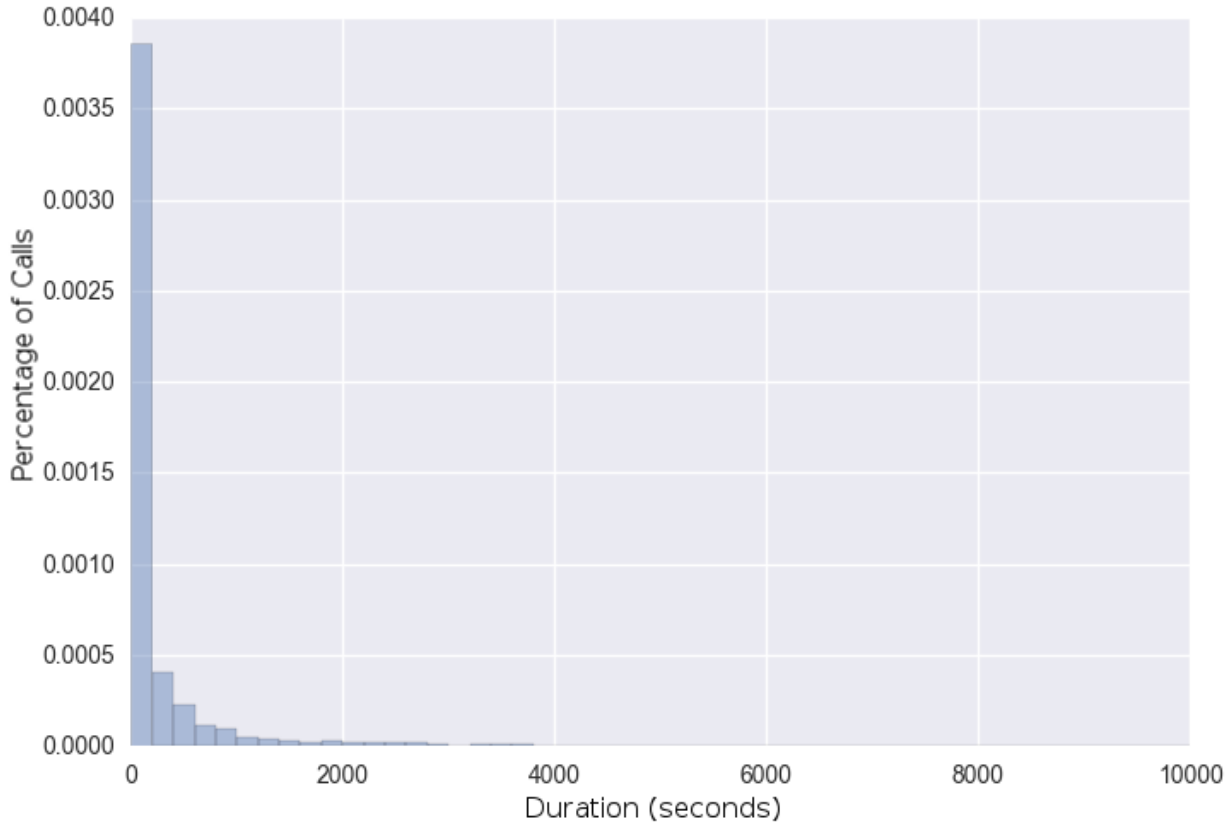

Fig. 4.8: Histogram of the duration of legitimate toll calls.

The next diagram is a histogram of call durations for calls tagged as fraud. The histogram is still skewed to the right, but has a bigger spread of call durations.

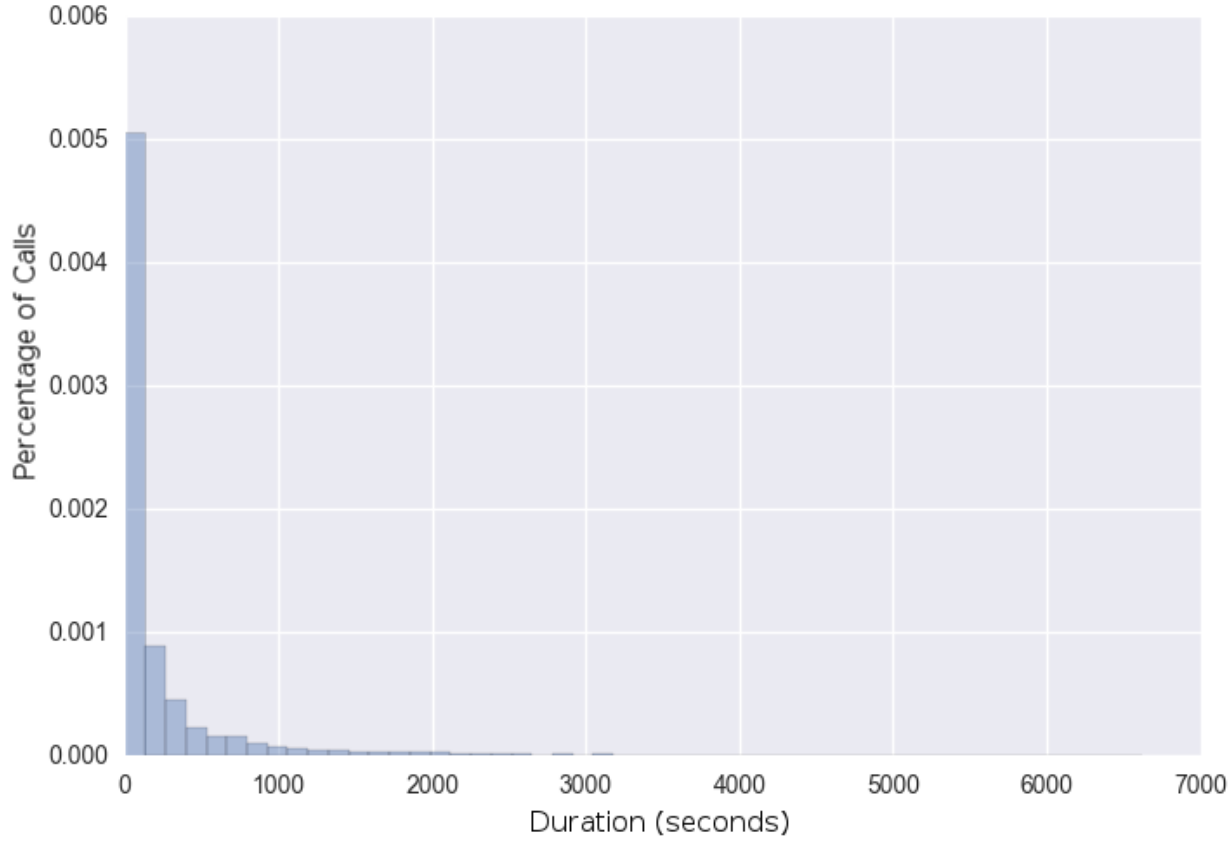

Fig. 4.9: Histogram of the duration of fraud toll calls.

4.1.4 Start Time Delta

Fig. 4.10 is a histogram of the time between calls. The values in the histogram are the number of seconds between the start time of the previous call and the start time of a toll call (of the same user). The previous call used in the time difference may or may not be a toll call, but it is the same user. This is a metric that shows how quickly in succession users make calls.

It is important to note that this is not the time between toll calls, but the time between a toll call and any other call by the same user. The maximum bucket value in the diagram is one hour. The first call of the business day for a user generally has a 16 hour time difference between their last call of the previous business day. The 95th percentile for the differences of calls is 2 hours 24 minutes. Keeping the full values in the diagram made it difficult to read because a small number of calls would have a very large time difference.

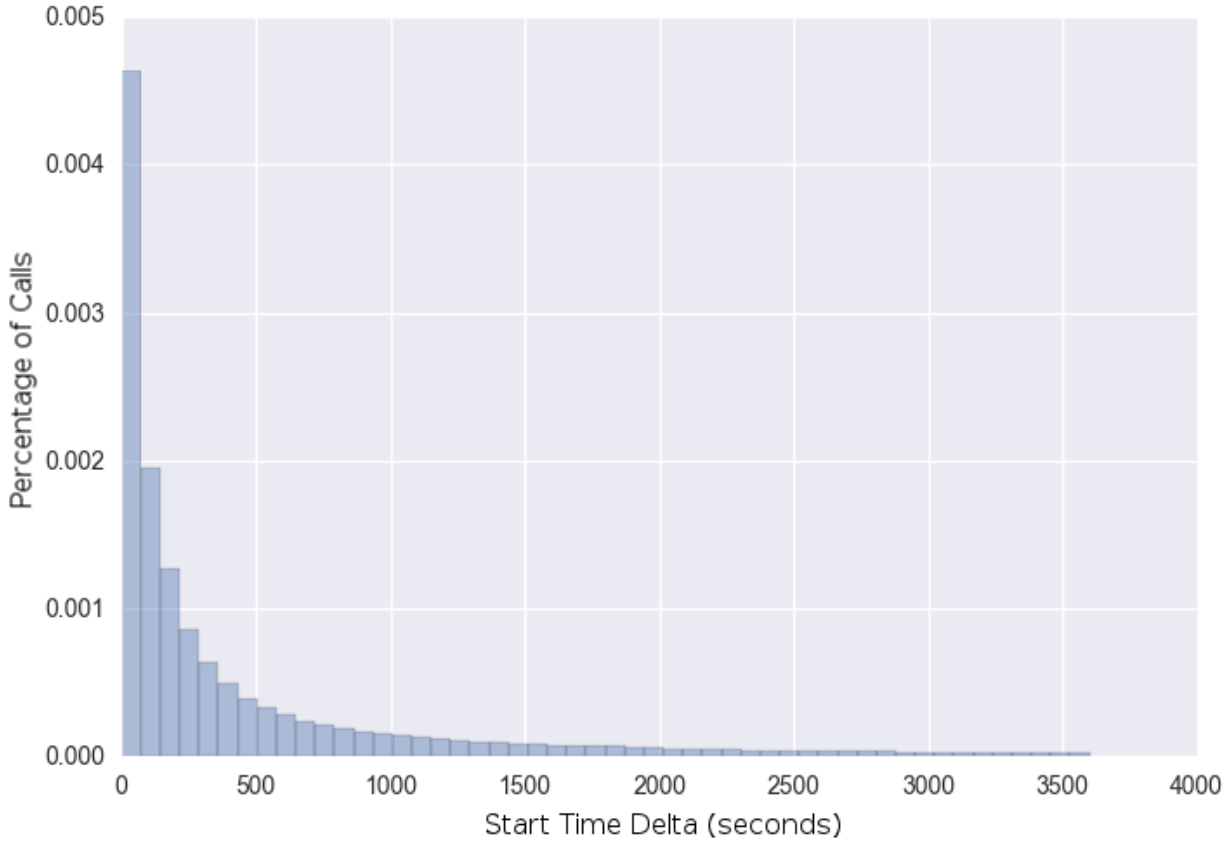

Fig. 4.10: Histogram of the time between legitimate calls.

The next diagram shows the histogram of the same time difference for calls that were tagged as fraud. The time difference is between any call and a toll call marked as fraud by the same user. This histogram is even more skewed right than the legitimate histogram above.

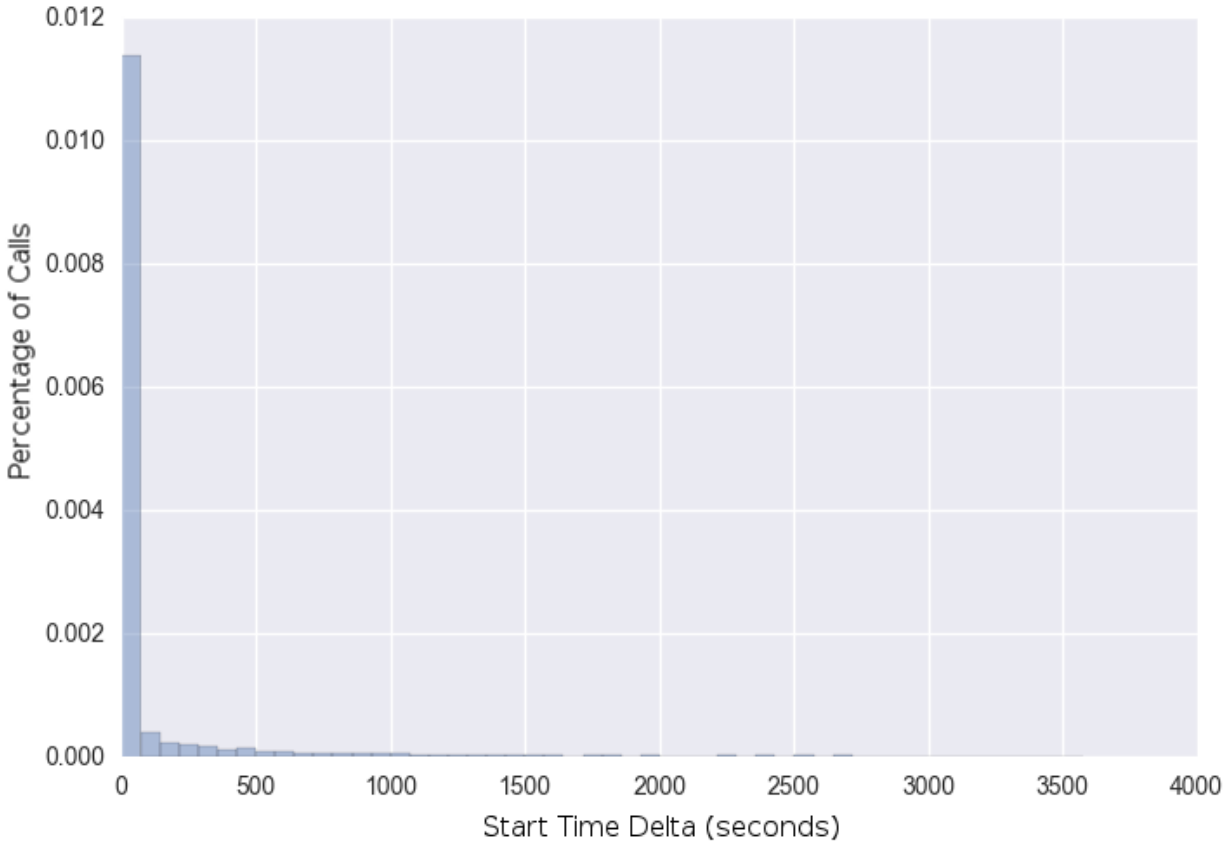

Fig. 4.11: Histogram of the time between fraud calls.

This shows that the fraud calls come in quick succession. The value is between start times which means that concurrent calls can result in short start time differences even if the duration last longer than the difference.

The next two diagrams visualize the start time difference between legitimate calls that are made sequentially and fraudulent calls made concurrently (by the same user).

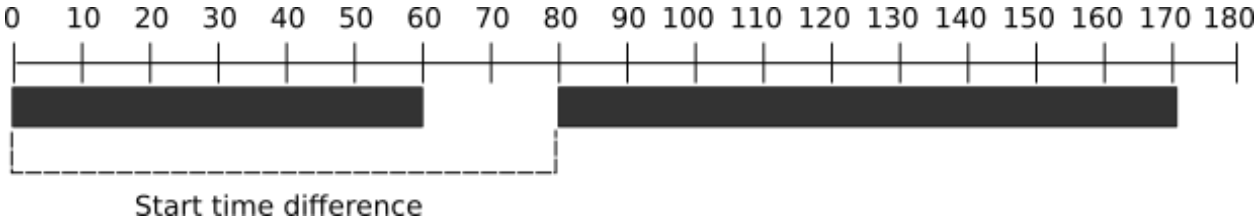

Fig. 4.12: Start time difference example for legitimate calls.

Fraud calls often have concurrent calls being made which decreases the start time difference between calls as shown below.

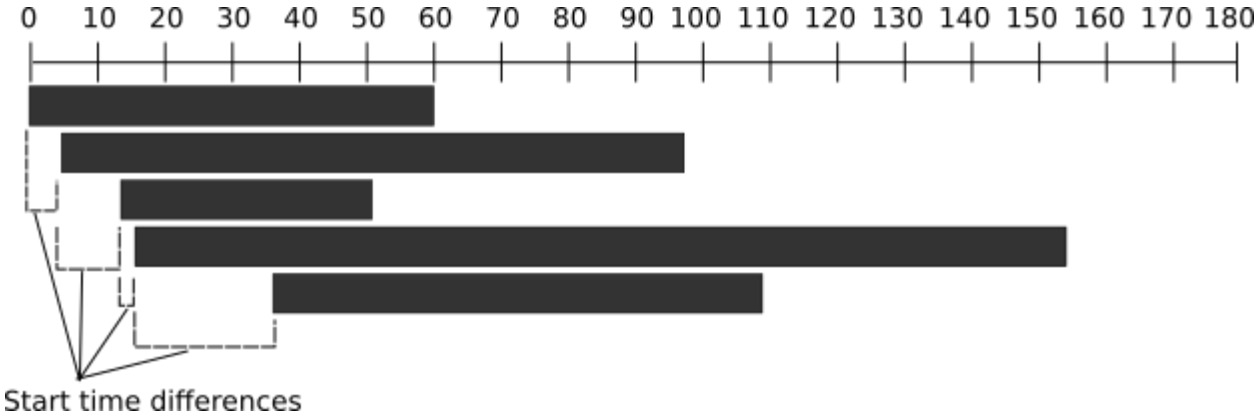

Fig. 4.13: Start time difference example for concurrent fraud calls.

4.1.5 Destinations

The following diagram is a bar chart of the top 25 destinations for legitimate toll calls. The service provider does not charge for calls within Canada meaning the “Canada / US” is strictly calls to the US. They are grouped together because they both use NANPA numbering with “1” as the country code. Other North American countries using NANPA include the NPA as part of their country code in their numbering schema.

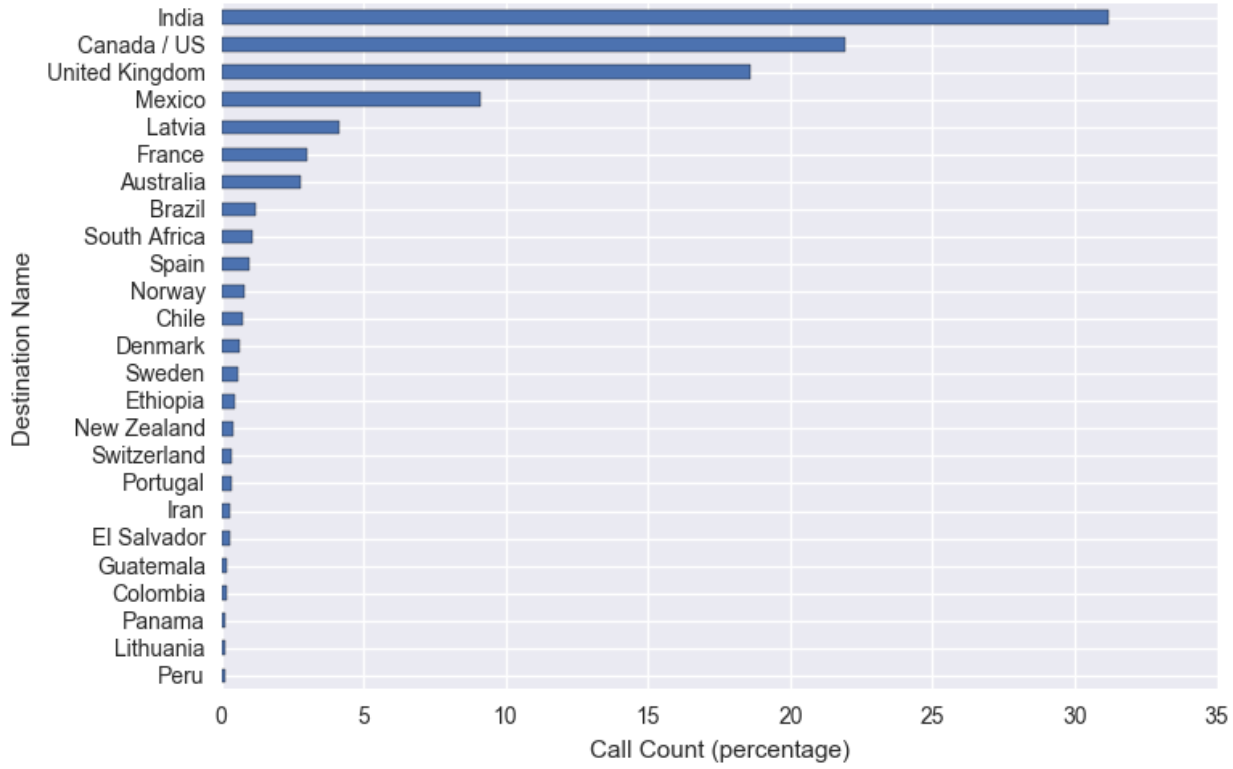

Fig. 4.14: Top 25 destinations for legitimate toll calls.

The next diagram is the top 25 destinations for calls marked as fraud as a percentage of total fraud calls. This graph shows that US and United Kingdom are significant destinations for both legitimate and fraud-tagged calls. This overlap of destinations increases the difficulty in determining if a call is fraud. Although Latvia, Chile, Morocco, and Sierra Leone are a significant percentage of destinations in the below chart it does not mean that all calls to those destinations can be tagged. In the dataset the number of fraud calls is much less than the number of legitimate calls. A legitimate destination that is not in the top 25 could be more than the number of calls in any of the fraud destination bars.

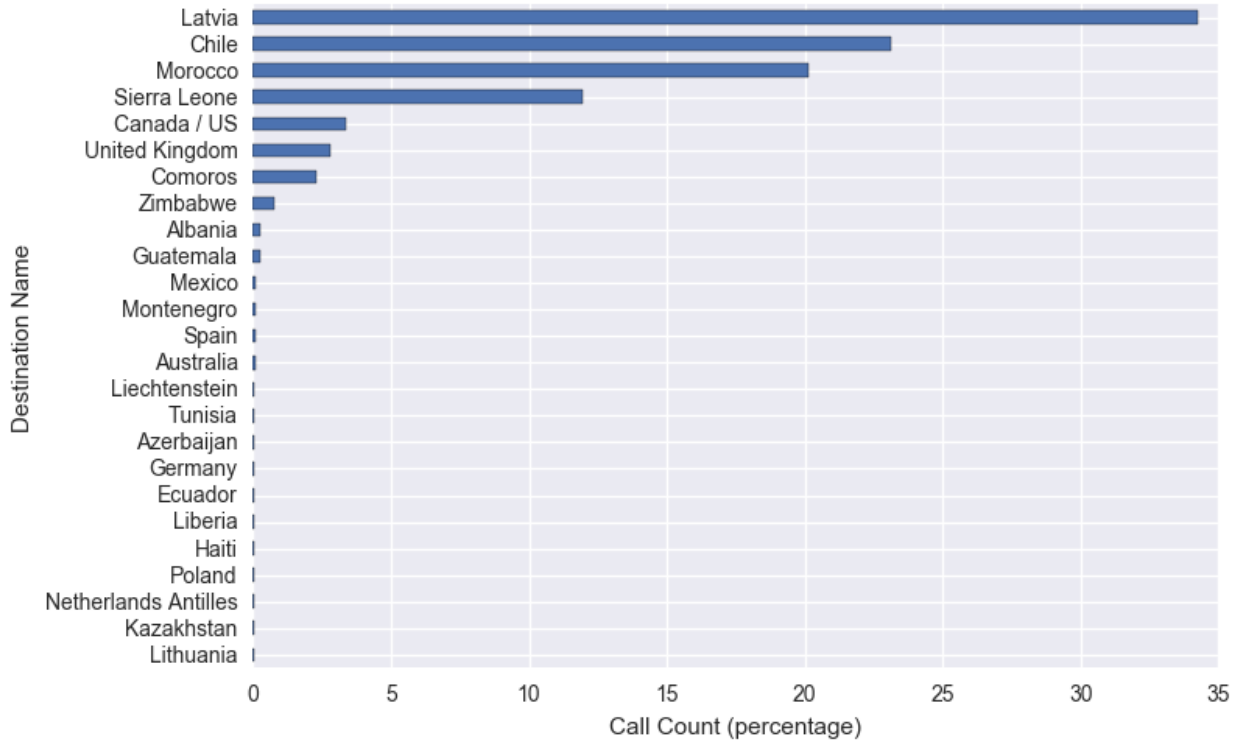

Fig. 4.15: Top 25 destinations for fraud toll calls.

To visualize the problem of both fraud and legitimate calls terminating at the same destination this next graph uses a stacked bar chart to show both. These are the top 25 fraud destinations that also have legitimate calls. This shows that although each of the top 25 legitimate and fraud calls do not overlap there is still significant overlap in other destinations. This precludes destinations from being an accurate measurement for fraud by themselves.

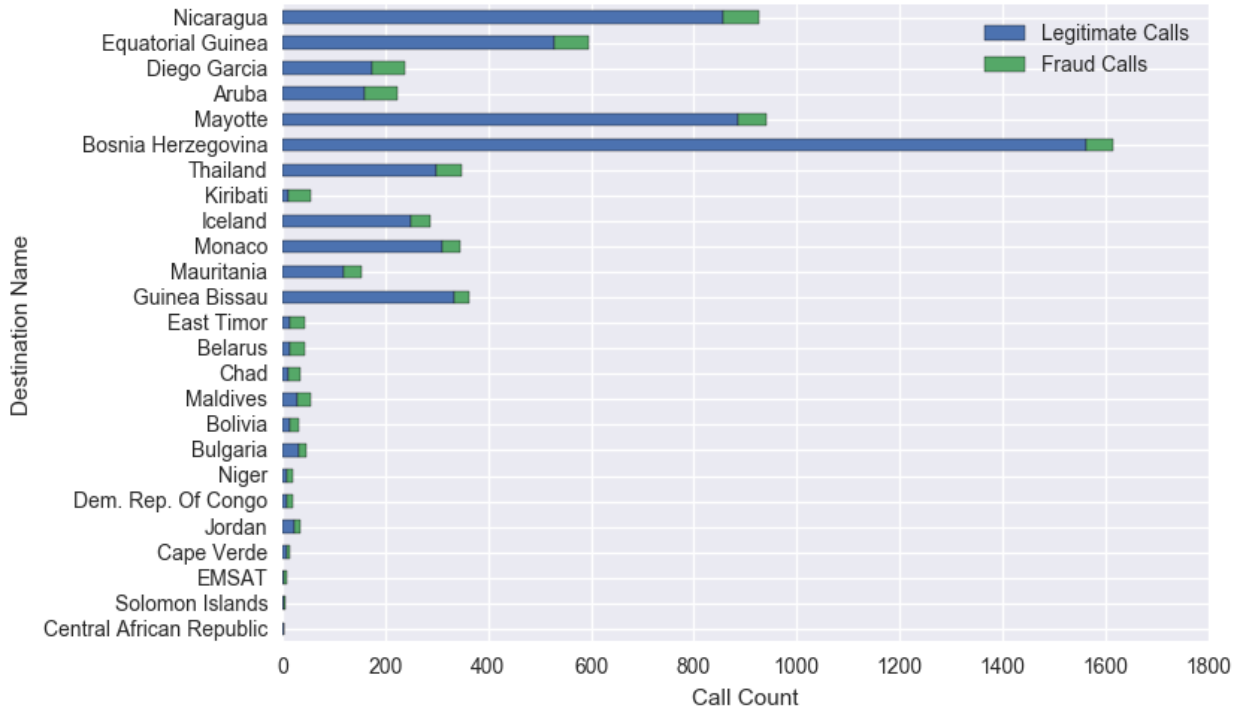

Fig. 4.16: Destinations with combined fraud and legitimate calls. Ordered by fraud call count.

4.2 Fraud Events

This section describes individual fraud events that occurred at the ITSP. The predominant destinations, toll charges, and durations have been given when possible. If the compromised account has relevant information then it has been included with a description that does not identify the user or customer.

4.2.1 August 31, 2010

In August 2010 there was a large fraud event that took place. The majority of calls terminated in Zimbabwe. This was a very high volume fraud event. Unfortunately no call records exist from this time period. This was just prior to deploying a new VoIP platform that tracked calls.

4.2.2 Morocco from 2011 to 2013

During the data analysis of call distributions for this paper it was revealed that a previously unknown fraud event had taken place between April 2011 and March 2013. All of the unauthorized calls associated with this event terminated in Morocco. At it's peak user accounts from 60 enterprises were making unauthorized calls to Morocco. The number of calls ranged from 100 to 1000 calls per month. The normal number of calls per month to Morocco is less than 10 (predominantly from a couple travel agencies).

This fraud event likely ended when passwords were changed for all users as part of an update to the ITSP's password policies. This even occurred very early into the deployment of the VoIP platform. Long distance tracking and billing had not been developed yet and therefore the customers had not seen these unauthorized calls to Morocco.

The following table shows the number of calls per month to Morocco for the lifetime of the VoIP platform. It also shows the number of customers that made calls to Morocco each month. Months with no calls have been omitted.

Table 4.1: Calls per month to Morocco.

Year-Month	Customers Affected	Call Count
2011-04	1	11
2011-05	4	20
2011-06	10	82
2011-07	15	143
2011-08	17	162
2011-09	24	287
2011-10	36	507
2011-11	37	558
2011-12	40	460
2012-01	42	701
2012-02	41	1098
2012-03	49	1101
2012-04	49	891
2012-05	55	1038
2012-06	51	889
2012-07	54	1288
2012-08	49	1113
2012-09	62	923
2012-10	60	1136
2012-11	59	1021
2012-12	56	668
2013-01	55	916
2013-02	59	817
2013-03	38	126
2013-04	1	4

Continued on next page

Table 4.1 – continued from previous page

Year-Month	Customers Affected	Call Count
2014-03	1	4
2014-04	1	4
2014-09	1	4
2015-05	2	8
2015-07	1	2
2015-10	1	2
2015-11	1	8

The unauthorized calls represent approximately 70 000 minutes in call at a cost of over $24 000. Spread out over two years this fraud went unnoticed.

4.2.3 July 2011

In July 2011 one enterprise had approximately 16 user accounts compromised. These accounts made unauthorized calls to at least 40 destinations. In the case the enterprise is a travel agency that regularly makes international telephone calls to many destinations. This makes detecting specific destinations for the unauthorized calls difficult. Twenty of the destinations during the fraud event were never called outside that event. These include:

- Ethiopia
- Sierra Leone
- Honduras
- Poland
- Lithuania
- Comoros

- Solomon Islands
- Tunisia
- Israel
- Guyana
- St. Helena
- North Korea
- Mariana Islands
- Zimbabwe
- Latvia
- EMSAT
- Congo
- Liechtenstein
- Greenland
- East Timor
- Azerbaijan

Over 1300 calls terminated in Sierra Leone and 600 in Zimbabwe. The fraud event took place between 2011-06-14 04:39 and 2011-07-05 03:49. The total number of minutes used was over 115 000 with a total cost of $47 000.

4.2.4 October 24, 2011

A user account for a phone kept in a utility closet in the basement of a building was found to be making phone calls. This user account is only used for ITSP technicians when cell phone service is unavailable in the basement. The main destinations include:

- Djibouti

- Maldives
- Azerbaijan
- Nicaragua
- Guyana

There were unauthorized calls to 145 unique destinations, but only five calls were answered. This fraud event lasted from 2011-10-24 15:53 to 2011-10-24 16:32 — under one hour. This shows how quickly a compromised account can make over 100 phone calls.

4.2.5 October 28, 2011

A single user account made 10 calls to unauthorized destinations. These destinations include:

- Djibouti
- Azerbaijan
- Cape Verde
- Congo
- Macedonia
- Maldives
- Marshall Islands
- United Kingdom

The fraud event occurred between 2011-10-28 18:17 and 2011-10-28 19:08. The total minutes duration was 412 with a cost of approximately $100.

4.2.6 November 18, 2011

A single user account made 250 unauthorized calls to two destinations: United Kingdom and Netherlands. The total minutes duration was 428 with a cost of approximately $135. This fraud event occurred between 2011-11-17 23:44 and 2011-11-18 00:45. In one hour a single compromised account was able to make 250 calls.

4.2.7 October 2011 to April 2012

Almost 100 users of an enterprise were compromised. The majority of calls terminated in Chile and Morocco. Other destinations include:

- Algeria
- Argentina
- Australia
- Austria
- Bahrain
- Belgium
- Bulgaria
- Canada
- Cape Verde
- Cayman Islands
- Chile
- Costa Rica
- Djibouti
- Egypt

- Eritrea
- France
- Gambia
- Germany
- Guatemala
- Haiti
- Honduras
- Iceland
- Israel
- Jamaica
- Japan
- Libya
- Luxembourg
- Macedonia
- Maldives
- Mauritania
- Mayotte
- Mexico
- Morocco
- Netherlands
- New Zealand
- Norway
- Palestine

- Puerto Rico
- San Marino
- Sao Tome Principe
- Senegal
- Seychelles
- South Africa
- Spain
- Tanzania
- Tunisia
- United Kingdom
- Vanuatu

The total duration of calls was over 11 000 minutes. The fraud event occurred between 2011-10-03 and 2012-04-26. This was very soon after the account was activated.

4.2.8 August to November, 2012

Nine users of a single service provider were compromised during a fraud event between 2012-08-09 and 2012-11-13. The total duration of compromised calls was over 7000 minutes. The accounts were compromised soon after the account was activated. The majority of calls terminate in Tunisia, Morocco, and Chile. Other destinations include:

- Algeria
- Australia
- Azerbaijan
- Bahrain
- Belarus

- Bosnia Herzegovina
- Bulgaria
- Chile
- Djibouti
- Egypt
- El Salvador
- Equatorial Guinea
- France
- Germany
- Guatemala
- Haiti
- Israel
- Jordan
- Kazakhstan
- Laos
- Latvia
- Liberia
- Montenegro
- Morocco
- Nicaragua
- Slovenia
- St. Pierre Miquelon
- Switzerland

- Tunisia
- United Kingdom
- Vanuatu

4.2.9 January 13, 2014

A single user was compromised between 2014-01-13 01:23 and 2014-01-13 02:36. During the hour long fraud event 476 calls were made with a total duration of over 3500 minutes. The destinations include:

- Albania
- Azerbaijan
- Bosnia Herzegovina
- Eritrea
- Gambia
- Guinea Bissau
- Israel
- Latvia
- Madagascar
- Moldova
- Monaco
- Oman
- Senegal
- Serbia

4.2.10 August 26, 2015

A user account for an elevator telephone was compromised between 2015-08-26 22:00 and 2015-08-26 23:54. This user account is normally not used and reserved for an emergency in an elevator. 64 calls were made during the fraud event with 9 minutes total duration. The destinations include:

- Ascension Island
- Azerbaijan
- Bolivia
- Bosnia Herzegovina
- Central African Republic
- Comoros
- East Timor
- Guinea
- Guinea Bissau
- Somalia
- Tunisia
- United Kingdom

4.2.11 June to December, 2015

Approximately 15 users of a single enterprise were found to be making unauthorized calls. The fraud event occurred between 2015-06-18 12:15 and 2015-12-09 16:19. The total duration was over 33 000 minutes with a cost of over $17 000. The unauthorized destinations called are:

- Albania

- Aruba
- Bermuda
- Cayman Islands
- Chad
- Comoros
- Ecuador
- Eritrea
- Guatemala
- Haiti
- Israel
- Kiribati
- Latvia
- Liberia
- Maldives
- Monaco
- Morocco
- Niger
- United Kingdom

Chapter V

VOIP DATA ANALYSIS ALGORITHM DISCUSSION

To determine the algorithm parameters to use for the LSTM algorithm three algorithm runs with test cases were developed. These use a smaller dataset to get an idea for the parameters to use on the full data set. The algorithm is then trained on the full data set with it's progress monitored during training. The algorithm is then tested against a separate testing dataset to calculate the accuracy. The trained weights will be analyzed to determine if there is any information that can be extracted.

5.1 Algorithm Parameters

To test algorithm parameters a small test was designed that could be run in a reasonable time frame. The test trains the algorithm on the fraud calls of a single user and uses validation data for the same user to measure performance. The algorithm parameters that are adjusted include:

- Sequence length — 10 and 20 call sequences.
- Include destinations — Specifies whether to include all destination names in the data vectors. These are a sub-designation of the location of the call within the country. The destination names are a large vocabulary
- Vector length — The length of the data vector. This is not adjusted directly, but changes due to the "include destinations" flag.

- Model activation — This is the activation function applied to the output of the algorithm.
- Optimizer — Function used for training the weights of the algorithm.
- LSTM Dimensions — Number of weights (dimensions) of the LSTM algorithm.

There are 60 variations of the algorithm parameters as shown in the table below.

Table 5.1: Algorithm parameter variations.

Variation Name	Sequence Length	Include Destinations	Vector length	Model Activation	Optimizer	LSTM Dimensions
A	10	True	1810	softmax	rmsprop	512
B	10	True	1810	softmax	rmsprop	1810
C	10	False	321	softmax	rmsprop	512
D	10	False	321	softmax	rmsprop	321
E	10	False	321	softmax	rmsprop	963
F	10	True	1810	softsign	rmsprop	512
G	10	True	1810	softsign	rmsprop	1810
H	10	False	321	softsign	rmsprop	512
I	10	False	321	softsign	rmsprop	321
J	10	False	321	softsign	rmsprop	963
K	10	True	1810	softmax	adam	512
L	10	True	1810	softmax	adam	1810
M	10	False	321	softmax	adam	512
N	10	False	321	softmax	adam	321
O	10	False	321	softmax	adam	963
P	10	True	1810	softsign	adam	512
Q	10	True	1810	softsign	adam	1810
R	10	False	321	softsign	adam	512

Continued on next page

Table 5.1 – continued from previous page

Variation Name	Sequence Length	Include Destinations	Vector length	Model Activation	Optimizer	LSTM Dimensions
S	10	False	321	softsign	adam	321
T	10	False	321	softsign	adam	963
U	10	True	1810	softmax	adadelta	512
V	10	True	1810	softmax	adadelta	1810
w	10	False	321	softmax	adadelta	512
X	10	False	321	softmax	adadelta	321
Y	10	False	321	softmax	adadelta	963
Z	10	True	1810	softsign	adadelta	512
AA	10	True	1810	softsign	adadelta	1810
AB	10	False	321	softsign	adadelta	512
AC	10	False	321	softsign	adadelta	321
AD	10	False	321	softsign	adadelta	963
AE	20	True	1810	softmax	rmsprop	512
AF	20	True	1810	softmax	rmsprop	1810
AG	20	False	321	softmax	rmsprop	512
AH	20	False	321	softmax	rmsprop	321
AI	20	False	321	softmax	rmsprop	963
AJ	20	True	1810	softsign	rmsprop	512
AK	20	True	1810	softsign	rmsprop	1810
AL	20	False	321	softsign	rmsprop	512
AM	20	False	321	softsign	rmsprop	321
AN	20	False	321	softsign	rmsprop	963
AO	20	True	1810	softmax	adam	512
AP	20	True	1810	softmax	adam	1810
AQ	20	False	321	softmax	adam	512

Continued on next page

Table 5.1 – continued from previous page

Variation Name	Sequence Length	Include Destinations	Vector length	Model Activation	Optimizer	LSTM Dimensions
AR	20	False	321	softmax	adam	321
AS	20	False	321	softmax	adam	963
AT	20	True	1810	softsign	adam	512
AU	20	True	1810	softsign	adam	1810
AV	20	False	321	softsign	adam	512
AW	20	False	321	softsign	adam	321
AX	20	False	321	softsign	adam	963
AY	20	True	1810	softmax	adadelta	512
AZ	20	True	1810	softmax	adadelta	1810
BA	20	False	321	softmax	adadelta	512
BB	20	False	321	softmax	adadelta	321
BC	20	False	321	softmax	adadelta	963
BD	20	True	1810	softsign	adadelta	512
BE	20	True	1810	softsign	adadelta	1810
BF	20	False	321	softsign	adadelta	512
BG	20	False	321	softsign	adadelta	321
BH	20	False	321	softsign	adadelta	963

The “Avg Duration per Epoch” field is average length of time to train the algorithm variation on the entire set of training data. The “Variation Name” field is used to identify this variation in the results analysis.

The algorithm was trained on 100 epochs of the training data. Metrics are saved at each epoch. The metrics are calculated on both the training data and the validation data. Only the training data is used to update the weights of the algorithm. The validation data is used for metrics, because over-training may give high accuracy on the training data while not being accurate on the validation data.

Multiple single user cases were trained and tested against the validation data. This will provide information about the accuracy of the different variations in parameters by randomly sampling individual users. The best variations will be used to train an algorithm against all users in the dataset.

The test cases were hand-picked from the training data to provide a variety of test cases. Ideally these would also be excluded from the training data to provide separate sets for hyperparameter optimization and algorithm training, however with a relatively low number of tagged calls the training set used for algorithm training does include the test cases which could affect the generalizability of the trained model.

5.1.1 Case 1A

The first case uses the user account a08428ca5a which has over 3000 calls tagged as occurring during a fraud event. This user has less than 400 calls that were not tagged.

The following diagram shows the accuracy during training on each epoch for every variation. The accuracy is calculated on the training data. It is the percentage of times the category was correctly calculated.

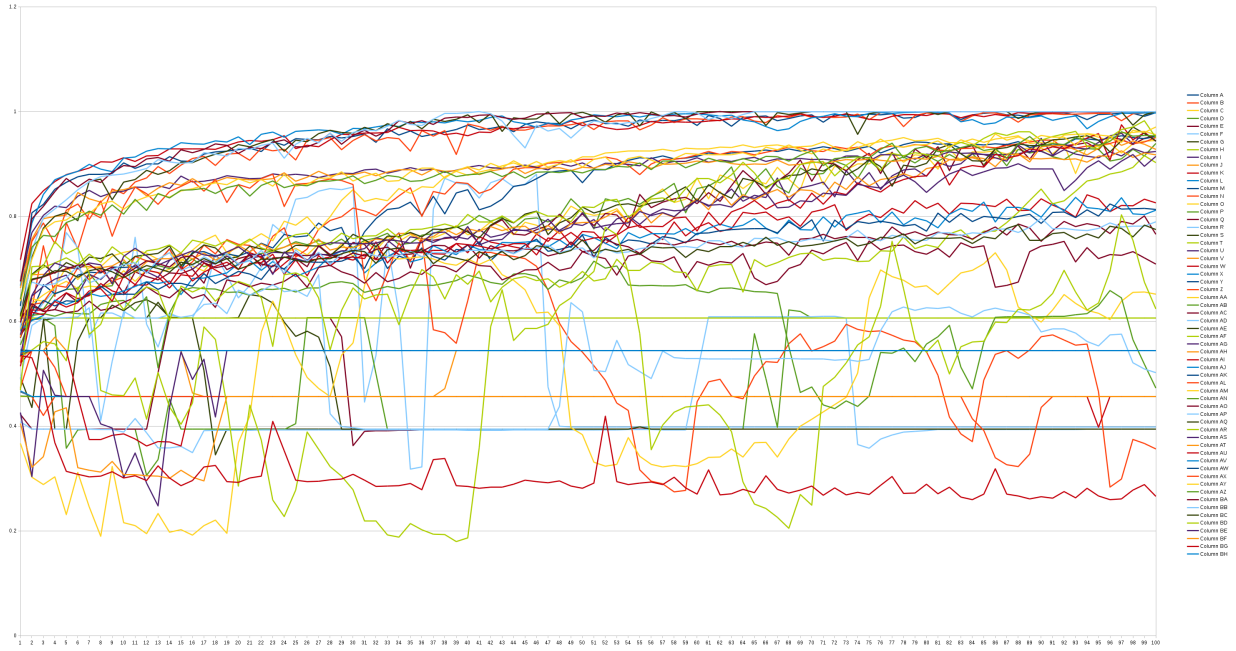

Fig. 5.1: Categorical accuracy on training set during training.

The diagram appears to have groups of variations that train more quickly and consistently and also variations that are more erratic during the training. Although the accuracy against the training data is improving during training this does not mean that the algorithm is applicable to other data. The diagram below shows the same training metrics applied to the validation data at each epoch.

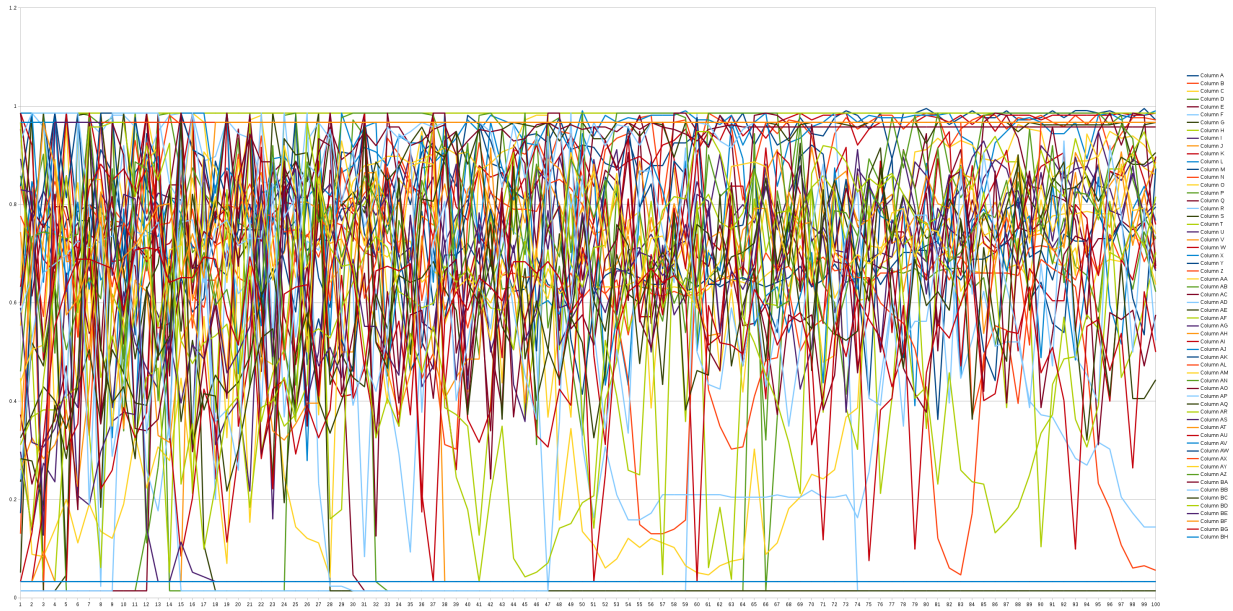

Fig. 5.2: Categorical accuracy on validation set during training.

As expected the accuracy against the validation data is much more erratic. Looking at the top seven variations shows a clearer picture of the data. These variations all end up with more than 98% accuracy on the validation data. The diagram below shows the accuracy through the training epochs on the validation data.

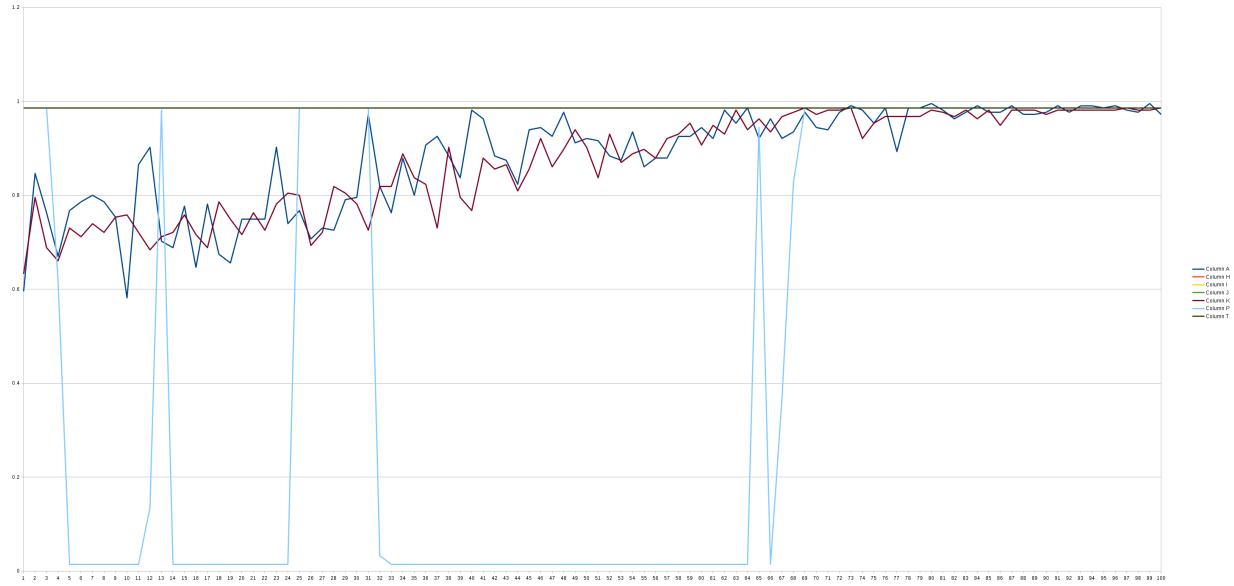

Fig. 5.3: Top variations categorical accuracy on validation set during training.

The accuracy of these same variations on the training data appear as:

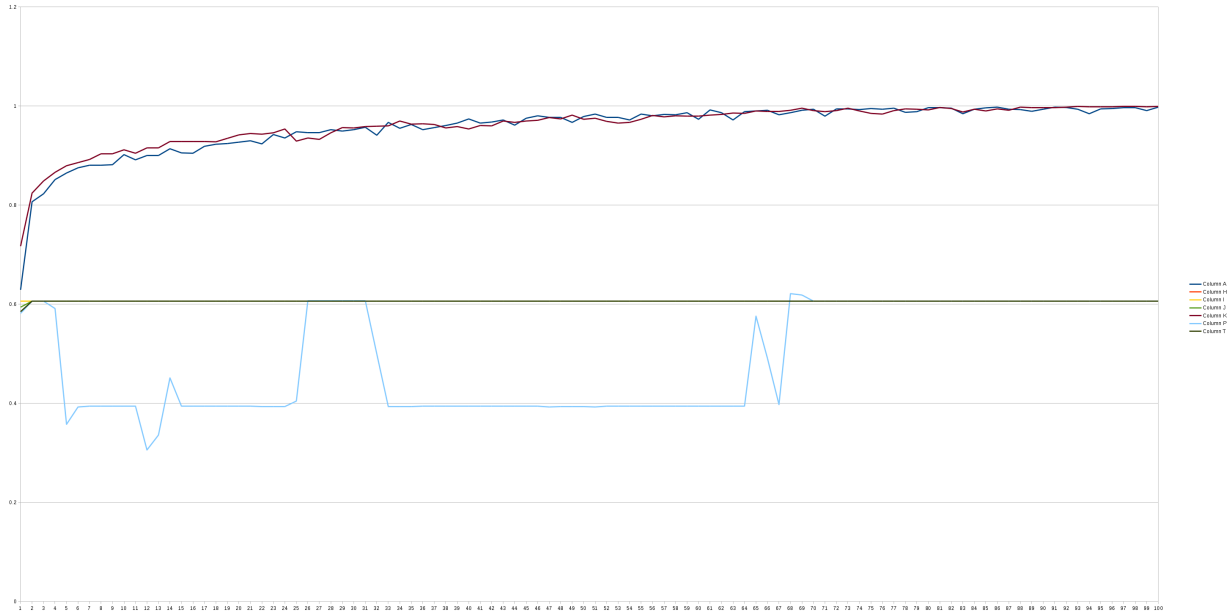

Fig. 5.4: Top variations categorical accuracy on training set during training.

The variations with accuracy over 98% are A, H, I, J, K, P, and T.

5.1.2 Case 1B

This case is the same as Case 1A, but is for the user 06cf210beb. This user has about 300 calls tagged as occurring during fraud and 200 calls tagged as not occuring during fraud. This case has a closer ratio of tagged to non-tagged calls than Case 1A.

The diagram of accuracy during training is similar to Case 1A, but has a single group of variations that reach high accuracy, while a few variations are erratic or converge on solutions that do not provide accurate classifications.

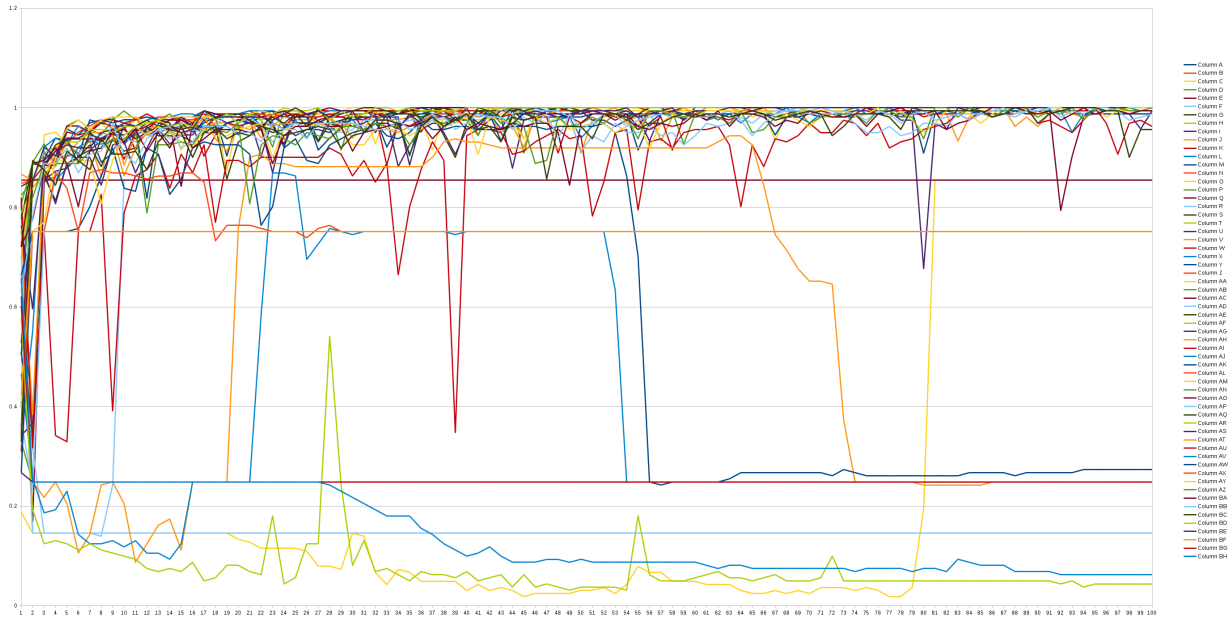

Fig. 5.5: Categorical accuracy on training set during training.

The accuracy on the validation set is much more consistent than Case 1A. This may be due to the more even ratio of tagged to non-tagged data. Or this user may have less attributes required to classify fraud calls.

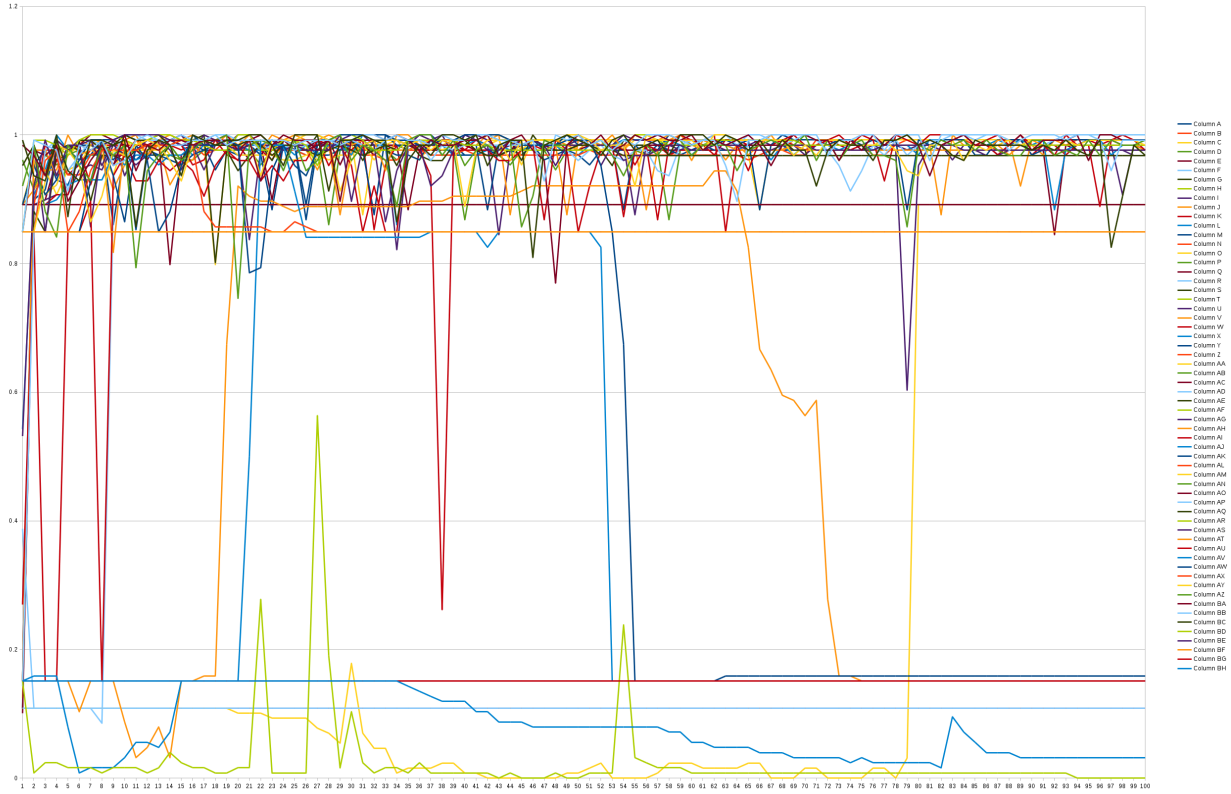

Fig. 5.6: Categorical accuracy on validation set during training.

There are more variations in this case where the accuracy is above 98%. The following diagram shows the accuracy on the validation set during training for the variations that had an accuracy over 98%.

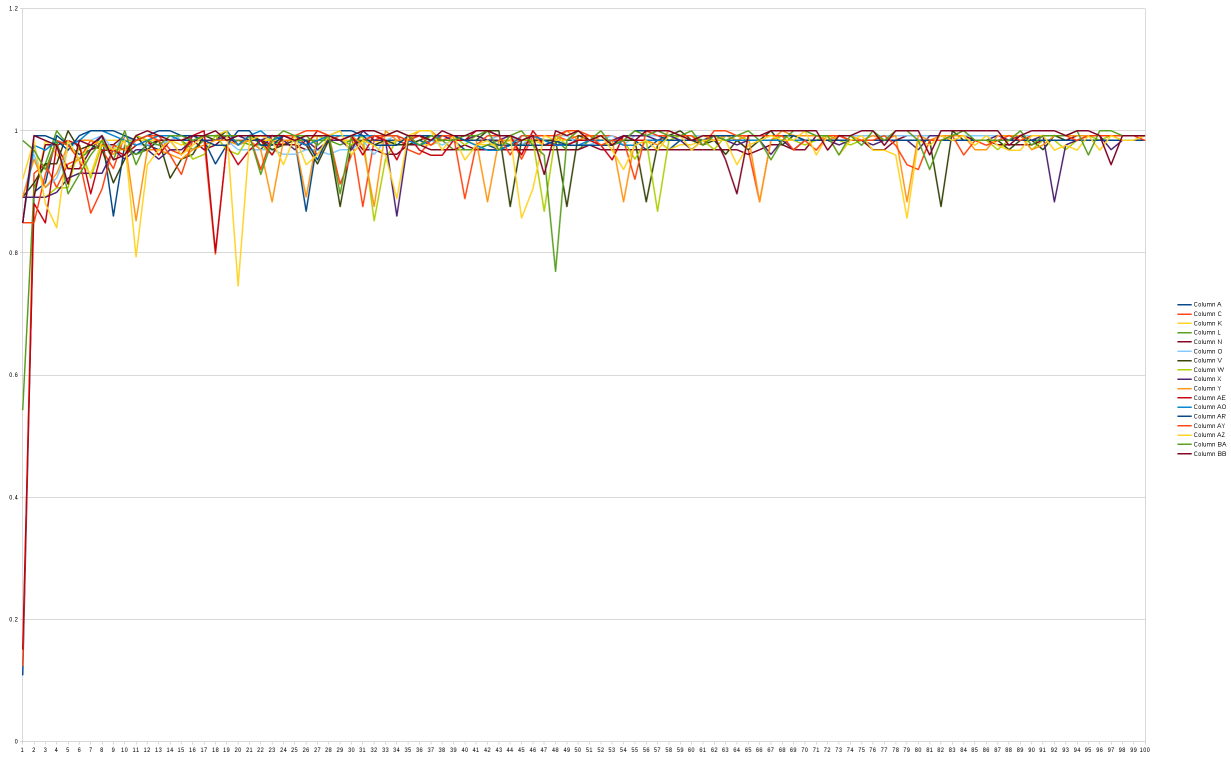

Fig. 5.7: Top variations categorical accuracy on validation set during training.

There are two groups of variations that resulted in the same accuracy after converging. Variations N, O, X, and Y converge on the same accuracy of 99.225%. This means that they have converged on the same weights for classifying the data. Variations AZ, BA, and BB converged with an accuracy of 99.206%.

Those variations above 99% accuracy do not correspond to the top variations in Case 1A. The other variations above 98% accuracy are: A, C, K, L, V, W, AE, AO, AP, AR, and AY. The only overlap is with variation A. Variation AP in Case 1B converged on 100% accuracy.

5.1.3 Case 1C

This case is is also trained with the calls from a single user account. The user 14c492693a has approximately 1000 calls tagged as occurring during fraud and 32000 calls not tagged.

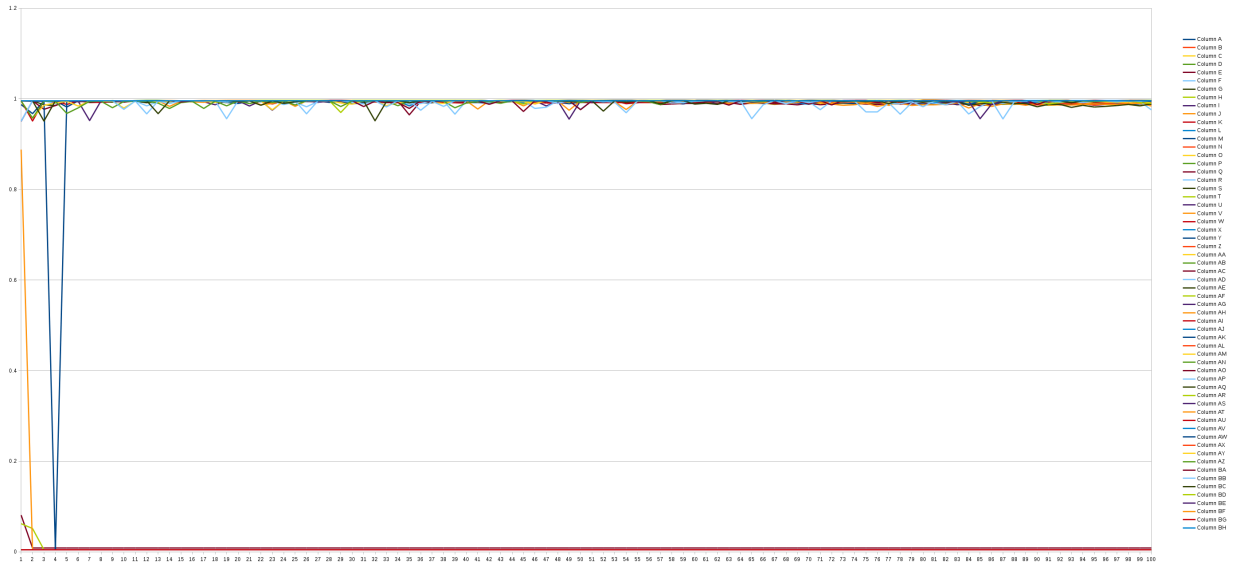

Fig. 5.8: Categorical accuracy on validation set during training.

The training on this case quickly converged most of the variations to over 99% or less than 1% accuracy.

5.1.4 Parameter Conclusions

The following tables shows a breakdown of the final accuracy and duration for each of the three cases above. The accuracy is the percentage of correct classifications in the final epoch of training after 100 epochs. The duration is the average duration in seconds per epoch of training.

Table 5.2: Hyperparameter test cases performance.

Variation	Case 1A Accuracy	Case 1A Duration	Case 1B Accuracy	Case 1B Duration	Case 1C Accuracy	Case 1C Duration	Avg Accuracy
L	0.9907	10.09	0.9845	1.05	0.9934	46.81	0.9895
K	0.9860	2.14	0.9845	0.39	0.9915	10.87	0.9873
A	0.9721	2.32	0.9845	0.36	0.9918	11.35	0.9828
AE	0.9670	6.58	0.9841	1.23	0.9962	101.99	0.9824
AP	0.9670	19.48	1.0000	2.62	0.9756	121.30	0.9809
AO	0.9575	8.36	0.9841	1.19	0.9965	100.94	0.9794
H	0.9860	1.76	0.8915	0.33	0.9953	8.42	0.9576
P	0.9860	2.14	0.8915	0.37	0.9953	11.68	0.9576
AQ	0.9057	2.35	0.9683	0.41	0.9886	13.81	0.9542
E	0.8977	4.17	0.9767	0.51	0.9877	19.28	0.9541
Y	0.8744	4.18	0.9922	0.53	0.9942	18.96	0.9536
B	0.9721	9.95	0.8915	1.06	0.9953	45.07	0.9529
AG	0.8915	3.02	0.9683	0.44	0.9951	14.28	0.9516
AH	0.8868	2.77	0.9762	0.39	0.9841	12.92	0.9490
AR	0.8726	4.01	0.9841	0.36	0.9896	12.34	0.9488
O	0.8605	4.25	0.9922	0.51	0.9933	20.91	0.9487
AJ	0.9670	4.86	0.8492	1.16	0.9951	98.66	0.9371

Continued on next page

Table 5.2 – continued from previous page

Variation	Case 1A Accuracy	Case 1A Duration	Case 1B Accuracy	Case 1B Duration	Case 1C Accuracy	Case 1C Duration	Avg Accuracy
AL	0.9670	2.94	0.8492	0.40	0.9951	14.75	0.9371
AX	0.9670	5.05	0.8492	0.78	0.9951	35.47	0.9371
BF	0.9670	2.77	0.8492	0.42	0.9951	14.34	0.9371
AA	0.8884	10.08	0.8915	1.10	0.9953	44.73	0.9250
AS	0.8019	5.16	0.9762	0.78	0.9953	35.43	0.9245
M	0.7953	1.82	0.9767	0.33	0.9897	9.32	0.9206
X	0.7674	1.57	0.9922	0.31	0.9958	7.46	0.9185
AI	0.7594	7.82	0.9762	0.82	0.9951	35.89	0.9102
AY	0.7406	9.27	0.9841	1.21	0.9967	17.57	0.9071
N	0.7349	1.65	0.9922	0.30	0.9885	8.38	0.9052
V	0.6977	10.14	0.9845	1.08	0.9957	44.64	0.8926
w	0.6698	1.83	0.9845	0.36	0.9959	8.61	0.8834
D	0.6837	1.62	0.9767	0.29	0.9881	7.62	0.8829
U	0.6651	2.2	0.9767	0.39	0.9961	10.72	0.8793
AZ	0.6226	16.2	0.9921	2.59	0.9965	107.82	0.8704
C	0.6372	1.66	0.9845	0.33	0.9886	8.50	0.8701
BA	0.5755	2.32	0.9921	0.42	0.9955	15.44	0.8543
AC	0.6744	1.54	0.8915	0.32	0.9953	7.46	0.8537
AF	0.7028	20.64	0.8492	2.73	0.9951	157.12	0.8490
BB	0.5236	3.04	0.9921	0.36	0.9958	13.02	0.8372
BC	0.4434	5.33	0.9683	0.77	0.9959	36.76	0.8025
AM	0.9670	2.68	0.1508	0.35	0.9951	13.30	0.7043
AN	0.9670	7.74	0.1508	0.76	0.9951	35.23	0.7043
AU	0.9670	18.37	0.1508	2.58	0.9951	162.10	0.7043

Continued on next page

Table 5.2 – continued from previous page

Variation	Case 1A Accuracy	Case 1A Duration	Case 1B Accuracy	Case 1B Duration	Case 1C Accuracy	Case 1C Duration	Avg Accuracy
S	0.0140	1.77	0.8915	0.30	0.9953	7.72	0.6336
I	0.9860	1.62	0.8915	0.29	0.0047	7.58	0.6274
J	0.9860	4.19	0.8915	0.52	0.0047	19.23	0.6274
T	0.9860	4.46	0.8915	0.52	0.0047	19.44	0.6274
AK	0.0330	19.16	0.8492	2.58	0.9951	152.20	0.6258
AB	0.7302	1.79	0.8915	0.36	0.0047	8.57	0.5421
AD	0.1442	4.14	0.1085	0.54	0.9953	19.01	0.4160
AW	0.0330	3.22	0.1587	0.36	0.9951	12.37	0.3956
AV	0.9670	2.84	0.1508	0.41	0.0049	13.83	0.3742
BH	0.0330	7.17	0.0317	0.77	0.9951	34.47	0.3533
Q	0.0140	10.23	0.8915	1.04	0.0086	46.96	0.3047
F	0.0140	2.06	0.8915	0.37	0.0047	10.67	0.3034
G	0.0140	9.85	0.8915	1.04	0.0047	44.99	0.3034
R	0.0140	1.87	0.8915	0.33	0.0047	8.56	0.3034
BD	0.8160	8.4	0.0000	1.23	0.0049	24.52	0.2736
BG	0.5000	2.39	0.1508	0.37	0.0049	12.46	0.2186
AT	0.0330	8.93	0.1508	1.24	0.0049	76.10	0.0629
BE	0.0330	20.61	0.1508	2.57	0.0049	128.43	0.0629
Z	0.0558	2.11	0.1085	0.38	0.0047	10.68	0.0564

The table above is sorted by the average accuracy accross all three cases. Variation K provides a good trade-off of high accuracy and short training duration. The parameters to be used are:

- Optimizer — adam
- Model activation — softmax

- LSTM dimensions — 512
- Sequence length — 10
- Vector length — 1808 (includes destination names vocabulary)

The full vector structure is shown in the following table including the indexes of each vocabulary value. The vocabulary for country codes and destination names are truncated in the table for brevity.

Table 5.3: Input vector structure.

Index	Feature	Value Type	Vocabulary Term
0	direction	Vocabulary	None
1	direction	Vocabulary	Terminating
2	direction	Vocabulary	Originating
3	answer_indicator	Vocabulary	None
4	answer_indicator	Vocabulary	Yes
5	answer_indicator	Vocabulary	No
6	answer_indicator	Vocabulary	Yes-PostRedirection
7	start_time_delta	Normalized	
8	start_time_category	Vocabulary	None
9	start_time_category	Vocabulary	day
10	start_time_category	Vocabulary	morning
11	start_time_category	Vocabulary	evening
12	start_time_category	Vocabulary	night
13	termination_cause	Vocabulary	None
14	termination_cause	Vocabulary	003
15	termination_cause	Vocabulary	01
16	termination_cause	Vocabulary	111
17	termination_cause	Vocabulary	016

Continued on next page

Table 5.3 – continued from previous page

Index	Feature	Value Type	Vocabulary Term
18	termination_cause	Vocabulary	027
19	termination_cause	Vocabulary	001
20	termination_cause	Vocabulary	017
21	termination_cause	Vocabulary	031
22	termination_cause	Vocabulary	9
23	termination_cause	Vocabulary	021
24	termination_cause	Vocabulary	041
25	call_category	Vocabulary	None
26	call_category	Vocabulary	national
27	call_category	Vocabulary	intralat
28	call_category	Vocabulary	private
29	call_category	Vocabulary	local
30	call_category	Vocabulary	emerg
31	call_category	Vocabulary	internat
32	call_category	Vocabulary	other
33	call_category	Vocabulary	interlat
34	network_call_type	Vocabulary	None
35	network_call_type	Vocabulary	sv
36	network_call_type	Vocabulary	poa
37	network_call_type	Vocabulary	da
38	network_call_type	Vocabulary	csv
39	network_call_type	Vocabulary	in
40	network_call_type	Vocabulary	oa
41	network_call_type	Vocabulary	pcs
42	network_call_type	Vocabulary	tf
43	network_call_type	Vocabulary	em

Continued on next page

Table 5.3 – continued from previous page

Index	Feature	Value Type	Vocabulary Term
44	network_call_type	Vocabulary	oap
45	network_call_type	Vocabulary	to
46	network_call_type	Vocabulary	lps
47	network_call_type	Vocabulary	nil
48	network_call_type	Vocabulary	lo
49	charge_indicator	Vocabulary	None
50	charge_indicator	Vocabulary	n
51	charge_indicator	Vocabulary	y
52	releasing_party	Vocabulary	None
53	releasing_party	Vocabulary	local
54	releasing_party	Vocabulary	remote
55	releasing_party	Vocabulary	none
56	redirecting_reason	Vocabulary	None
57	redirecting_reason	Vocabulary	BW-ExplicitID.4002
58	redirecting_reason	Vocabulary	follow-me
59	redirecting_reason	Vocabulary	BW-ExplicitID.4001
60	redirecting_reason	Vocabulary	BW-ExplicitID.1515
61	redirecting_reason	Vocabulary	call-center
62	redirecting_reason	Vocabulary	BW-ExplicitID.1609
63	redirecting_reason	Vocabulary	hunt-group
64	redirecting_reason	Vocabulary	deflection
65	redirecting_reason	Vocabulary	BW-ExplicitID.6619
66	redirecting_reason	Vocabulary	BW-ImplicitID
67	redirecting_reason	Vocabulary	user-busy
68	redirecting_reason	Vocabulary	BW-ExplicitID.4000
69	redirecting_reason	Vocabulary	unconditional

Continued on next page

Table 5.3 – continued from previous page

Index	Feature	Value Type	Vocabulary Term
70	redirecting_reason	Vocabulary	unknown
71	redirecting_reason	Vocabulary	time-of-day
72	redirecting_reason	Vocabulary	unavailable
73	redirecting_reason	Vocabulary	no-answer
74	redirecting_reason	Vocabulary	BW-ExplicitID.1608
75	redirecting_reason	Vocabulary	BW-ExplicitID.1504
76	country_code	Vocabulary	None
77	country_code	Vocabulary	248
78	country_code	Vocabulary	359
79	country_code	Vocabulary	218
...	...	...	...
316	destination_name	Vocabulary	None
317	destination_name	Vocabulary	Netherlands Antilles - Other
318	destination_name	Vocabulary	Central African Republic - Telecel Mobile
319	destination_name	Vocabulary	Iraq - WLL
...	...	...	...
1805	rate	Normalized	
1806	duration	Normalized	
1807	charge	Normalized	

5.2 Algorithm Training

An LSTM model was created with the following parameters:

- Optimizer — adam
- Model activation — softmax
- LSTM dimensions — 512

- Sequence length — 10
- Vector length — 1810 (includes destination names vocabulary)

This model was trained on all calls of users that had at least some of their calls being tagged as during fraud in the training data set. Once the calls are processed into data vectors and into overlapping sequences of 10 calls each the data set takes up approximately 16GB of space. In order to train on the large amount of data it was generated iteratively and fed into the algorithm in batches. These batches ensure that less than 4GB of data were necessarily stored in memory at any point in time during training.

The algorithm was trained for 47 iterations of 20 epochs which took approximately two weeks. The computer used to run the calculations had 16GB of RAM, Intel i5 processor with four cores, and Nvidia GeForce GTX 960 GPU. At the end of each training iteration the current weights for the algorithm are saved. This will allow loading the algorithm weights at any given iteration for testing. After each iteration the categorical accuracy and categorical cross-entropy loss is calculated against the validation data set. The metrics are saved to provide insight into the training process.

5.3 Algorithm Testing

The following diagram shows the categorical cross-entropy loss against both the validation and training datasets after each epoch of training.

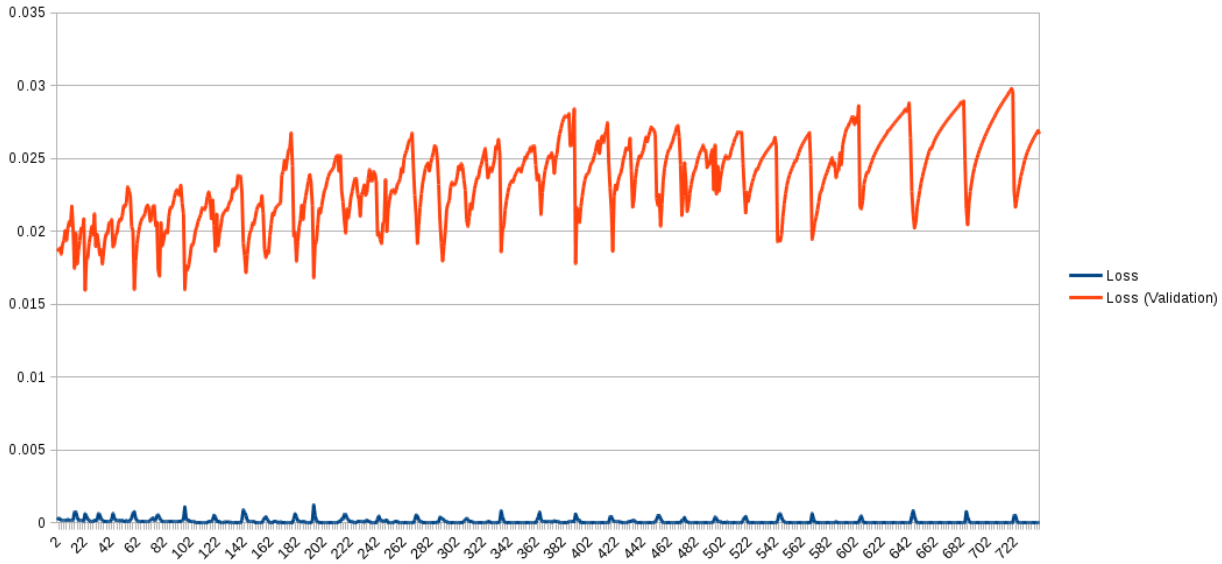

Fig. 5.9: Categorical cross-entropy loss over training iterations.

The following diagram shows the categorical accuracy against both the validation and the training datasets after each epoch of training.

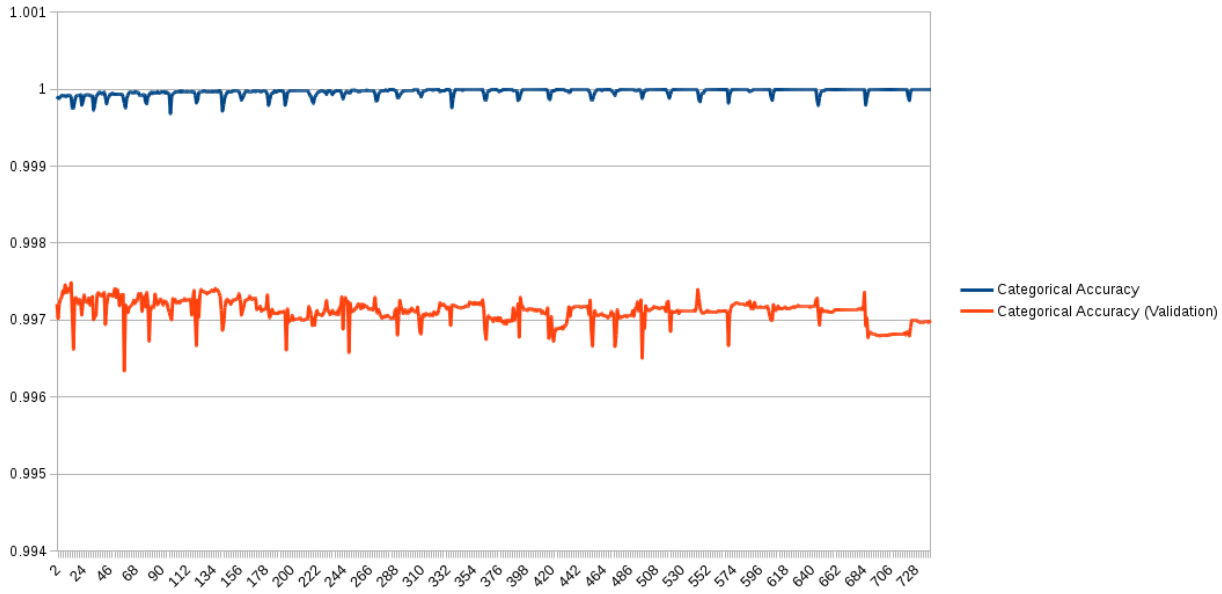

Fig. 5.10: Categorical accuracy over training iterations.

After training the algorithm was tested against the test data set of 1,149,382 call records. This dataset had no part in developing the algorithm parameters and provides a clean set of calls to evaluate the performance. After 200 epochs of training the categorical cross-entropy loss is 0.0362 and the categorical accuracy is 99.52% when applied against the test dataset. After the complete 940 epochs of training the categorical cross-entropy loss is 0.0505 and the categorical accuracy is 99.50% when applied against the test dataset.

The loss against the testing set was tested after every 20 epochs of training. The weights of the algorithm were also saved to provide snapshots of the algorithm during training. The following two diagrams show the categorical cross-entropy loss and the categorical accuracy against the test set during training.

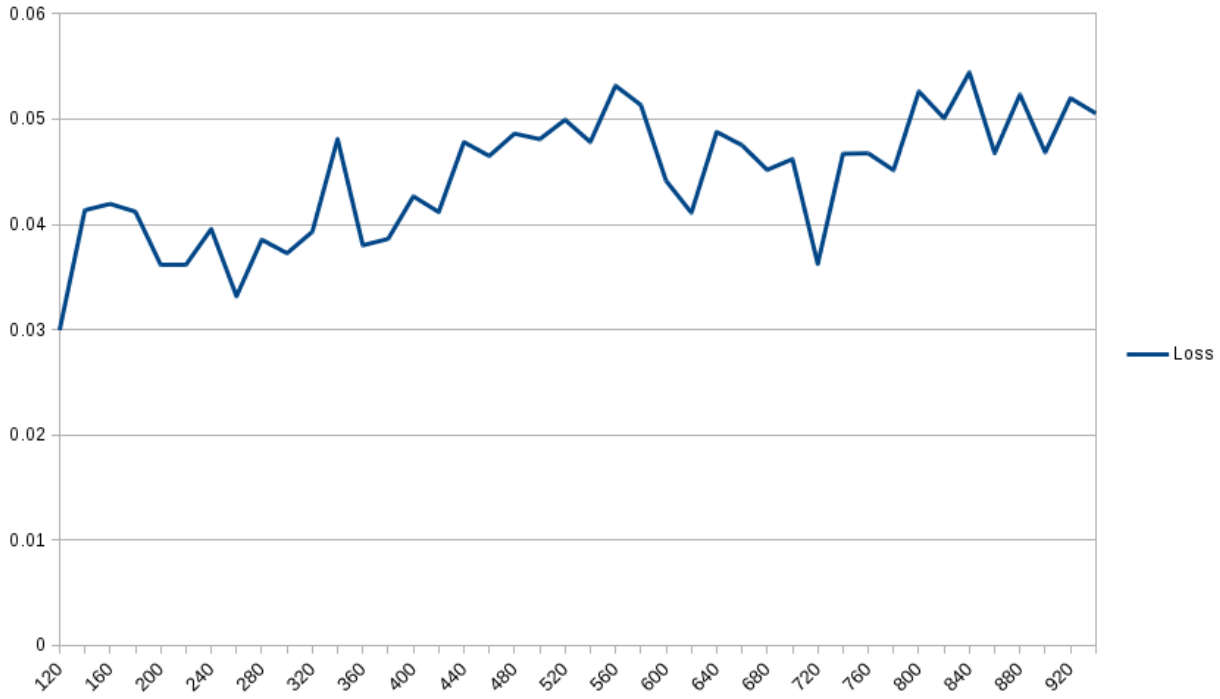

Fig. 5.11: Categorical cross-entropy loss every iteration of 20 epochs during testing.

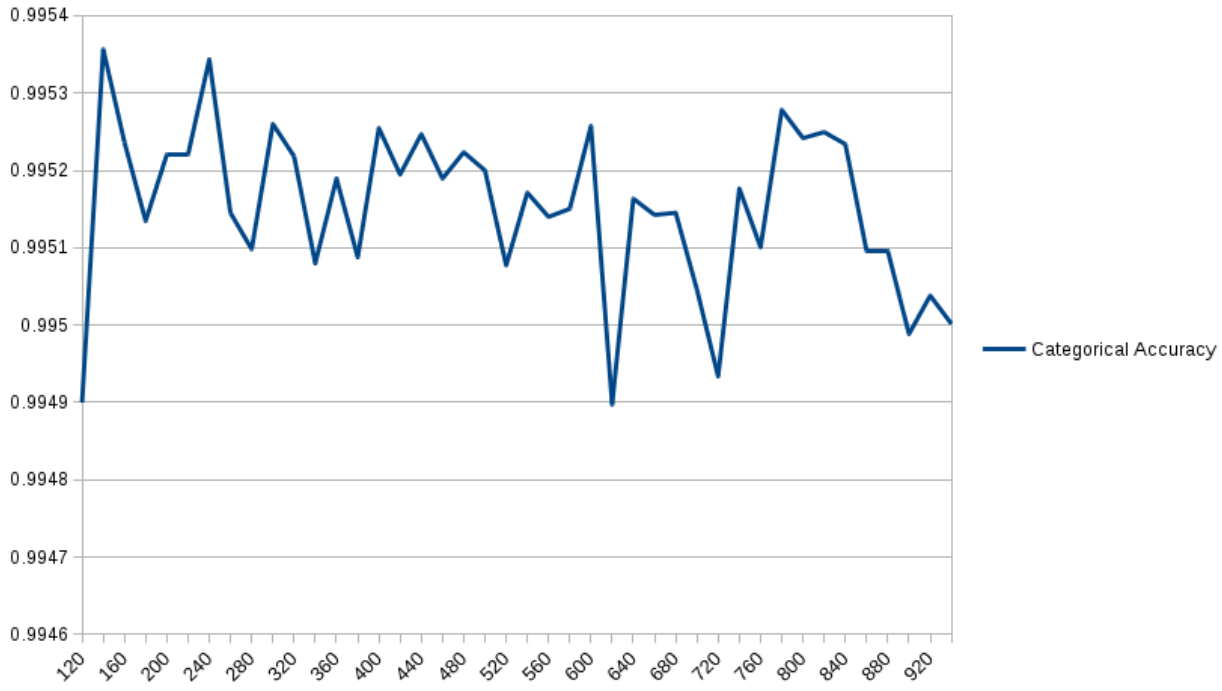

Fig. 5.12: Categorical accuracy loss every iteration of 20 epochs during testing.

The increase in loss (and decrease in accuracy) near the last epochs of training show slightly lower performance. This can happen if the algorithm is over trained on the training set and therefore less applicable to the general case as tested against the test data set. However the performance is high very quickly in training with the accuracy immediately over 99%.

5.4 Weights

The weights after training can be visualized to see how their values vary. In the LSTM there are 512 (LSTM output dimension) vectors of 1810 weights (the vector length).

The W_i weight vector is shown in the following diagram.

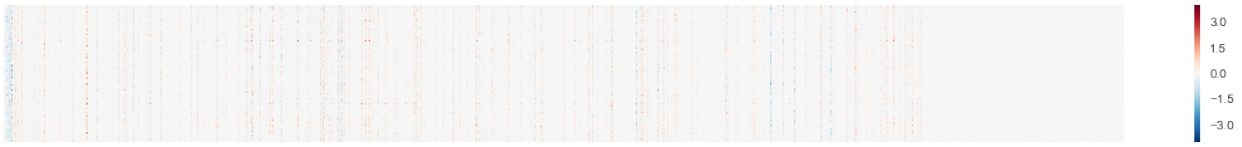

Fig. 5.13: W_i values after training is complete.

The above diagram has too many values to be visualized effectively. In the below diagrams the average over the 512 dimensions for each of the 1810 indexes is shown in the heatmap. This corresponds to the vertical lines shown in the above diagram.

Fig. 5.14: Average W_i values after training is complete.

Fig. 5.15: Average W_f values after training is complete.

Fig. 5.16: Average W_c values after training is complete.

Fig. 5.17: Average W_o values after training is complete.

Examining the indexes with extreme weights does not yield interpretable results. The calculations within the algorithm happen with so many steps, weights, and biases that a high value weight in any of the vectors may not correspond to the final output of the algorithm. For example the input weight vector W_i has it's highest weights at indexes that correspond to the country code for Angola, the destination name "Switzerland - Bebbicell Mobile", and the destination name "India - Other".

The U weights that are applied to the hidden state can also be visualized. These weights have two dimensions that are 512 elements long. The complete weight vectors are shown in the below heatmaps.

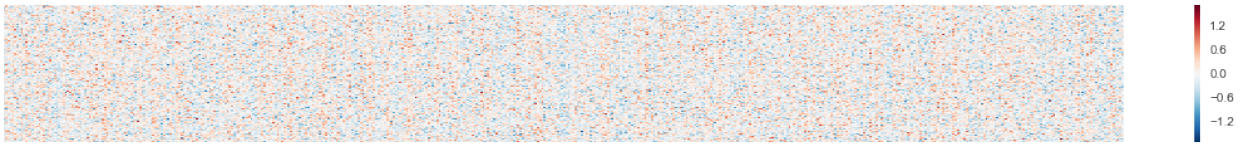

Fig. 5.18: U_i values after training is complete.

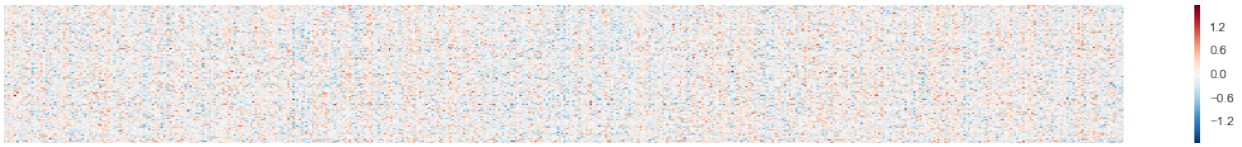

Fig. 5.19: U_f values after training is complete.

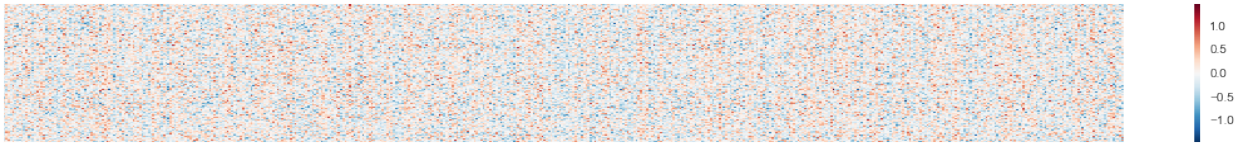

Fig. 5.20: U_c values after training is complete.

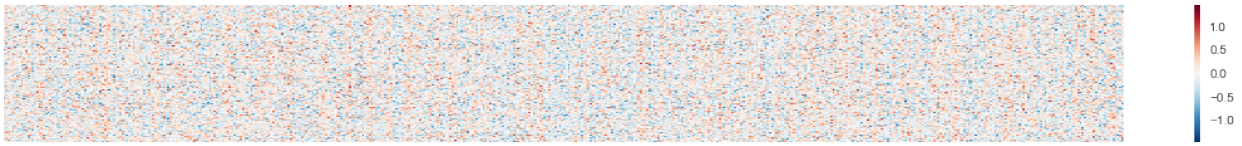

Fig. 5.21: U_o values after training is complete.

These weights are even less interpretable than the W weight vectors. These weights that are applied against hidden state do not show any obvious patterns in the heatmaps above.

Chapter VI

CONCLUSION AND RECOMMENDATION

An LSTM recurrent neural network that is trained and optimized can predict toll-fraud with over 99% accuracy. The results of this research are specific to the data provided by the VoIP provider. It's possible the algorithm could generalize to other providers, however another VoIP provider will have different users with different behaviours. This model could be trained on another VoIP providers data to test performance, although another VoIP provider is unlikely to provide their own usage data for research. This opens the possibility for future research by creating a common data set of VoIP usage and toll-fraud events on which to test various algorithms. Interpretability is a downside of using a recurrent neural network or most other artificial intelligence algorithms. There is no way to determine which call features are the most important to the model or if a simpler model can be made from attempting to interpret the learned weights. Another avenue for further exploration is to train an artificial neural network with input equal to the entire sequence of call vectors passed to the LSTM and comparing the performance and generalization of the algorithms. This will lose the sequence of specific call events while keeping together calls that occurred near each other in time.

This research is significant to the telecommunications industry because detecting toll-fraud accurately and quickly will reduce the damage to the service provider and VoIP users who are the victims. For the VoIP provider this research improves improves their detection rate and reduces the manpower necessary to monitor accounts. For the researcher this provides direction to further explore the problem area such as defining a large common data set to benchmark.

REFERENCES

[1] Richard A. Becker, Chris Volinsky, and Allan R. Wilks. Fraud Detection in Telecommunications: History and Lessons Learned. *Technometrics*, 52(1):20–33, feb 2010. URL: http://0-amstat.tandfonline.com.aupac.lib.athabascau.ca/doi/abs/10.1198/tech.2009.08136#.VRITand-Vs, doi:10.1198/TECH.2009.08136.

[2] James Bergstra, Olivier Breuleux, Frederic Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math compiler in Python. *Proceedings of the Python for Scientific Computing Conference (SciPy)*, pages 1–7, 2010. URL: http://www-etud.iro.umontreal.ca/~wardefar/publications/theano_scipy2010.pdf.

[3] Broadsoft. Accounting Call Detail Record Interface Specification. 2010.

[4] P. Burge. Fraud detection and management in mobile telecommunications networks. In *European Conference on Security and Detection - ECOS97 Incorporating the One Day Symposium on Technology Used for Combatting Fraud*, volume 1997, 91–96. IEE, 1997. URL: http://digital-library.theiet.org/content/conferences/10.1049/cp_19970429, doi:10.1049/cp:19970429.

[5] CFCA. 2013 Global Fraud Loss Survey. Technical Report, Communications Fraud Control Association, 2013.

[6] CRTC. Communications Monitoring Report 2014: Telecommunications Sector. Technical Report, CRTC, sep 2014. URL: http://www.crtc.gc.ca/eng/publications/reports/policymonitoring/2014/cmr5.htm.

[7] Wang Dong, Wang Quan-yu, Zhan Shou-yi, Li Feng-xia, and Wang Da-zhen. A feature extraction method for fraud detection in mobile communication networks. In *Fifth World Congress on Intelligent Control and Automation (IEEE Cat. No.04EX788)*,

volume 2, 1853–1856. IEEE, 2004. URL: http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=1340996, doi:10.1109/WCICA.2004.1340996.

[8] Abdikarim Hussein Elmi and Subariah Ibrahim. Detecting SIM Box Fraud Using Neural Network Abdikarim. 2012. URL: http://link.springer.com/10.1007/978-94-007-5860-5, doi:10.1007/978-94-007-5860-5.

[9] FBI. Wanted by the FBI. URL: https://www.fbi.gov/wanted/cyber/noor-aziz-uddin/view.

[10] FBI. Hacker Pleads Guilty to Infiltrating VOIP Networks and Reselling Services for Profit. 2010. URL: https://www.fbi.gov/newark/press-releases/2010/nk020310a.htm.

[11] FCC. Local Telephone Competition : Status as of June 30 , 2013. Technical Report June, FCC, 2014.

[12] Sharon Gaudin. Hacker admits stealing, reselling VoIP services. feb 2010. URL: http://www.computerworld.com/article/2520651/government-it/hacker-admits-stealing--reselling-voip-services.html.

[13] Felix A Gers, Fred Cummins, and Jürgen Schmidhuber. Learning to Forget: Continual Predication with LSTM. Technical Report, IDSIA, 1999.

[14] Google. Inceptionism: Going Deeper into Neural Networks. 2015. URL: http://googleresearch.blogspot.ca/2015/06/inceptionism-going-deeper-into-neural.html.

[15] Alex Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*. PhD thesis, Technische Universität München, 2008.

[16] Constantinos S. Hilas. Designing an expert system for fraud detection in private telecommunications networks. *Expert Systems with Applications*, 36(9):11559–11569, nov 2009. URL: http://www.sciencedirect.com/science/article/pii/S095741740900284X, doi:10.1016/j.eswa.2009.03.031.

[17] Constantinos S. Hilas and Paris As. Mastorocostas. An application of supervised and unsupervised learning approaches to telecommunications fraud detection.

Knowledge-Based Systems, 21(7):721–726, oct 2008. URL: http://www.sciencedirect.com/science/article/pii/S0950705108000786, doi:10.1016/j.knosys.2008.03.026.

[18] CS Hilas, PA Mastorocostas, and IT Rekanos. Clustering of Telecommunications User Profiles for Fraud Detection and Security Enhancement in Large Corporate Networks: A case Study. *Applied Mathematics & Information Sciences*, 1718(4):1709–1718, 2015. URL: http://www.naturalspublishing.com/files/published/ec1497291c9mqn.pdf.

[19] CS Hilas and JN Sahalos. User profiling for fraud detection in telecommunication networks. *5th International Conference on Technology . . .*, 2005. URL: http://icta05.teithe.gr/papers/69.pdf.

[20] Sepp Hochreiter and Jurgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1–32, 1997. doi:10.1162/neco.1997.9.8.1735.

[21] Dirk Hoffstadt, Erwin Rathgeb, Matthias Liebig, Ralf Meister, and Yacine Rebahi. A comprehensive framework for detecting and preventing VoIP fraud and misuse. In *2014 International Conference on Computing, Networking and Communications (ICNC)*, 807–813. IEEE, feb 2014. URL: http://0-ieeexplore.ieee.org.aupac.lib.athabascau.ca/articleDetails.jsp?arnumber=6785441, doi:10.1109/ICCNC.2014.6785441.

[22] IBM. IBM Watson: How it Works. 2015. URL: https://www.youtube.com/watch?v=_Xcmh1LQB9I.

[23] Eric Markowitz. The Inside Story Of How Pakistan Took Down The FBI's Most-Wanted Cybercriminal. 2015. URL: http://www.ibtimes.com/inside-story-how-pakistan-took-down-fbis-most-wanted-cybercriminal-1860808.

[24] Paris A. Mastorocostas, Constantinos S. Hilas, Dimitris N. Varsamis, and Stergiani C. Dova. Telecommunications call volume forecasting with a block-diagonal recurrent fuzzy neural network. *Telecommunication Systems*, jan 2015. URL: http://link.springer.com/10.1007/s11235-015-9968-x.

[25] Felipe Mata, Piotr Zuraniewsk, Michel Mandjes, and Marco Mellia. Anomaly detection in VoIP traffic with trends. sep 2012. URL: http://0-dl.acm.org.aupac.lib.

athabascau.ca/citation.cfm?id=2414276.2414279.

[26] Yacine Rebahi, Mohamed Nassar, Thomas Magedanz, and Olivier Festor. A survey on fraud and service misuse in voice over IP (VoIP) networks. *Information Security Technical Report*, 16(1):12–19, feb 2011. URL: http://www.sciencedirect.com/science/article/pii/S1363412710000373, doi:10.1016/j.istr.2010.10.012.

[27] Sebastian Ruder. An overview of gradient descent optimization algorithms. *Web Page*, pages 1–12, 2016. URL: http://arxiv.org/abs/1609.04747, arXiv:1609.04747.

[28] J.S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2003. ISBN 0137903952. URL: http://amazon.de/o/ASIN/0130803022/, doi:10.1017/S0269888900007724.

[29] Ilya Sutskever. *Training Recurrent Neural networks*. PhD thesis, University of Toronto, 2013.

[30] M. Taniguchi, M. Haft, J. Hollmen, and V. Tresp. Fraud detection in communication networks using neural and probabilistic methods. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98 (Cat. No.98CH36181)*, volume 2, 1241–1244. IEEE, 1998. URL: http://0-ieeexplore.ieee.org.aupac.lib.athabascau.ca/articleDetails.jsp?arnumber=675496, doi:10.1109/ICASSP.1998.675496.

[31] Pieter Tjerk De Boer, Dirk P. Kroese, Shie Mannor, and Reuven Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134(1):19–67, 2005. doi:10.1007/s10479-005-5724-z.

[32] Paolo De Lutiis and Dario Lombardo. An innovative way to analyze large ISP data for IMS security and monitoring. In *2009 13th International Conference on Intelligence in Next Generation Networks*, 1–6. IEEE, oct 2009. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5357065, doi:10.1109/ICIN.2009.5357065.