AJEENKYA
D Y PATIL UNIVERSITY
THE INNOVATION UNIVERSITY

ANOMALY DETECTION IN BLOCKCHAIN

STUDENT NAME:	**Sanjyot Mankar**
URN:	**2020-B-03022003**
COURSE NAME:	**Introduction to Windows Azure (IWZ)**
COURSE CODE:	**CSC310**
PROGRAM & SPECIALIZATION:	**BCA (MACT)**
ACADEMIC YEAR:	**2022-23**

Abstract

Anomaly detection has long been a well-researched field. Its applications in the financial sector helped identify suspicious activity by hackers. But with the development of the financial sector, such as block and artificial deceiving intelligence, financial systems is more difficult. Despite these technological advances, there are still many cases of fraud. Many artificial intelligence techniques have been proposed to address this anomaly detection problem; Some of the results seem quite convincing, but there is no clear better solution. The purpose of this thesis is to bridge the gap between artificial intelligence and blockchain using various anomaly detection techniques. From data from the transaction network of a public financial blockchain called Bitcoin. This thesis also provides an overview of blockchain technology and its application in the financial sector in light of anomaly detection. In addition, it accumulates bitcoin blockchain transaction data and malicious transaction analysis using unsupervised machine learning techniques. Several algorithms such as isolation forest, histogram-based outlier detection (HBOS), cluster-based local anomaly factor (CBLOF), principal component analysis (PCA), K-means, deep autoencoder networks and the ensemble method are evaluated and compared.

Introduction

Suspicious activities in network structures are as old as the invention of the network structure itself. Entities or their activities which tends to behave abnormally within the system is referred to as anomalies. Anomaly detection is widely used in a variety of cybersecurity applications for targeting financial fraud identification, network invasion detection, anti-money laundering, virus detection, and more. Usually, a common goal in these networks is to detect those anomalies and prevent such illegal activities from happening in future. With advancements in technology, blockchain emerges to plays a vital role in securing these network structures. Carlozo (2017) elaborates that a blockchain is a decentralised distributed database that maintains on-growing records and ensures that fraudulent records do not become part of this database and previously added records stay immutable. However, participants of a blockchain network can try to conduct illegal activities and, in some cases, succeedin deceiving the system into their advantage. Current state-of-the-art anomaly detection methodologies are designed and implemented in light of the centralised systems. With the advent of blockchain technology, it brings a need for anomaly detection

procedures within these systems as well. In this thesis, we extract and analyse the data from a publicly available blockchain named bitcoin. Nakamoto (2008) states that bitcoin is a peer-to-peer digital currency blockchain, by using which users can send and receive a form of electronic cash to each other anonymously without the need for any intermediaries. This thesis aims to detect anomalous or suspicious transactions in the bitcoin network, where all nodes are unlabeled, and there is no evidence that if any given transaction is an illicit activity. The primary focus is to detect anomalies within the bitcoin transaction network. Farren, Pham, and Alban-Hidalgo (2016) explained that the problem is related to the study of fraud detection in all types of financial transaction blockchain systems. The problem can also be generalised to other blockchain networks such as health service blockchains, public sector blockchains and many others, and therefore this thesis investigates a more general problem of anomaly detection in blockchains. Chapter 2 gives an in-depth background of blockchains and its procedures, and it also explains the problem statement along with some examples of problem use-cases. Chapter 3 discusses the history and development of anomaly detection research in the focus of distributed ledger and blockchains. Some research related to anomaly detection in financial systems is also discussed. explains in detail the unsupervised machine learning techniques that have been used for fraud detection in this thesis. Chapter 5 describes the technologies used to perform experiments and explains the data exploration and pre-processing along with the evaluation metrics used in this research. Chapter 6 describes the anomaly detection experiments and their evaluation results.

Background

Use of ledger in the financial area is as ancient as money itself. From clay and stones to papyrus and paper and later to spreadsheets in the digital era, ledgers have been an essential part of accounting journey. Early digital ledgers have just merely mimicked the paper-based approaches of double entry book-keeping. However, still, there remained a vast gap in consensus when these ledgers grew in volume and spatiality. The rise in computing power and cryptography breakthroughs along with the discovery of some new and sophisticated algorithms have given birth to distributed ledgers.

Distributed ledger technology (DLT)

In the crudest form distribute ledger is merely a database whose copy is independently held by each participant who wants to update it. These participants usually are nodes on a network and records written on these databases are usually called transactions. Distribution of these records is unique as they are not communicated to other nodes in the network using a central authority node. However, instead of that, each record is independently composed by each node individually. These leads to all the participant nodes of the network to process all incoming transactions and reach a conclusion about its authenticity. Finally, a consensus is achieved based on majority votes of conclusions of the network. Once there is this consensus achieved the distributed ledger is updated, and at this point, all nodes on the network maintain an identical copy of the ledger which holds all transactions. This architecture enables a new dexterity as the system of records can go beyond being a simple database. Distributed ledger is a dynamic type of media that possesses the attributes and capability go beyond legacy databases. The innovation of distributed ledgers allows its users not only to accumulate and communicate information in a distributed secure manner but also enables them to go beyond relationships among data. Ibanez etal. (2017) state that distributed ledgers provide a trustworthy, secure, and accountable way of tracking transactions without the need for a central validating authority, and they could provide the foundation to make the web a genuinely decentralised autonomous system.

Blockchain

The blockchain is a type of distributed ledger. Apparent from its name it is a chain of logically connected data blocks. A list of growing records refer to as blocks, and each block contains a timestamp, transactional data and a unique cryptographic hash. The hash in the block links it to the previous block in the distributed ledger all the way to form a chain of logical links to the first block called genesis block using these cryptographic hashes (Figure 2.1). The blockchain is a discrete design which makes it resistant to modification of the data as well as duplicate entries. Ian Siti and Lakhani (2017) said that it is an open-distributed ledger that can be utilised to document transactions among two unknown parties verifiably and permanently.

A blockchain usually uses a peer-to-peer network for seamless inter-node communication and validating new blocks. Once transaction data is appended to

the block, it cannot be altered retroactively without modification of all subsequent blocks. In order to alter any record, it would require a consensus of most network nodes, which is highly challenging to achieve. Although records of a blockchain are not immutable, still it may be considered the secure design and less prone to vulnerabilities. It also illustrates a distributed computing system with high byzantine fault tolerance. The decentralization and byzantine problem will be explained later in this chapter. Ravel and Siraj (2016) state that the blockchain claims to have achieved decentralized consensus.

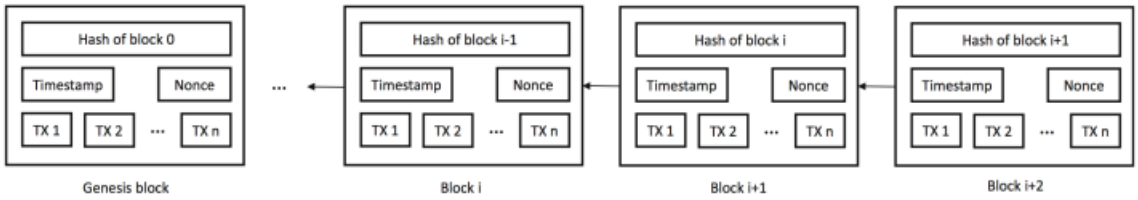

Figure 2.1. Example of a blockchain; Z. Zheng et al. (2017)

Hash graph

In the world of distributed ledgers, blockchain has been taking all the spotlight. However, blockchain comes with some design limitations. A consensus mechanism called proof of work (POW) which will be discussed later in this chapter under subsection protocols of blockchain is widely in use. As Bitcoin-Wiki (2016) explains, proof of work algorithm can be considerably time-consuming in nature. Another inherited inefficiency of blockchain design is that it consumes considerable amount electricity even when it decides to discard the blocks. As Baird (2016) explains, hash graph algorithm is a consensus mechanism based on the directed acyclic graph (DAG) which uses a virtual voting system combined with a gossip protocol described on Wikipedia (2018) to achieve consensus in a faster, more reliable, and secure manner. Hash graph intends to provide the benefits of blockchain as a distributed ledger but without the limitations. Contrary to many ledgers which only use the gossip protocol, hash graph utilises a voting algorithm in combination with gossip of gossips in order to reach consensus without using proof of work. This gossip of gossips is designed to share new information with other participant nodes of the network which are unaware of this new knowledge using hash graph itself. Veronese et al. (2013) states that this guarantees Byzantine Fault Tolerance (BFT) as all participants are part of the gossip events. Burkhardt, Welling, and Lasi (2018) explain that hash graph is DDoS attack resilient and provides high transaction

throughput, low consensus latency and proper absolute ordering of transactions. It works asynchronously to nondeterministic achieve consensus using

probabilities. Since hash graph is an early bird patented solution and is not opensource, its commercial applications are still under consideration in industry.

Blockchain network

Blockchain networks define the communication channel of participant network nodes. Gossip peer-to-peer network is the most used network, but with the rise in need of privacy, semi-gossiping networks and simple peer-to-peer networks are also utilised. A blockchain also defines its scope and protocols concerning its network with clarifies its intentions as a platform. Depending on the applications a blockchain scope and protocol may vary from a small-scale node infrastructure to a massive hub platform. This thesis aims to propose a possible solution for all blockchain scopes, but due to data availability reasons, a public blockchain is under study.

Scopes of blockchain

As mentioned by the Lin, Liao, and Lin (2017) there are three significant scopes of a distributed ledger or a blockchain:

• Public - A state of the art public blockchain is open source and permissionless. Participants do not require any permission to join the network. Anyone can download its source code or binaries and run it locally on his or her computer and start validating transactions coming to the network, consequently joining the consensus process. This gives everyone on the network the ability to process and determine which new blocks will become part of the chain. Participants of the network can send transactions onto the network and expect them to become part of the ledger if they are valid. In public blockchains transparency plays a key role, Each transaction on the blockchain ledger is visible to everyone and can be read using block explorer. However, these transactions are anonymous, so it is almost impossible to track the identity of the transaction owner. Milutinovic (2018) has explained Bitcoin, Ethereum and Litecoin which are some of the few examples of public blockchains.

• Consortium - Consortium blockchain regulates under the leadership of certain specific groups which follow same vision. Unlike public blockchains, they publicly do not allow everyone with internet access to become a participant of the network to verify the transactions. However, in some cases, the right to read

the blockchain can be defined by the groups so that the access to the blockchain can be either public, restricted or hybrid. Consortium blockchains are highly scalable and fast and provide more transactional privacy, so their practical applications are more welcomed in industries. According to Butlerin (2016), consortium blockchains are" partially decentralised". Usage of consortium blockchains is often in association with enterprises. Collaborating group of companies leverage blockchain technology for improved business processes. Its application is already making waves in healthcare, supply chain, finance, and other industries. Valent and Sander (2017) have mentioned that Quorum, R3 Corda and Hyperledger are some examples of consortium blockchains.

• Private - In most cases, private blockchains are developed, maintained, and kept centralised under an organisation and are not open sourced. Read permissions can be either public, restricted or hybrid based on system requirements. In this closed environment, external public audits are required to maintain the integrity of the system. Private blockchains take advantage of the technology while keeping the solution to themselves. A potential security risks is involved with the centralisation factor of private blockchains, but at the same time, it enables several advantages over public blockchains such as preapproved network participants and known identities. MONAX and Multichain are few known examples of private blockchains mentioned by Sanjana, Sindhu, and Seth Madhavan (2018).

Consensus of blockchain

Skvorc (2018) elaborates that there are three primary consensus mechanisms of a distributed ledger or a blockchain:

• Proof of work (POW) - To keep the decentralised system secure and robust, proof of work plays an important role. The idea behind proof of work is to make participants of the network approve actual transactions and disapprove fraudulent ones. Approved transactions are added to a block which later becomes part of the blockchain ledger, and for this, the network rewards the participants. At core proof of work solely depends on computing power. Some participant nodes of the network engage in a competition known as the mining process to finding a hash called nonce (number used once). This hash is an extract of solving a complex mathematical problem that is part of the blockchain program. Combining this hash with the data in the block and then passing it through the hash function produces a result in a certain range which can become part of the blockchain. Since hash function makes it impossible to predict the output, so the participant nodes must guess to find this hash. The node which

finds the hash first is allowed to add its block to the blockchain and receive the reward. This reward usually comprises tokens that user can utilise on the same network. Amount of the reward is defined dynamically within the blockchain program and can mutate over time.

• Proof of stake (POS) - Proof of stake proposes to overcome the computing race in proof of work. In proof of work participants of the network can join to create a pool of nodes in order to mine blocks faster and collect rewards. This network pool ends up utilising more electricity and approaches toward a more centralised network topology. Instead of letting all the participants competing for mining blocks proof of stake uses an election process in which a participant is selected to validate the next block. These selected participants are called validators. To become a validator a node must deposit a certain number of tokens into the network as a stake of security collateral. Size of the stake determines the chances of a validator be chosen to forge the next block. A validator node checks for all the transactions within a block are valid and signs the block before adding it too the blockchain, consequently receiving the reward. Trust of the validator nodes depends on their deposited stake; validators will lose a part of their deposit if they approve fraudulent transactions.

• Proof of Authority (POA) - With popularity in private blockchains, proof of authority uses a voting algorithm to validate the blocks. A group of known and authorised nodes votes on which transaction should be approved and added to the block. This results in higher throughput and shorter verification time compared to proof of work. Many industries back this consensus mechanism due to its control over the system approvals. However, to reach the accurate decentralisation decisions of trusted participant nodes should not be influenced by anyone.

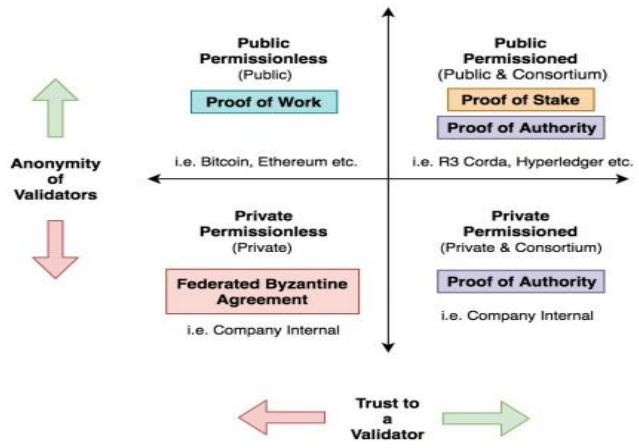

Figure 2. Scopes and consensus mechanisms of a blockchain.

Blockchain security

The core idea behind blockchain security is to allow people, particularly those who do not trust one another to share valuable information in a secure and tamperproof manner. Sophisticated decentralised storage and consensus algorithms make it extremely difficult for attackers to manipulate the system. Encryption plays an essential role in the whole process and validates the communication channels of the network. Another critical problem blockchain claims to resolve is byzantine agreement. It defines the dependability of the system in case of failure.

Decentralization

Centrally held data is highly prone to a variety of risks. Blockchain eliminates this risk by storing data in a decentralised fashion on a peer-to-peer network. It is hard to exploit the vulnerabilities of a peer-to-peer network due to lack of centralised points. Moreover, it also ensures that there is no single point of failure in the system. Brito and Castillo (2013) explain that blockchain uses cryptographic public-private keys to communicate ensuring privacy and security of network nodes. Usage of private keys to encrypt and public keys to decrypt makes blockchain highly secure. The Economist (2015) stated that generally, the data on blockchain is incorruptible. Data quality is also ensured with decentralisation as there is no official, centralised copy of a record and trust of each node is on the same level.

Byzantine Problem

A Byzantine problem, as explained by Lampert, Shostak, and Pease (1982), states that computers on a decentralised network cannot be entirely and irrefutably guaranteed that they are displaying the same data. Considering the unreliability of the network nodes can never be sure if the data they communicated has reached its destination. At its essence byzantine problem is about achieving consensus across a distributed network of nodes, while few nodes on the network could be possibly faulty and may also attempt to attack the network. A Byzantine node can mislead and may provide corrupted information to other nodes involved in the consensus process. Blockchain must operate with robustness in such situation and reach consensus despite the interference of these byzantine nodes. Even though Byzantine problem cannot be fully solved, different blockchains and distributed ledgers utilise various

consensus mechanisms like proof of work and proof of stake to overcome the problem in a probabilistic manner.

Bitcoin

The earliest known design of a cryptographically chained data linked in chains and blocks was by Haber and Stornetta (1990). They purposed a system in which documents time stamps could not be tempered or backdated. Later on Bayer, Haber, and Stornetta (1993) added Merkle trees to the design which enhanced efficiency and allowed accumulation of multiple documents into a block. However, the first ever conceptualisation of a blockchain was introduced by a person with pseudonym Nakamoto (2008). Consequently, giving birth to the first blockchain system. This blockchain system was named 'Bitcoin' and is designed as peer-to-peer electronic cash system otherwise known as a cryptocurrency. Bitcoin serves a public ledger for all transactions of the network and can represent currency digitally hence working as electronic cash. A digital representation of an asset can lead to multiple problems. Bitcoin is developed to tackle some significant problems. Firstly, in theory, the digitally represented asset can technically be duplicated which is not acceptable. As this replication of a digital asset can confuse actual ownership of the asset. Secondly, the replicated asset can be spent multiple times resulting in chaos in the system, Usman W. Chohan (2017) states that this problem is also known as the double spend problem. Bitcoin solves this problem by using proof of work (POW) algorithm (Fig 2.3). Utilising proof of work mechanism miners verify transactions for their authenticity and get rewarded in newly generated bitcoins.

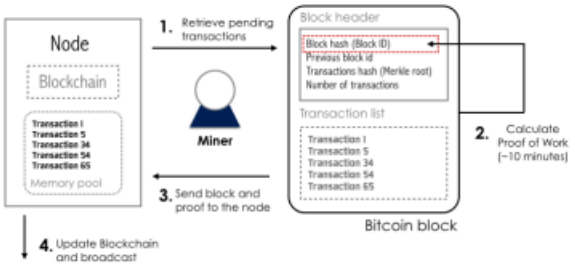

Figure 2.3. Proof of work illustration; Barrera (2014)

However, bitcoin blockchain is designed with some specific rules, such as there can be only 21 million bitcoins. As fig 2.4 illustrates, each transaction is encrypted with the private key of the sender. This transaction is enclosed with a digital signature and public key of the receiver and sent to the receiver address. This technique allows the asset to be securely delivered only to the receiver address and can be verified using the digital signature. Identities of sender and

receiver are anonymous as they can generate new addresses before initiating a new transaction.

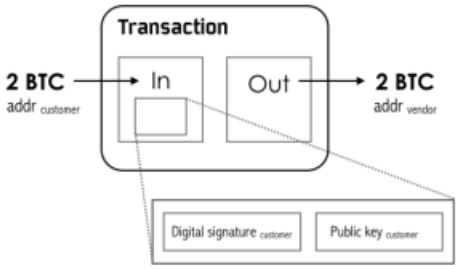

Figure 2.4. Simple bitcoin transaction; Barrera (2014)

Bitcoin transactions can consist of one or more inputs and one or more outputs. When sending bitcoins to an address, a user designates the address and the amount being sent and encapsulates it in the output of transaction in order to keep track of double spending. As mentioned by Delgado-Segura et al. (2018) it is known as unspent transaction output (UTXO) model, which implies that each transaction input is a pointer to the output of the previous transaction. Figure 2.5 shows an example where a customer sends 2 BTC to the address of the vendor, then the vendor sends 0.8 BTC of those 2 BTC to the provider address, and change of 1.2 BTC is the returned to the vendor.

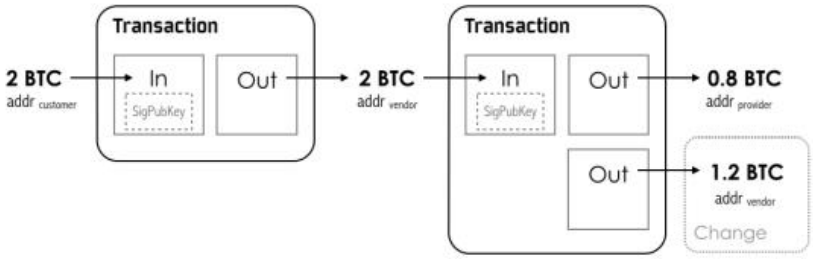

Figure 2.5. Linked bitcoin transactions; Barrera (2014)

There are many elements involved when participants of network transmit bitcoins. However, we can break down the activities in two groups; User activities and Transaction activities. In this thesis, our prime interest is in transaction activities and its analysis for suspicious behaviour. Transactions on bitcoin blockchain are more complex in depth but since the scope of this thesis is to analyse the bitcoin transactional data this thesis sheds light on the analytical perspective of bitcoin data. Chapter 5 discuss the anatomy of transaction activities regarding elements of data and structure.

Anomaly detection methods

Anomaly detection is categorized into three main types of learning methods; Supervised learning methods, Semi-Supervised learning methods and Unsupervised learning methods. Majority of real-world anomaly detection problems fall under unsupervised learning domain such as credit card fraud detection, network intrusion detection and our blockchain fraudulent transaction detection. This thesis deems supervised and semi-supervised learning methods out of the scope and aims to focus entirely on unsupervised learning methods. Unsupervised learning is the most flexible type of learning, as it does not require any class labels for training. It scores the data based on its intrinsic attributes such as distance, density or more and attempts to distinguish the anomalies from the standard data based on estimation. As illustrated in Figure 4.1, most unsupervised anomaly detection algorithms can be roughly categorized into the following main groups Chandola, Banerjee, and Kumar (2009). This illustration is updated and differs from the original publication. Figure 4.1. A taxonomy of unsupervised anomaly detection algorithms that are used most in practice; Goldstein and Uchida (2016) This thesis selectively implements a set of unsupervised anomaly detection methods that can handle a large amount of data and can compute results in a reasonable time. Following sub-sections of this chapter will explain these algorithms in more depth.

Isolation Forest

Liu, Ting, and Zhou (2008) published an algorithm called" isolation forest" for unsupervised anomaly detection that achieved much popularity in recent years. The idea behind isolation forest is that it is comparatively more straightforward to isolate anomalies from the data than to build a model that could estimate normal behaviour. In order to isolate an anomalous observation, the dataset is randomly and recursively partitioned until the only data point left in the partition is the observation. A tree structure is used to represent the recursive partition. The base of the isolation forest algorithm is on the ensemble of isolation trees, A forest of random isolation trees is constructed to estimate the measure of normality and abnormality of observations in the dataset. An isolation forest is a combination of isolation trees, and these trees are actual binary trees in which each node has either zero or two child nodes. Let N be a node which is either a leaf node with no children or parent node with two child nodes Nl and Nr To decide which children nodes go to which parent node a test is associated with node N. This test involves selecting a random feature q from the data and a random split point p such that the nodes $p < q$ are under Nl and $p \geq q$ are under Nr Multiple random trees are formed recursively, and data is

partitioned in each one of them unless all observations are isolated for each iteration. The path length from the root node of the tree to the leaf defines the number of partitions required to isolate that observation. It is observed that for randomly partitioned data, the path length of anomalies is shorter than of standard observations. One of the reasons for this phenomenon to occur is that usually the number of anomalies in a dataset is smaller than the number of standard observations, which results in a shorter number of partitions and hence makes isolation of anomalous observation easier. Another reason for anomalous observations to be separated in earlier partitioning is that they have distinct attribute values as compared with standard observations.

Figure 4.2 illustrates that anomalies are more sensitive to isolation. The figure below depicts the anomalous observation xo requires only four partitions to be isolated, whereas isolating standard observation xi requires twelve random partitions. Given a dataset

$X = \{x1, x2, \ldots, xn\}$

containing n observations, random recursive isolation trees are built until each tree reaches the hight limit of $|X| = 1$ or all data in X have the same value. Presuming majority of data in X is distinct, and the isolation tree is fully grown, each observation is isolated to a leaf node. Then the tree will have n leaves and n-1 internal nodes, in total 2n-1 nodes. The path length of an observation x can be calculated by heuristic h(x) which is a measurement of the count of edges it takes to traverse from the root node to the leaf node. In order to use the isolation forest as an anomaly detection technique, it has to generate comparable anomaly scores as h(x) cannot be directly used as an anomaly score. It

Figure 4.2. The visual difference between the isolation of a standard observation xi versus an anomalous observation xo; Liu, Ting, and Zhou (2008) is because while the maximum height of a tree grows in the order of n, the average height grows in the order of log n. Isolation trees have a similar structure as binary search trees (BSTs). Therefore, the average path length of a failed search in a binary search tree is comparable to the average of h(x). As stated by Preiss (2008), the average path length of a failed BST search in a tree with n nodes can be calculated as

$$c(n) = 2H(n-1) - (2(n-1)/n) \quad (4.1)$$

where $H(i) \approx \ln(i) + 0.5772156649$ (Euler's constant).

The anomaly score s for an observation x is defined as:

$$s(x, n) = 2^{-E(h(x))c(n)} \quad (4.2)$$

Here E(h(x)) is the mean of all the h(x) from all isolation trees. While training an isolation forest algorithm, isolation trees are randomly created as explained above. Training requires two parameters: the number of trees t and the sub-sampling size. Tree height limit l is calculated based on the sub-sampling size. The sub-sampling process controls the data size for training and is an essential variable as it can directly affect the performance of the classifier. As a different sub-sample is picked to create an isolation tree each time, it can help isolate the anomalies in a better way. As for anomaly prediction from the trained model, Equation (4.2) is used to calculate an anomaly score s as explained above. This anomaly score s ranges between the values from [0,1] where values being more close to 1 are more likely to be anomalies.

Histogram-based Outlier Score (HBOS)

Goldstein and Denel (2012) introduced an unsupervised algorithm for anomaly detection. Histogram-based Outlier Score (HBOS) is a statistics-based method designed to handle large datasets. Its computation complexity is an attractive attribute for researchers. A univariant histogram is created for each data feature. For categorical data frequency of values for each category is calculated, and the height of the histogram is computed. For numerical features, either static bin-width histograms technique or dynamic bin-width histogram technique is used. The static bin-width method uses k equal width bins over a value range, and the height of the bin is estimated by counting the samples which fall in each bin. The dynamic bin-width is determined by sorting all data points and then grouping a fixed number N/k of consecutive values, where N represents the number of total observations k represents the number of bins. Both static and dynamic approaches are introduced in this algorithm because real-world data have very different feature distributions, especially when features have a significant range of gaps in between. A fixed bin may estimate the density of anomalies poorly resulting in adding most of the data to a few bins. Anomaly detection problems usually have considerable gaps in the data range as most outliers are far away from standard observations. The authors mostly prefer the dynamic method if data distributions are unknown. The technique used to determine the value of k bins is the square root of instances N. A particular histogram is computed for each dimension d, and density is estimated for the height of each bin. Normalization is applied over histograms so that their maximum height is 1.0. By doing so each feature is ensured to be weighted equally for anomaly score. HBOS for each observation p is computed using equation

(4.3)

HBOS(p) =Õdi=0log 1histi(p)

(4.3)

An inverse multiplication of the estimated densities defines the score, assuming independence of features in similar manner to Kim et al. (2004). This score can alsobe represented as an inverse of discrete Naive Bayes were instead of multiplication = a sum of logarithms is used, using the fact that log(a·b) = log(a) + log(b). The reason for this trick is to make the HBOS algorithm less susceptible to errors due to floating-point precision, which can cause very high anomaly scores for extremely unbalanced distributions.

Deep Autoencoders

Artificial neural network algorithms are relatively old but have gained popularity recently due to significant breakthroughs in technology. An autoencoder is a particular type of artificial neural network that attempts to replicate its input as closely as possible to its output. Concisely, its an algorithm that can learn a way to regenerate what is fed to it. J. Zheng and Peng (2018) state in their research that autoencoders achieve that by determining the encoding of input data. It allows them to reconstruct the input data using a computed encoding. As figure 4.3 illustrates the schematic structure of an autoencoder can be broken into two parts encoder and decoder. Figure 4.3. Schematic structure of an autoencoder with 3 fully connected hidden layers; Commons (2016). The transition from the first layer X of the network to the middle layer z of the network represents encoder, and the transition from the middle layer of the network to the rightmost last layer X’ of the network represents decoder. The first layer of an autoencoder neural network is an input layer X, which may consist of m nodes, the same number of nodes as the dimensions of the data. The middle layer of the network z may have n nodes, where n represents the dimensions of encoding. The rightmost layer X’ is the output layer and similar to the input layer shall have m nodes. As explained by Wang, Yao, and Zhao (2016) autoencoders are usually used for dimensionality reduction so that the dimensions of the learned encoding is usually smaller in size than the input. As stated by the J. Zheng and Peng (2018) equation (4.11) represents the encoder e in the autoencoder and equation (4.12) represents the decoder d in the autoencoder. In the equations below W is an n x n matrix consisting of weights in the layers of neural network. b is the vector with size n which contains the bias of the layer and x is the input vector of length n. The σ represents a nonlinear transformation function such as sigmoid.

$e = \sigma (W_{enc} x + b_{enc})$ (4.11)

$d = \sigma (W_{dece} + b_{dec})$ (4.12)

Deep autoencoders are neural networks with more complex structures such that each layer of encoder reduce the dimensions of the input to determine the hidden encoding behind the data. Vice versa each layer of decoder enlarges the dimensions until it reaches the input size. Autoencoder is an unsupervised learning algorithm that can encode and decode data while minimizing the error using the Hecht-Nielsen (1992) backpropagation algorithm. The reconstruction error is the metric used to quantify the similarity between the input and output of autoencoder. To detect anomalies unlabelled data is given to the autoencoder for training, and it tries to minimize the reconstruction error over the training set. One of the most common reconstruction errors used is Lehmann and Casella (2006) mean squared error. Autoencoder tries to fit the model onto the normal dataset without anomalies, and since the autoencoder has not encountered any anomalies, the reconstruction should perform better on normal observations than on anomalies. Hence, we can differentiate the anomalies from the standard data by calculating the reconstruction error. Typically, the reconstruction error of an anomaly should be higher than the reconstruction error of normal observation. A threshold value α can be determined, which can set the boundary for classifying anomalous observations.

Research Results

This chapter explains the evaluation results for all algorithms and how they were obtained. Various evaluation metrics explained in section 5.4 were used to perform the assessment. Due to the vast amount of data, the time complexity for some algorithms was enormous. However, to tackle the problem, we randomly subsampled 1/10th of the training data and fit 20 different models with fine-tuned hyper-parameters. Parameters such as sub-sampling 1/10th of data and training exactly 20 models were calculated based on experiments and the values were finalized when the model's performance stopped changing even when the sample size or the number of models were increased.

Conclusion

Evaluation of multiple unsupervised algorithms to detect anomalous behaviour in an unspent transaction output (UTXO) based blockchain shows that malicious activities can be successfully identified. This goes in line with other

research on anomaly detection in blockchains. If the evaluation metrics of previous researches were disclosed, the comparison would have been even more apparent. Although it can be indirectly compared to credit card fraud detection, the dynamics of data are entirely different and hence induce a variety of complex challenges. It is possible that better results would have been obtained with more malicious data. The small size of malicious transactional data points in the dataset means that there is little anomalous patterns to recognize. An attempt was made to overcome this problem by synthetically generating malicious data points while making sure that it is not overdone. However, this affects the quality of anomaly detection, which leads to weaker performance of the models. The selection of extracted features from the blockchain transaction graph to determine anomalous behavior seems an essential factor as certain variables may hold a unique prospect. Such as when treated as a time-series problem, it may disclose different solutions. However, this thesis research focuses on finding anomalous patterns using unsupervised learning techniques, and having a time-series element in the problem would increase the complexity of the scope. Nonetheless, researchers have experimented with various unsupervised techniques to detect anomalous transactions in a blockchain and discovered somewhat successful ways to achieve it. However, the lack of relevant data and computing power limitations, along with algorithm's data handling capabilities, induce complex challenges. Anomaly detection within blockchains can improve in the future if certain elements progress, such as the availability of relevant and rich data. Feature engineering and approaching the problem from another prospect such as time-series or network analysis can unlock new potential solutions. Additionally, distributed computing can be utilized to enhance algorithm's computing capabilities.

References

1. Arthur, David and Sergei Vassilvitskii (2007). "k-means++: The advantages of careful seeding". In: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics, pp. 1027–1035.
2. Baird, Leemon (2016). The Swirlds Hashgraph Consensus Algorithm: Fair, Fast, Byzantine Fault Tolerance. English.
3. Barrera, Alex (Apr. 2014). A Guide to Bitcoin (Part 1 and 2): A look under the hood. url: http://tech.eu/features/808/bitcoin-part-one/ (visited on 10/16/2018).

4. Bayer, Dave, Stuart Haber, and W Scott Stornetta (1993). “Improving the efficiency and reliability of digital time-stamping”. In: Sequences II. Springer, pp. 329–334.
5. Bentley, Jon Louis (1975). “Multidimensional binary search trees used for associative searching”. In: Communications of the ACM 18.9, pp. 509–517.
6. Bitcoin Forum (2014). Bitcoin Forum. https://bitcointalk.org/index.php?topic=576337. [Online; accessed 7-September-2019].
7. Bitcoin-Wiki (May 2016). Proof of work. url: https://en.bitcoin.it/wiki/Proof of work (visited on 08/30/2018).
8. Blockchain.com (2018). Stone Man Loss Transaction Visualization. [Online; accessed 12-November-2018]. url: https://www.blockchain.com/btc/tree/120749. (2019). Blockchain.com. https://www.blockchain.com/en/charts/n- transactionstotal?timespan=all. [Online; accessed 7-September-2019].
9. Bogner, Andreas (2017). “Seeing is understanding: anomaly detection in blockchains with visualized features”. In: Proceedings of the 2017 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2017 ACM
10. International Symposium on Wearable Computers. ACM, pp. 5–8.
11. Bolton, Richard J, David J Hand, et al. (2001). “Unsupervised profiling methods for fraud detection”. In: Credit Scoring and Credit Control VII, pp. 235–255.
12. Breiman, Leo (2001). “Random forests”. In: Machine learning 45.1, pp. 5–32.
13. Breunig, Markus M et al. (2000). “LOF: identifying density-based local outliers”. In: ACM sigmod record. Vol. 29. 2. ACM, pp. 93–104.
14. Brito, Jerry and Andrea Castillo (2013). “Bitcoin: A primer for policymakers”. In: Policy: A Journal of Public Policy and Ideas 29.4, pp. 3–12.
15. Brugere, Ivan (2013). Bitcoin Transaction Network Dataset. [Online; accessed 29-November2018]. url: http://compbio.cs.uic.edu/data/bitcoin/.
16. Burkhardt, Daniel, Maximilian Werling, and Heiner Lasi (2018). “Distributed Ledger”.English. In: IEEE, pp. 1–9.
17. Buterin, Vitalik (Aug. 6, 2016). On Public and Private Blockchains. url: https://blog.ethereum.org/2015/08/07/on-public-and-private-blockchains/ (visited on 09/26/2018).