UNIVERSITY OF NAIROBI

SCHOOL OF COMPUTING AND INFORMATICS

APPLICATION OF METALEARNING TO DETECT FINANCIAL STATEMENTS FRAUD IN ORGANISATIONS

GATHUMBI COLLINS GITHIARI

P52/32084/2019

Supervisor

Dr Lawrence Muchemi

A Project Proposal Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computational Intelligence, School of Computing and Informatics, University of Nairobi.

August, 2021

Declaration

I hereby declare that this thesis is my own work and has, to the best of my knowledge, not been submitted to any other institution of higher learning.

Signature: **Date: 26th August 2021**

Name: Gathumbi Collins Githiari **Registration Number:** P52/32084/2019

This research project has been submitted for examination towards fulfillment for the award of degree of Masters in Computational Intelligence with my approval as the supervisor.

Name: Dr Lawrence Muchemi

School of Computing and Informatics

Signature: .... **Date:** 1st Sept 2021

ABSTRACT

Financial statements fraud detection techniques have been classified into various categories and this study will focus on one of them: artificial and computational intelligence techniques. One of the major challenges facing financial statements fraud detection is that financial data needed to train detection models, is hugely unavailable due to regulations that prohibit the transmission and distribution of the highly confidential data.

The aim of this research is to come up with a fraud detection technique that overcomes the challenge faced in fraud detection of unavailability of financial data. This research was conducted through first finding out which features of financial statements are key to financial statements fraud detection. An experiment was also done to find out which hybrid algorithm performs best at detecting fraud in financial statements. The features standing out from the first experiment would form the feature set, and a model built on the algorithm that performs best.

The results showed that 3 of the top features were related to the assets of the business and out of all the 20 features identified, ones dealing with assets were 7 in total. Ensemble methods showed great accuracy when it comes to classification tasks that have a high dimensionality as all the methods scored 80% and the best performing being Reptile at 87.86%. The model built on reptile algorithm and trained using the identified feature set had an accuracy of 86.33%.

The key limitation of this research is the inaccessibility of financial statements data in the public domain. It is even harder to find these statements where fraud has occurred as efforts will have been put into place to conceal the presence of such fraud.

The project concludes that metalearning algorithms are performing better than other algorithms where the data to train is limited as is in this case, and that feature selection is important to increasing the accuracy of the model.

This study contributes to the knowledge of how to accurately detect fraud occurrence in financial statements by providing an insight as to which features of these statements are more key in indicating the possibility of fraud. The research also shows how key metalearning algorithms are in this new technological era, this being backed up by the accuracy of the two metalearning algorithms as they emerged the top two from all the hybrid algorithms identified.

Key words:

Reptile, financial statements fraud, fraud detection, metalearning, neural networks.

DEDICATION

I dedicate this work to my parents and sister for always believing in me, supporting me and always being there with me.

ACKNOWLEDGMENT

I am foremost grateful to God for giving me the self-discipline, good health, and dedication to complete this research study.

My gratitude goes also to my supervisor, Dr Lawrence Muchemi, for his supportive mentoring during this study. The advice, feedback, and encouragement he has given me has kept me focused on doing a good job on the study.

I am indebted to my fiancé, Isabel, for providing me with encouragement and support, which has been needed during the period of this study. Her efforts to keep me motivated are deeply appreciated.

To my mother Ms. Regina, my father Mr. Francis, and my sister Darlene for all the support and prayers they have offered. They are duly appreciated.

To all the SCI staff, colleagues and friends who have lent a helping hand in one way or the other, I thank you all for your support.

List of Figures

List of Tables

Definition of Terms

Financial Statements – companies' basic documents to reflect their financial status.

Financial Fraud – an intentionally deceitful action designed to provide the perpetrator with unlawful gain.

Metalearning - designing new models that can learn new skills or adapt to new environments fast and with a few training examples**.**

Business Intelligence – combines business analytics, data mining, data visualization, data tools and infrastructure and best practices to help organizations to make more data-driven decisions.

Probabilistic Neural Network - this is a feedforward neural network that is widely used in classification and pattern recognition problems.

Depreciation index – is used to judge whether companies are depreciating assets faster or slower.

Wc Accruals – this is the year over year difference in net current operating assets.

RSST Accruals – this measure shows changes in long-term operating assets and long-term operating liabilities.

Book to market – this is a ratio that compares a company's book value to its market value.

Actual issuance – these are a set of securities that a company or government offers for sale.

Table of Contents

1. INTRODUCTION

1.1 Background

Fraud is a willing act of using deceit to provide the perpetrator with gain unlawfully. It can occur in finance, investment, and insurance (Chen, 2020). Common fraudulent schemes are identity theft, income, or asset falsification.

Financial statements are documents that show the financial status of a company. Some reasons behind financial statements fraud are to make the business appear more profitable, to show improvement in performance or to reduce tax obligations (I.SADGALI 2019).

Financial statements fraud detection techniques have been classified into categories such as descriptive/unsupervised techniques which focus on relations and interconnectedness, predictive techniques which predict target objects such as fraud occurrence and artificial and computational intelligence techniques (I.SADGALI 2019).

In the Big Data era, detecting fraud has proven to be a challenge and one approach that has been increasingly used to analyze relations and connectivity patterns is Graph-based anomaly detection (GBAD) (Pourhabibi, et al. 2020). Research was conducted which analyzed studies published from 2007 to 2018 and showed a growing trend of GBAD techniques for fraud detection (Pourhabibi, et al. 2020). The research showed that approximately 87.2% of the reviewed research have exclusively developed their models using unsupervised learning techniques. This was driven by the fact that data labels are often in short supply or even nonexistent in fraud detection.

Most studies appreciate the fact that prior research on fraud detection has faced the challenges of accessing internal data (auditor-client relationships, personal and behavioral characteristics) since this data is not readily available to investors, auditors, and regulators (Ahmed Abbasi 2012). Privacy concerns have led to stakeholders and organizations being reluctant to share their fraud information. The direct impact of this is that there is a hinderance of research and this affects the integrity of the experiments conducted (Pourhabibi, et al. 2020).

It is apparent that fraud detection is faced by the challenge of not having enough data to improve with. This leads to the need for a fraud detection technique that can work and self-improve with little data. A technique that is standing out in this aspect is metalearning.

Metalearning aims to design new models that are capable of adapting to new environments fast and learning new skills quickly as well. A good metalearning model can adapt to new tasks and environments with limited exposure to new task configurations (Weng 2018). The usual approaches to metalearning are model-based, metric-based, and optimization-based.

A metalearning framework called Metafraud was proposed to address research gaps that exist due to unavailability of data.

This research shall aim to increase the accuracy of fraud detection using metalearning to overcome the challenge that modern fraud detection techniques are facing, that is unavailability of financial data due to its sensitive nature.

Many fraud schemes have led to losses that span several years and this has shown that existing fraud detection and detection mechanisms are ineffective. Enhanced financial fraud detection capabilities will also greatly benefit the stakeholders, these being: audit firms, investors, and government regulators (Abbasi 2012).

Financial fraud has been shown to cause serious consequences for organizations in terms of their long-term sustainability as well as affect the employees and the economy as well. Research indicates that not only is the risk of experiencing fraud high, but it is also increasing (Abbasi 2012).

1.2 Problem Statement

Fraudsters have been adapting over time and in doing so, they invent new ways of beating fraud detection systems. By doing this, financial fraud continues to grow (I.SADGALI 2019).

Fraud in financial statements is difficult to detect, and even after detection, the damage already inflicted is serious (Yang, et al. 2020). Therefore, efficient, and effective measures to detect financial statements fraud would offer important value to regulators and other stakeholders.

One of the major challenges facing fraud detection is that financial data needed to train detection models, is hugely unavailable due to laws that prevent the highly confidential financial data to be transmitted or distributed (Pourhabibi, et al. 2020).

The purpose of this research is to find a way to overcome this challenge by improving on the current metalearning based fraud detection techniques that are currently being used.

1.3 Objectives

1.3.1 Overall Objective

The aim of this research is to come up with a fraud detection technique that overcomes the challenge faced in fraud detection of unavailability of financial data due to its confidential nature.

1.3.2 Research Questions

More specifically, the research questions to be addressed are:

1. How do the different features of financial statements affect the accuracy in detecting financial fraud?
2. What is a modern classification algorithm that can be used to identify financial fraud?
3. How can these key features be used to create a prototype that is more accurate in detecting financial fraud?
4. How effective is the model created in detecting financial fraud?

1.3.3 Specific Objectives

The research has the following sub-objectives:

1. Identify key features that aid in accurate financial fraud detection.
2. Identify a classification algorithm to detect financial statements fraud.
3. Train a metalearning model to detect financial statements fraud using the features identified.
4. Test the model's accuracy in detecting financial fraud.

1.4 Problem Justification

It has emerged that the chances and risk of fraud increases more during periods of recession (Bănărescu, 2015), such as the ones currently being experienced due to the covid-19 pandemic. It therefore becomes important to organizations to implement a series of anti-fraud techniques.

1.5 Significance

The expected outcome of this research is that the proposed fraud detection method will be more accurate in detecting the presence of fraud in financial statements. This will facilitate stakeholders such as auditors and government agencies to detect fraud better.

The contributions to society will be that the funds that are lost to financial fraud can then be used more productively by organizations, and their growth is beneficial to communities in that there are more employment opportunities and better standards of living for the people.

1.6 Scope of the study

This research limits its review of previous financial fraud detection techniques to those that used data that was publicly available. This is due to the difficulty in obtaining companies financial data mostly where fraud is involved.

This research only focuses on income statement, balance sheets and statement of cash flow financial statements.

2. LITERATURE REVIEW

2.1 Introduction

Fraud definition according to the Association of Certified Fraud Examiners (ACFE) (I.SADGALI 2019) is act, deliberate or intentional, of denying one of property or money by unfair acts such as deception and cunning.

Financial fraud has far reaching affects going into the society, everyday life, and the financial industry. It is for this reason that a lot of investors, regulators and the public as well continue to pay key interest in fraud detection and prevention. This has led to good number of research projects being conducted over the years finding out ways of improving the fraud detection methods currently being used. Despite these efforts, financial fraud continues to grow, and this can largely be attributed to the fact that fraudsters are also getting smarter over time and end up bypassing the fraud detection methods set up so far.

Different stakeholders stand to benefit from improved fraud detection techniques, these being: auditors, investors and other partners, and government regulators (Abbasi 2012). Investors frequently possess little inside information on fraud occurring inside the organizations. Fraud detections tools could facilitate investors to making better informed decisions. Audit firms could benefit during the client routine audits. Government regulators could benefit from effective and accurate fraud detection tools since they would allow them to better prioritize and effectively target their investigatory efforts on the right places.

This research seeks to leverage on the fraud detection methods that have performed well in the past and seek to combine the best aspects of these methods, into a self-learning and ever improving fraud detection model that will keep off fraudsters for a long while.

2.2 Financial Statements

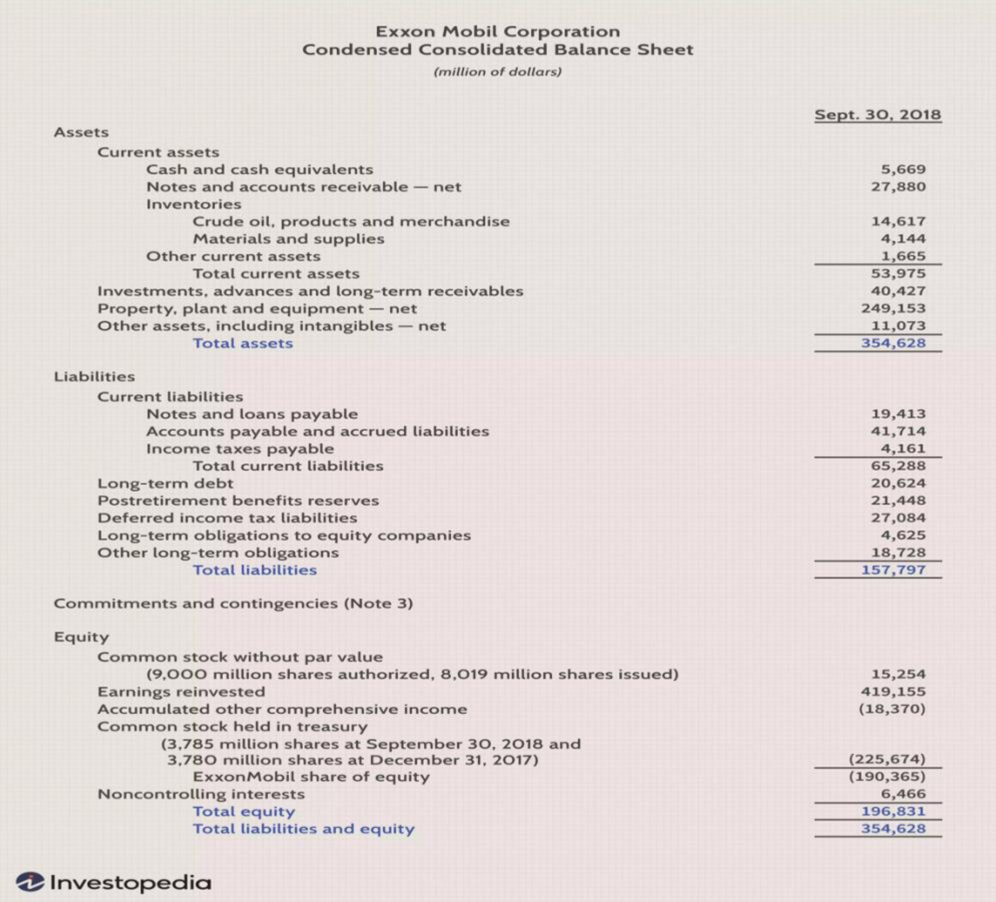
Financial statements are documents that show the financial status of a company. These are in the form of in periods of either a quarter year or a year (Maka, S and Sujata 2020). All companies report their financial statements to their governing body for tax filing purposes and to auditors to ensure accuracy. The companies' stakeholders also review the financial statements to understand the current state of the company, and this then helps in decision making to either turn things around if the company is in a crisis, or to continue the same trends if the records are showing the company is performing well.

Financial statements that will be looked at in this case are: income statements, balance sheets, and statements of cash flow, which are discussed further individually.

2.2.1 Balance Sheets

Balance sheets give a summary of assets, liabilities, and stockholder's equity at any specific time since they are dated (Murphy 2020). They show how assets are funded.

A balance sheet example is as below.



Exxon Mobil Corporation
Condensed Consolidated Balance Sheet
(million of dollars)

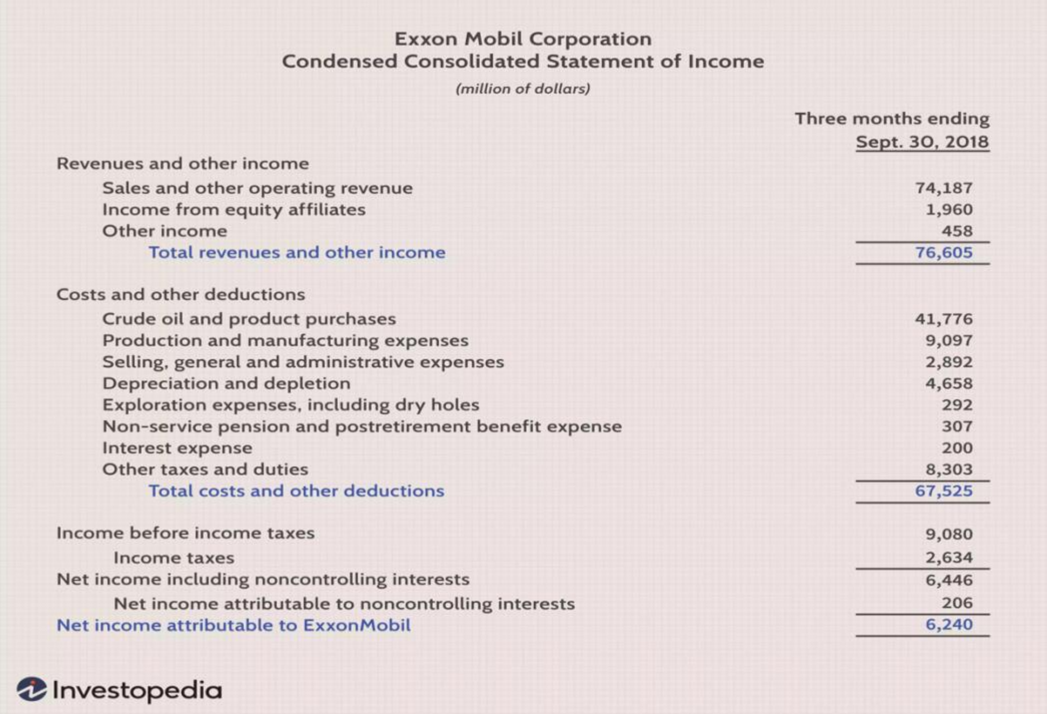
	Sept. 30, 2018
Assets	
Current assets	
Cash and cash equivalents	5,669
Notes and accounts receivable — net	27,880
Inventories	
Crude oil, products and merchandise	14,617
Materials and supplies	4,144
Other current assets	1,665
Total current assets	53,975
Investments, advances and long-term receivables	40,427
Property, plant and equipment — net	249,153
Other assets, including intangibles — net	11,073
Total assets	354,628
Liabilities	
Current liabilities	
Notes and loans payable	19,413
Accounts payable and accrued liabilities	41,714
Income taxes payable	4,161
Total current liabilities	65,288
Long-term debt	20,624
Postretirement benefits reserves	21,448
Deferred income tax liabilities	27,084
Long-term obligations to equity companies	4,625
Other long-term obligations	18,728
Total liabilities	157,797
Commitments and contingencies (Note 3)	
Equity	
Common stock without par value (9,000 million shares authorized, 8,019 million shares issued)	15,254
Earnings reinvested	419,155
Accumulated other comprehensive income	(18,370)
Common stock held in treasury (3,785 million shares at September 30, 2018 and 3,780 million shares at December 31, 2017)	(225,674)
ExxonMobil share of equity	(190,365)
Noncontrolling interests	6,466
Total equity	196,831
Total liabilities and equity	354,628

Investopedia

Figure 1 Balance Sheet

2.2.2 Income Statement

The income statement covers a period, such as a year or quarter a year. They provide an outline of expenses, revenues, earnings per share and net income (Murphy 2020). An example of an income statements is as below.



Exxon Mobil Corporation
Condensed Consolidated Statement of Income
(million of dollars)

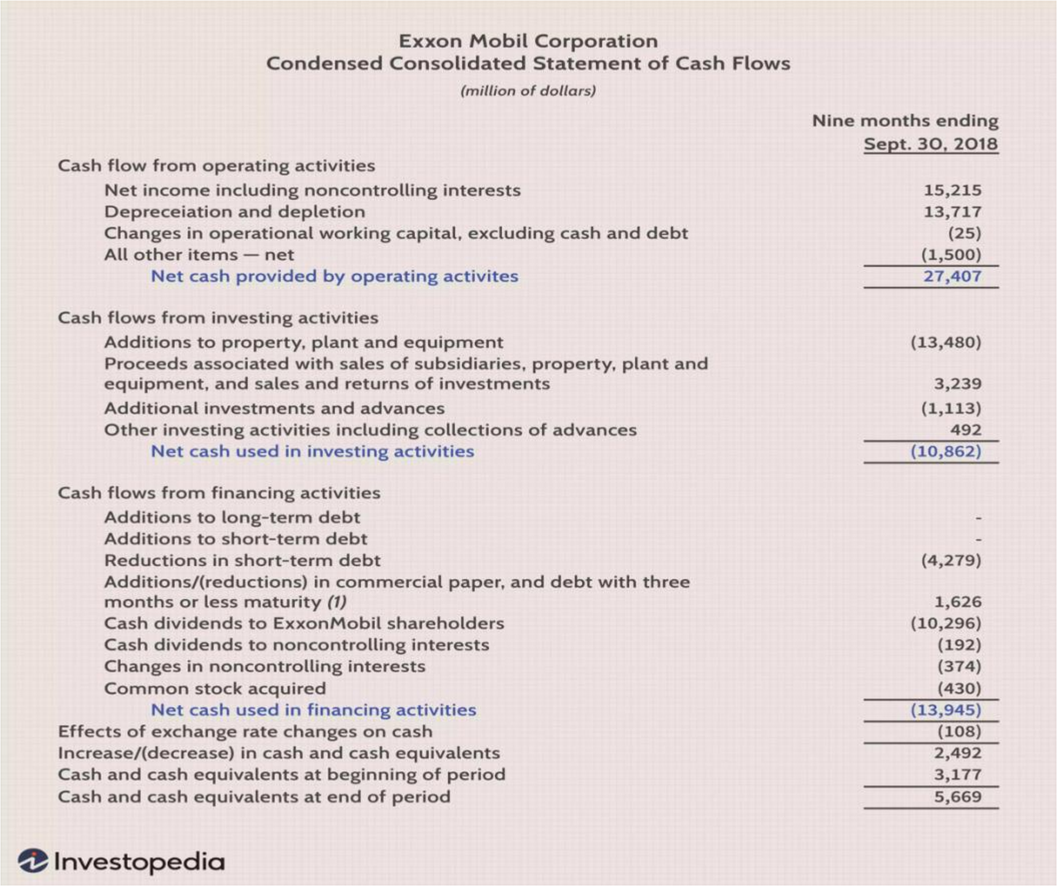
	Three months ending Sept. 30, 2018
Revenues and other income	
Sales and other operating revenue	74,187
Income from equity affiliates	1,960
Other income	458
Total revenues and other income	76,605
Costs and other deductions	
Crude oil and product purchases	41,776
Production and manufacturing expenses	9,097
Selling, general and administrative expenses	2,892
Depreciation and depletion	4,658
Exploration expenses, including dry holes	292
Non-service pension and postretirement benefit expense	307
Interest expense	200
Other taxes and duties	8,303
Total costs and other deductions	67,525
Income before income taxes	9,080
Income taxes	2,634
Net income including noncontrolling interests	6,446
Net income attributable to noncontrolling interests	206
Net income attributable to ExxonMobil	6,240

Investopedia

Figure 2 Income Statement

2.2.3 Cash Flow Statement

Cash flow statements (CFS) indicate how well companies fund their operating expenses, generate cash to pay their debt obligations, and fund investments (Murphy 2020). An example of a cash flow statement is as below.



Exxon Mobil Corporation
Condensed Consolidated Statement of Cash Flows
(million of dollars)

	Nine months ending Sept. 30, 2018
Cash flow from operating activities	
Net income including noncontrolling interests	15,215
Depreceiation and depletion	13,717
Changes in operational working capital, excluding cash and debt	(25)
All other items — net	(1,500)
Net cash provided by operating activites	27,407
Cash flows from investing activities	
Additions to property, plant and equipment	(13,480)
Proceeds associated with sales of subsidiaries, property, plant and equipment, and sales and returns of investments	3,239
Additional investments and advances	(1,113)
Other investing activities including collections of advances	492
Net cash used in investing activities	(10,862)
Cash flows from financing activities	
Additions to long-term debt	-
Additions to short-term debt	-
Reductions in short-term debt	(4,279)
Additions/(reductions) in commercial paper, and debt with three months or less maturity *(1)*	1,626
Cash dividends to ExxonMobil shareholders	(10,296)
Cash dividends to noncontrolling interests	(192)
Changes in noncontrolling interests	(374)
Common stock acquired	(430)
Net cash used in financing activities	(13,945)
Effects of exchange rate changes on cash	(108)
Increase/(decrease) in cash and cash equivalents	2,492
Cash and cash equivalents at beginning of period	3,177
Cash and cash equivalents at end of period	5,669

Investopedia

Figure 3 Cash Flow Statement

2.3 Current state of financial statements fraud detection

In the Big Data era, detecting fraud has proven to be a challenge and one approach that has been increasingly used to analyze relations and connectivity patterns is Graph-based anomaly detection (GBAD) (Pourhabibi, et al. 2020). Research was conducted which analyzed studies published from 2007 to 2018 and showed a growing trend of GBAD techniques for fraud detection (Pourhabibi, et al. 2020). The research showed that approximately 87.2% of the reviewed research have exclusively developed their models using unsupervised learning techniques. This was driven by the fact that data labels are often in short supply or even nonexistent in fraud detection.

Algorithms that are currently popular in detecting and predicting fraud in financial statements are genetic algorithms, support vector machines, naïve bayes, logistic regression, neural networks, and discriminant analysis (Maka, S and Sujata 2020). Some features of financial statements have also been used to detect fraud in these statements. Some of these features are:

- Gross profit of the company.
- Net profit of the company.
- Primary business income.
- Ratio of primary business income to total assets.
- Ratio of net profit to primary business income.
- Ratio of primary business income to fixed assets.
- Ratio of primary business profit to primary business profit of previous year.

These variables identified have a huge range of values, therefore need to be transformed, normalized, and standardized.

Most studies appreciate the fact that prior research on fraud detection has faced the challenges of accessing internal data (auditor-client relationships, personal and behavioral characteristics) since this data is not readily available to investors, auditors, and regulators (Ahmed Abbasi 2012). Privacy concerns have led to stakeholders and organizations being reluctant to share their fraud information. The direct impact of this is that there is a hinderance of research and this affects the integrity of the experiments conducted (Pourhabibi, et al. 2020).

2.4 Previous work done

Research was conducted on the performance of various machine learning techniques to detect fraud (I.SADGALI 2019). The different fraud detection technique types: descriptive and unsupervised, predictive, and artificial & computational intelligence were investigated in how effective they are towards fraud detection. The study also differentiated different types of fraud such as: credit card, insurance and financial statements and tested several techniques on each.

This study observed that majority of the deployed algorithms do not work in real time. For insurance fraud, the non-requirements of real time detection made it easier. Bayes logistic model was shown to have more superior ability to detect insurance fraud. Results on financial statements fraud detection showed that Probabilistic Neural Networks performed best with 98.09% accuracy, followed by Genetic Algorithm (95%).

This study concluded that hybrid detection methods were more preferred as they combined the individual strengths of each method. Suggestions for future work were to include the current algorithms, by bringing along a hybrid model that can handle both an imbalanced dataset as well as work real time.

Research has also been conducted on using evolutionary computation-based rule miners in order to detect financial statements fraud. The proposed method to do so was by improving firefly algorithm (FF) to a firefly and threshold acceptance hybrid (FFTA) miner and threshold acceptance (TA) rule miner. The modification was done by changing the position of the firefly.

The three algorithms (FF, FFTA and TA) were run on a financial statements' dataset with different number of features each. FFTA had the best sensitivity in all three cases (35, 18 and 10 features), with the max sensitivity being 79.52% from 18 features (Kaushik 2014). TA algorithm closely followed FFTA with 79.05% with 10 features. The algorithm with the best accuracy in detecting financial statements fraud was FFTA with a 73.14% accuracy on the 10 features dataset, followed by the same algorithm on the 18 features dataset.

From the three evolutionary algorithms, it was then concluded that Threshold acceptance (TA) and firefly and threshold acceptance hybrid (FFTA) gave decent results especially with 10 features.

A study was also conducted on a new accounting fraud prediction model (Yang, et al. 2020). The proposed prediction model differed from the current benchmarks of logistic regression and support vector machines (SVM) by using ensemble learning to predict fraud. This study also sought to use raw financial variables from financial statements. The basis for this was that they were the core of the accounting system.

The study used 28 raw financial variables to determine the performance of the prediction models. Tests were conducted on whether a model built from an ensemble algorithm and based on 14 financial ratios would perform better instead, or whether the combination of the 28 raw variables and the 14 ratios would perform best. The ensemble learning used was RUSBoost which has shown great performance and is computationally efficient (Yang, et al. 2020).

The evaluation metric used was the Area under Receiver Operating Characteristics (ROC) curve (AUC).

From the experiment, the top 5 features were (Yang, et al. 2020):

1. Common shares outstanding
2. Total current assets
3. Sale of common and preferred stock
4. Total property, plant and equipment
5. Account payable

Results from the study also showed that the ensemble model created outperformed the two models that were used as benchmarks (logistic regression and SVM). RUSBoost had an AUC score of 0.801 while Logistic regression came second with 0,708, while SVM being last with 0.661.

A project that uses meta-learning to detect financial fraud is Metafraud (Abbasi 2012). The project appreciates that recent developments in business intelligence has elevated the capability of discovering patterns associated with domains such as fraud. It was found useful to use meta-learning because of its ability to learn by itself, hence increasing the quality of results obtained. This is important because of the complexities associated with fraud. The framework uses a set of 12 financial ratios, influenced by prior research:

The experimental results of the research (Abbasi 2012) showed that a fraud recall of over 80% for different settings. These results indicated greater performance over what was currently being used. Thus, the feasibility of using metalearning methods to detect fraud was proved. Metafraud also contributed to the field of fraud detection with the generation of confidence scores.

The research points out to possible enhancements that could be made from Metafraud, such as having fraud detection systems that also use analytical business intelligence tools.

2.5 Gap

An aspect that emerges as important for fraud detection is real time detection and prevention (I.SADGALI 2019). This has led to suggestions that future work include hybrid algorithms that work in real time.

Previous work done with ensemble methods has shown that they perform better at detecting fraud (Yang, et al. 2020) and that there is need to research on using other ensemble methods as well.

It is apparent that fraud detection is faced by the challenge of not having enough data to improve with (Pourhabibi, et al. 2020). This leads to the need for a fraud detection technique that can work and self-improve with little data.

2.6 Ensemble Methods of Fraud Detection

These models combine multiple machine learning models to create a near perfect classification and predictive model that performs well in terms of both being accurate in its tasks and with fewer computation needed. Ensemble models analyze the same data and then decides on the next steps based on a majority.

Some of these methods are:

Random Forests

This method combines a forest from individual decision trees which when combined form an ensemble (Stojanovi´c, et al. 2021). Each tree

carries out a prediction by carrying taking majority votes and the class with the most votes wins. This enables individual unrelated models to combine and give the best prediction of data. Minimum correlation is ensured because the trees are built on randomly sampled data with replacement.

A problem that faces random forests is overfitting (Altexsoft 2021). This is when a model over-recalls the patterns in data and that makes it unable to make accurate predictions. Another issue is dataset imbalance.

Adaptive Boosting (AdaBoost)

Adaptive boosting works by building a strong classifiers from the combination of other weaker classifiers. It commonly uses decision trees as the individual classifier (Stojanovi´c, et al. 2021). The algorithm works by fitting the training data on the classifiers then the one with the least weighted error gets selected and the weights of the other data points get updated as well. The model ensures that after each iteration the classification errors of the classifiers are minimized. It would be affected by a dataset with outliers and noise.

Extreme Gradient Boosting (XGBoost)

This approach is an ensemble because it also combines decision trees. It uses gradient descent to boost a set of weaker decision trees (Stojanovi´c, et al. 2021). The algorithm also offers algorithm enhancements and optimizations.

A challenge faced with this model is that it is sensitive to outliers (Corporate Finance Institute 2021). This is because every classifier is obliged to fix the errors in the predecessors. Their implementation is also relatively slow since model training must follow a sequence. This also makes scalability almost impossible.

2.7 Metalearning

Metalearning introduces intelligent data mining process which is able to learn and adapt from experience it acquired previously (Razak and Ahmed 2015). This leads to minimal user interaction needed to perform informed data analyzation task.

Metalearning methods can be divided into three categories (Wenbo, et al. 2020):

1. Metric learning methods such as MatchingNets and RelationNets – these are more efficient with few shot training examples.
2. Memory network methods such as Meta Networks and TADAM – which keep experiences in memory.
3. Gradient descent based metalearning methods such as MAML, LEO, LGM-Net, CTM – which enable the model converging with few quantization steps by altering the optimization algorithm.

Metalearning algorithms that perform gradient descent at test time appeal more because of their simplicity and generalization properties (Nichol, Achiam and Schulman 2018). The effectiveness of finetuning gives more confidence of these approaches. One of these approaches is Reptile algorithm.

2.7.1 Reptile

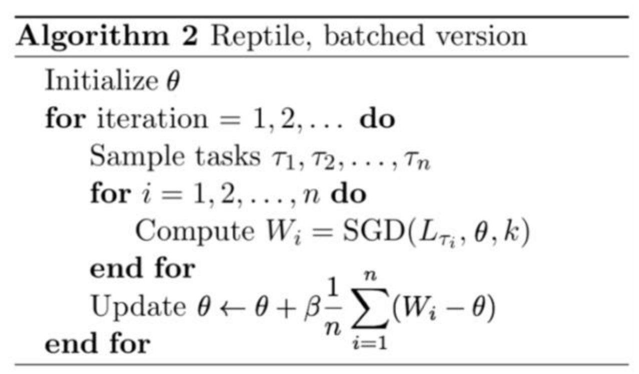
Reptile is a simple metalearning optimization algorithm that works similarly to Model-Agnostic Metalearning (MAML) in that both are model agnostic and use gradient descent (Weng 2018). The reptile algorithm loops through:

1. Sampling a task,

2. Training it on multiple gradient descent steps,
3. Then moving the model weights towards the new parameters.

Its algorithm carries out stochastic gradient descent on each task starting with initial parameter θ and returns the final parameter vector.

The batched version works by sampling multiple tasks instead of one within each iteration.



```
Algorithm 2 Reptile, batched version
  Initialize $\theta$
  for iteration $= 1, 2, \ldots$ do
      Sample tasks $\tau_1, \tau_2, \ldots, \tau_n$
      for $i = 1, 2, \ldots, n$ do
          Compute $W_i = \mathrm{SGD}(L_{\tau_i}, \theta, k)$
      end for
      Update $\theta \leftarrow \theta + \beta \frac{1}{n} \sum_{i=1}^{n} (W_i - \theta)$
  end for
```

Figure 4 Batched version of Reptile Algorithm

2.8 Link with existing work

The research relied heavily on ensemble algorithms of classification as it emerges that hybrid algorithms are more preferred as they combine the strengths of each method (I.SADGALI 2019) and they have also shown to perform better when it comes to detecting financials statements fraud (Yang, et al. 2020).

The research focused on overcoming the challenge of unavailability of financial datasets. This shall be done by taking a particular interest in a technique that works efficiently with little training data, which is metalearning (Abbasi 2012).

The use of raw features of financial statements has also shown increased accuracy in detecting fraud in financial statements (Yang, et al. 2020), therefore this project sought to compile all the raw features from financial statements and conduct experiments to determine their importance in predicting financial statements fraud.

2.9 Conceptual Framework

The conceptual model of the fraud detection model is illustrated in the figure below.

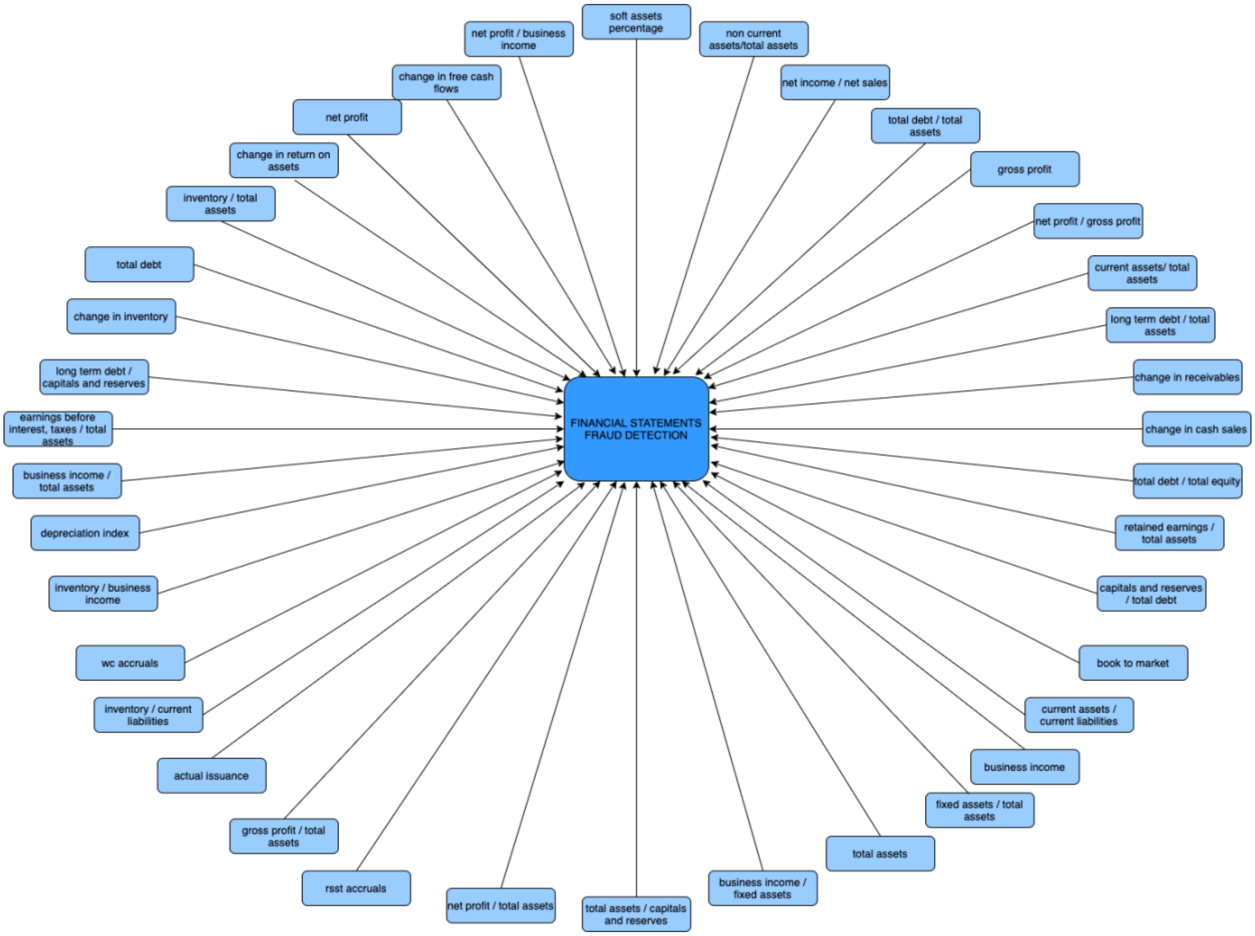

FINANCIAL STATEMENTS FRAUD DETECTION
soft assets percentage
non current assets/total assets
net income / net sales
total debt / total assets
gross profit
net profit / gross profit
current assets/ total assets
long term debt / total assets
change in receivables
change in cash sales
total debt / total equity
retained earnings / total assets
capitals and reserves / total debt
book to market
current assets / current liabilities
business income
fixed assets / total assets
total assets
business income / fixed assets
total assets / capitals and reserves
net profit / total assets
rsst accruals
gross profit / total assets
actual issuance
inventory / current liabilities
wc accruals
inventory / business income
depreciation index
business income / total assets
earnings before interest, taxes / total assets
long term debt / capitals and reserves
change in inventory
total debt
inventory / total assets
change in return on assets
net profit
change in free cash flows
net profit / business income

Figure 5 Conceptual Framework

3. RESEARCH METHODOLOGY

3.1 Introduction

The problem to be studied is the need for more accurate fraud detection methods because the current methods produce accuracies that are not ideal. As time progresses, fraudsters also become smarter hence the need for fraud detection methods that are not only recent, but which can also learn and improve over time.

The specific type of research will be applied research since the aim is to develop a technique and a product (McCombes 2019). The research shall use secondary data due to the inability to publicly access company's financial statements.

3.2 Research design

The research was divided into three major phases: (a) Identify key features that aid in accurate financial fraud detection, (b) Outline the accuracy of existing financial statements fraud detection techniques, (c) Train a metalearning model to detect financial statements fraud using the features identified and test the model's accuracy in detecting financial fraud.

3.2.1 Identify key features that aid in accurate financial fraud detection

This was an exploratory study on the key financial parameters that aid in accurate financial statements fraud detection. This was done through:

Data Collection

Four different sets of financial statements' features were identified as being key to detect the presence of fraud in the statements.

A literature review on detecting financial statement fraud identified 18 features that were used in detecting fraud in financial statements (Maka, S and Sujata 2020).

- Gross profit of the organization.
- Net profit of the organization.
- Primary business income.
- Proportion of primary business income to total assets.
- Proportion of inventory to primary business income.
- Proportion of inventory to total assets.
- Proportion of inventory to current liabilities.
- Proportion of net profit to primary business income.
- Proportion of primary business income to fixed assets.
- Proportion of primary business profit to primary business profit of previous year.
- Proportion of primary business income to previous year's primary business income.
- Proportion of fixed assets to total assets.
- Proportion of current assets to current liabilities.
- Proportion of capitals and reserves to total debt.

- Proportion of gross profit to total assets.
- Proportion of net profit to total assets.
- Proportion of long-term debt to total capital and reserves.
- Cash and Deposits.

An article on using machine learning to detect Account Fraud in US firms identified 14 financial ratios that could be used to detect financial statements fraud (Yang, et al. 2020).

- WC Accruals.
- RSST Accruals.
- Change in Receivables.
- Change in Inventory.
- Percentage of Soft Assets.
- Depreciation Index.
- Change in cash sales.
- Change in cash margin.
- Change in return on assets.
- Change in free cash flows.
- Retained earnings over total assets.
- Earnings before interest and taxes over total assets.
- Actual Issuance.
- Book to market.

A project to detect fraud in financial statements using evolutionary computation identified 35 features that were used to detect the presence of fraud (Kaushik 2014) which were almost similar to those already collected.

A metalearning framework for detecting financial fraud used 12 financial ratios in its detection (Abbasi 2012).

- Asset Quality Index (Non-current assets/Total assets).
- Asset Turnover (Net sales/Total sales).
- Inventory Growth (ratio of inventory to previous period’s).
- Leverage (ratio of total debt to total assets relative to previous period’s).

- Cash Flow Earnings Difference.
- Day sales in Receivables.
- Depreciation Index.
- Gross Margin Index.
- Operating Performance Margin.
- Receivables Growth.
- Sales Growth.
- SGE Expense.

Data Cleaning and Preparation

These parameters were all combined to come up with a full feature set. Some parameters were already present in financial statements such as ‘Total Assets’ and ‘Gross Profit’, while some parameters had to be obtained mathematically from the combination of their constituent features such as ‘Primary business income/Fixed assets’.

After generating all the required parameters and removing the duplicates, we were left with a feature set containing 39 financial variables.

The next step was to scale the data, which was done through sklearn’s Standard Scaler. Scaling is important since it standardizes values and ensures the machine learning algorithm does not give greater importance to greater or heavier values.

Seeing as the data was numerical, there was the need to convert the features into categorical features, which work best with Weight of Evidence and Information Value. This was done using bin creation. For each column, 100 bins of equal size were created from its smallest value to its largest value. Then each entry in a column would fall in either one of the 100 bins, hence changing the data to categorical.

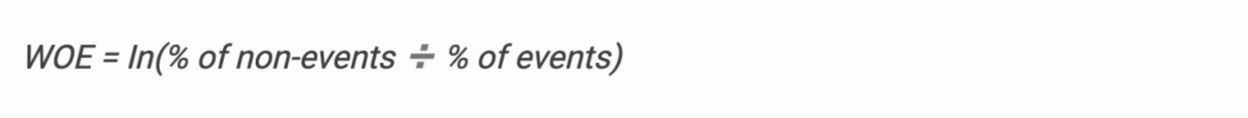
The dataset ranged from year 1991 to 2014. This project also split the dataset into 5 bands of 5 years each as follows: 1990 - 1994, 1995 – 1999, 2000 – 2004, 2005 – 2009, 2010 – 2014. Feature selection was done on each of those bands to understand how different features were key to financial statements fraud detection over different periods of years.

Validation

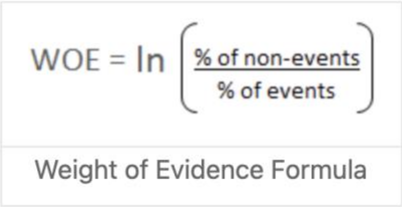
Validation of these features was done using Weight of Evidence (WOE) and Information Value (IV).

Weight of evidence algorithm is important since it gives a measure of how values support or undermine a hypothesis.

In this case the concept of WOE will be used in terms of events (fraud case) and non-events (non-fraud case). The algorithm is thus given by:



WOE = In(% of non-events ÷ % of events)



$$\text{WOE} = \ln\left(\frac{\text{\% of non-events}}{\text{\% of events}}\right)$$

Weight of Evidence Formula

Figure 6 Weight of Evidence Algorithm

ln – natural log

Information value is related to the weight of evidence algorithm as it is derived from it as follows:

$$IV = \sum(\%\text{of non-events} - \%\text{of events}) * WOE$$

The following table (Stojanovi´c, et al. 2021) then provides a standard rule of thumb for using the Information Value to understand the predictive power of each variable.

Table 1 Information Value

Information Value	**Variable Predictiveness**
< 0.02	Not useful for prediction
0.02 – 0.1	Weak predictive power
0.1 – 0.3	Medium predictive power
>0.3	Strong predictive power

The output of this phase will provide the features to be used by the metalearning system in training its model thus satisfying the first research objective.

3.2.2 Identify a metalearning algorithm to detect financial statements fraud

This phase of the research was an exploratory study, focusing on the performance of current financial fraud detection techniques.

Data Collection

This phase required different machine algorithms that could be used to classify fraud occurrence from large volumes of data. Literature review of what's currently been done in the financial statements fraud detection sector showed that hybrid methods were coming out as more accurate in detecting fraud. These are

- Random Forest
- Adaptive Boost (AdaBoost)
- Extreme Gradient Boost (XGBoost)
- First Order MAML
- Reptile

Running the Experiment

The experiment was conducted by creating models from the algorithms identified in the data collection step and testing their accuracy in classifying data. It was key to find data with high dimensionality as is the nature of financial statements, but low volume due to the unavailability of financial statements data.

A dataset was obtained from Kaggle which was for loan prediction using 20 various inputs such as: gender, marital status, dependents, education, self-employed, loan amount, credit history and so on. The target class was 'Loan_Status'.

This dataset was first preprocessed, by removing unnecessary fields such as Loan Id, then replacing the null values with means of that column. Since the dataset mainly contained numerical features, there was need to do categorical encoding in order to enable the algorithms work better with the data. This was done through panda's dummies creation function.

The dataset was then split to training and test datasets in the ratio of 7 to 3 respectively.

Testing and Validation

To validate our choice of algorithm, the models were trained using the training dataset to correctly classify an entry as either a successful loan application or not depending on the input variables.

The models were then tested in their accuracy to determine the loan classification of records in the test dataset. The metric used in evaluation was accuracy score.

3.2.3 Train a metalearning model to detect financial statements and test the model’s accuracy

The model on which to build the system will be determined by the outcome of the second phase, while the parameters to be used in training the model will be based on the output of the first phase. To implement this phase, applied experimentation technique (Edgar and Manz 2017) will be used.

Steps to be taken in this phase are:

Data Collection and Preparation

Data to train the model was obtained from research conducted on using machine learning to detect accounting fraud in US firms (Yang, et al. 2020). The accounting data is from fiscal year 1991 to 2014.

The final dataset contains 28 raw accounting variables, 14 financial ratios and the fraud labels.

The raw accounting variables are:

- Total Current Assets
- Trade Account Payable
- Total Assets
- Total Common/Ordinary Equity
- Cash and Short-Term Investments
- Cost of Goods Sold
- Common Shares Outstanding
- Total Long-Term Debt
- Depreciation and Amortization
- Income before Extraordinary Items
- Total Inventories
- Other Investments and Advances
- Total Short-Term Investments
- Total Current Liabilities
- Total Property, Plant and Equipment
- Total Preferred Stock (Capital)
- Retained Earnings
- Total Receivables
- Sales/Turnover (Net)
- Sale of Common and Preferred Stock
- Income Taxes Payable

- Total Debt in Current Liabilities
- Long Term Debt Issuance
- Price Close, Annual, Fiscal
- Total Liabilities
- Net Income (Loss)
- Total Income Taxes
- Total Interest and Related Expense

The financial ratios were already recorded in the data collection step for phase one.

The raw accounting variables were used as the features of the dataset, with the fraud labels being the target class.

The dataset was split into train, validate and test datasets in the ratio of 6 to 2 to 2 consecutively. Each dataset was then scaled using sklearn's Standard Scaler. After this, the datasets were split into their feature set (x) and target class (y) attributes.

To prepare them to be used by the model, the datasets were then transformed into a custom dataset, which enabled them to be transformed into Meta Datasets belonging to learn2learn MetaDataset's class. The final step was to then transform these meta datasets into Task Datasets which enabled the model in use to take samples from each dataset.

Prototype Design

The system design approach for developing the model was the CRISP-DM methodology. The approach incorporates six design phases that comprehensively cover the model development process. The figure below illustrates the development cycle of the methodology.

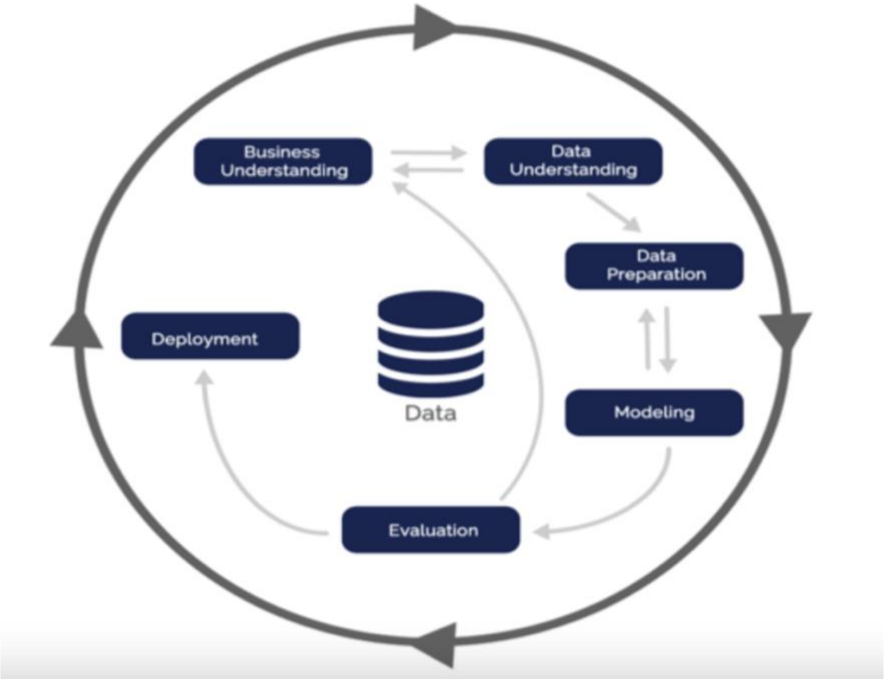


Figure 7 Prototype Development Cycle

The model design describes the organization of the Reptile model, and it also defines the number of input features. The input features were used to capture each independent variable.

The design of the model also involves determining the number of hidden layers, the number of neurons in the hidden layer, the number of output neurons and the activation function.

The prototype was formulated using the following parameters:

a) Number of input layers = 20

b) Number of hidden layers = 1

c) Number of neurons per hidden layer = 256

d) Number of outputs = 2

e) Activation function = Tanh

f) Meta learning rate = 1.0

g) Fast learning rate = 0.001

h) Train batch size = 10

i) Test batch size = 15

j) Number of iterations = 100000

The structure of the model can be visualized as below.

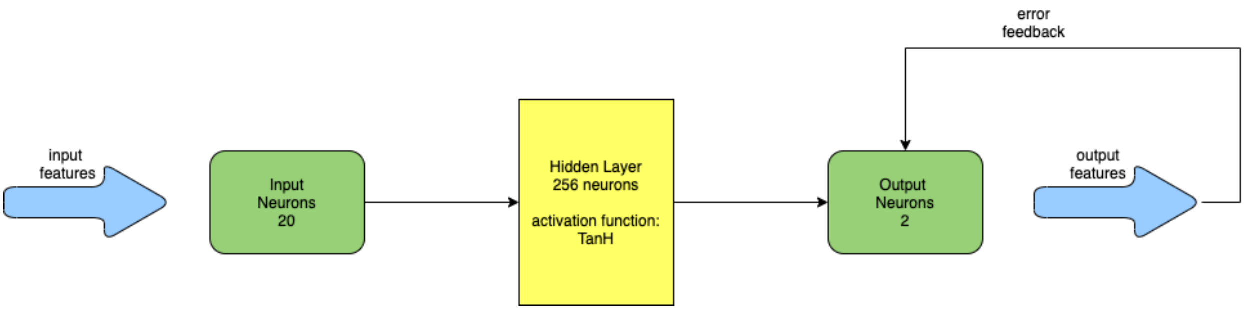


Figure 8 Model Structure

Prototype Development

The hardware resource required was aa personal computer that will conduct the machine learning model training and testing. For this, minimum specifications are 8 GB RAM, Intel Core i5 processor and 256 GB storage space. The software requirements are:

- A Unix based Operating System.
- Python programming language.
- Pycharm IDE
- Microsoft Office suite.
- Jupyter Notebook
- Torch library
- Learn2learn library

The model was developed using Torch's implementation of neural networks which enabled the project to build its model on. Torch is a machine learning library that provides a wide array of deep learning algorithms. Its nn module allows for the building of neural networks. The parameters used to design the neural network are outlined in the previous section on prototype design. The prototype also used torch's optim package which contains various optimization algorithms, the specific ones in use for this project being Adam algorithm and SGD (for stochastic gradient descent).

The model was trained using the train datasets, while validation was done on the validation datasets and testing carried out on the test datasets. For further validation of the first objective, the model was trained and tested on both the full dataset, which contains all the 39 features, and on the second dataset which contains the features obtained from feature selection.

The source code for the creation of the prototype as well as for creating our custom datasets, are in the appendix.

Testing and Validation

The evaluation of the prototype was in the form of its accuracy. Accuracy is the validation measure of precision.

Validation was conducted on the validation dataset after the model had finished ‘learning’.

Testing was then conducted on the test dataset, with both validation and testing being evaluated on the accuracy of the model.

4. RESULTS AND DISCUSSION

4.1 Introduction

This research had the main goal of coming up with a fraud detection technique that overcomes the challenge faced in fraud detection of unavailability of financial data due to its confidential nature. In chapter 3, data was collected on features contained in financial statements and their importance in detecting fraud outlined. Evaluation was also conducted on various ensemble techniques to classify multifaceted data, and a model was then built on the best performing algorithm.

This chapter outlines the results from conducting the experiments described in chapter 3. It also discusses what the results mean in relation to the problem being addressed.

4.2 Data Analysis

In this section, analysis was conducted on the dataset used to train and test the model.

4.2.1 Target class distribution

This analysis was conducted to find out the distribution of the target class in our dataset, which is this case is fraud.

The distribution is as below.

Table 2 Fraud Target Class Distribution

Fraud State	Count
no fraud	145081
fraud	964

This was also presented graphically in a bar chart.

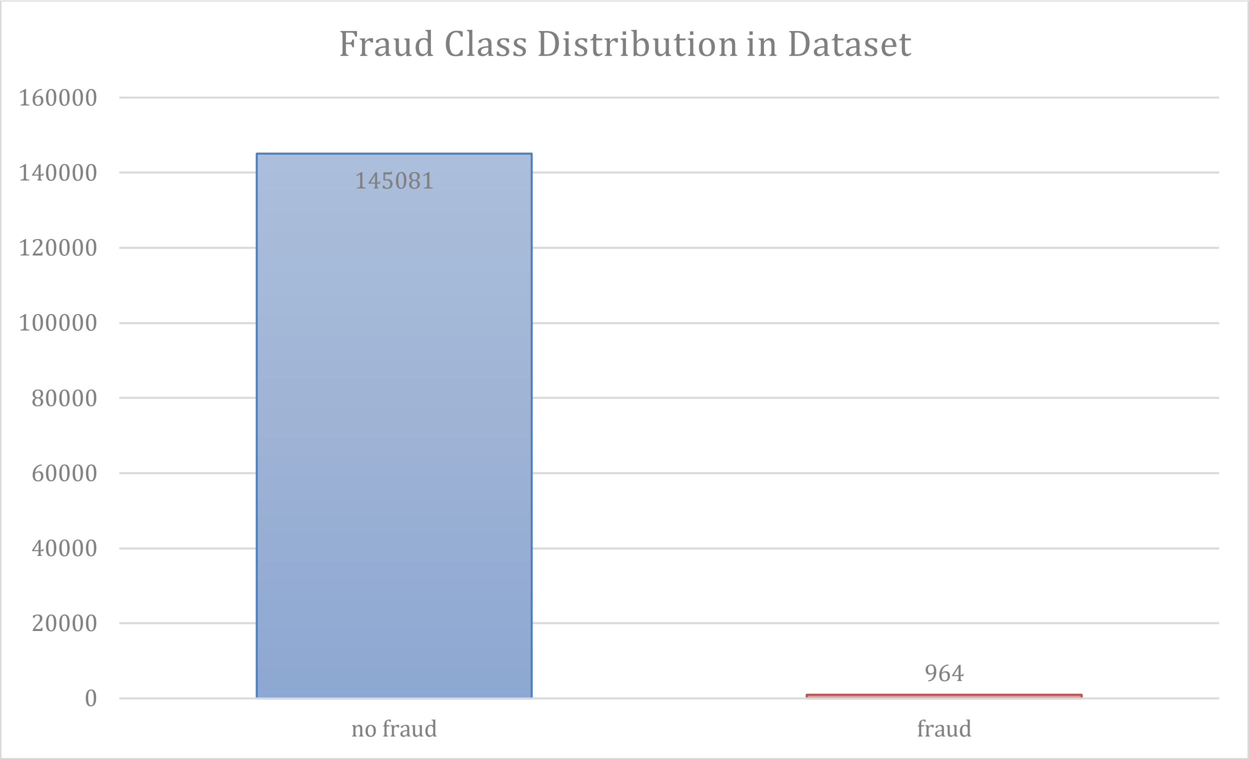


Figure 9 Fraud Target Class Distribution

This shows that there is a huge imbalance of the dataset when it comes to the target class as the fraud cases account to only 0.66% of the whole dataset with 964 fraud instances only. The non-fraud cases accounted for 145,081 instances.

4.2.2 Key Features Distribution

These results show the distribution of the raw features in the form of histograms of 100 bins.

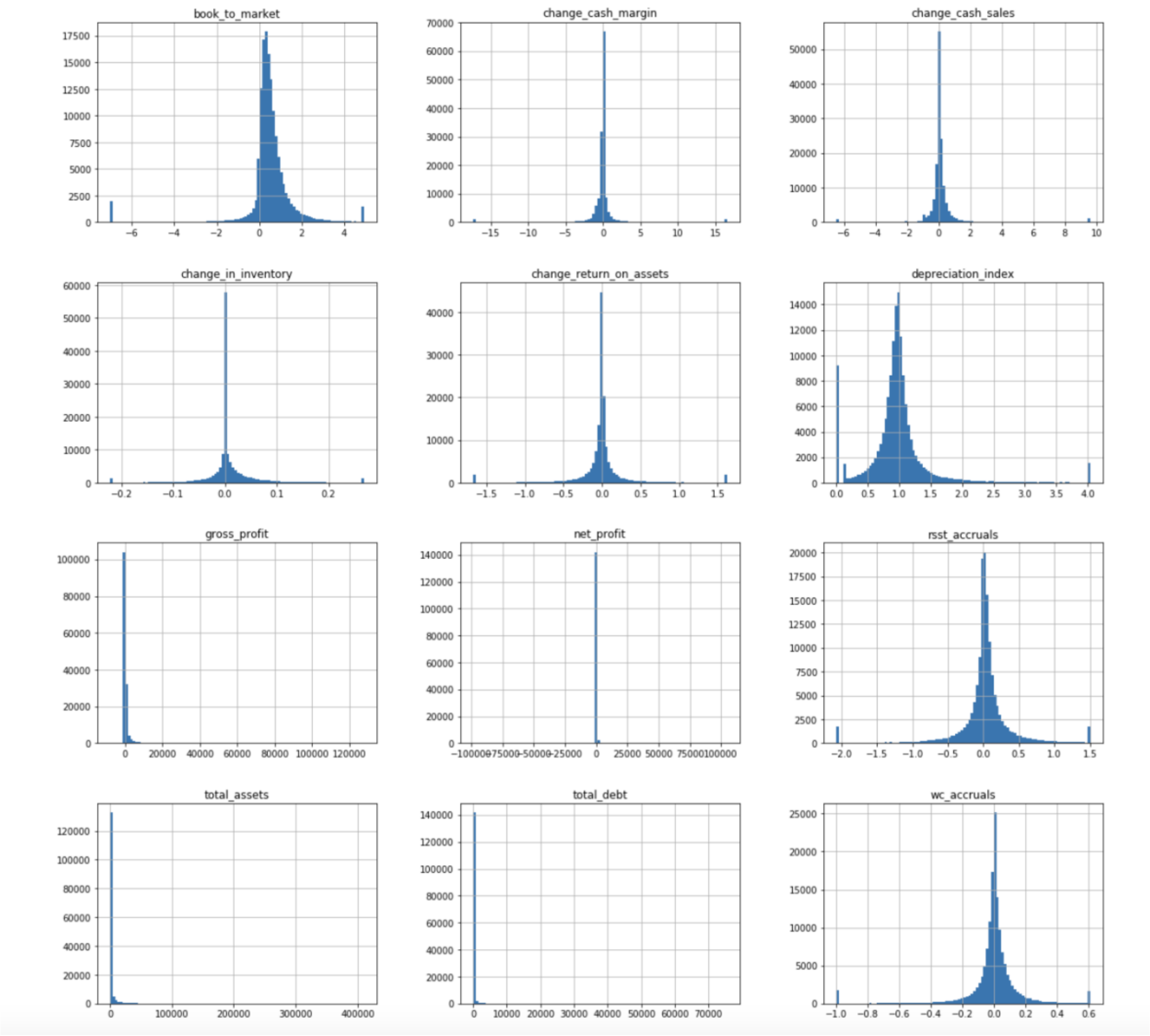


Figure 10 Raw Financial Features Distribution

The observations that can be made from these results is that for some features such as gross profit, net profit, total assets, and total debt are hugely undistributed as most of the records are all clustered on one place. Change in cash margins, change in cash sales, change in return on assets, depreciation index, rsst accruals and wc accruals had a normal distribution of records as shown above.

4.2.3 Key Features Correlation

Another analysis done on the data was to find the correlation between the different key features of financial statements. This was presented using a heatmap as below.

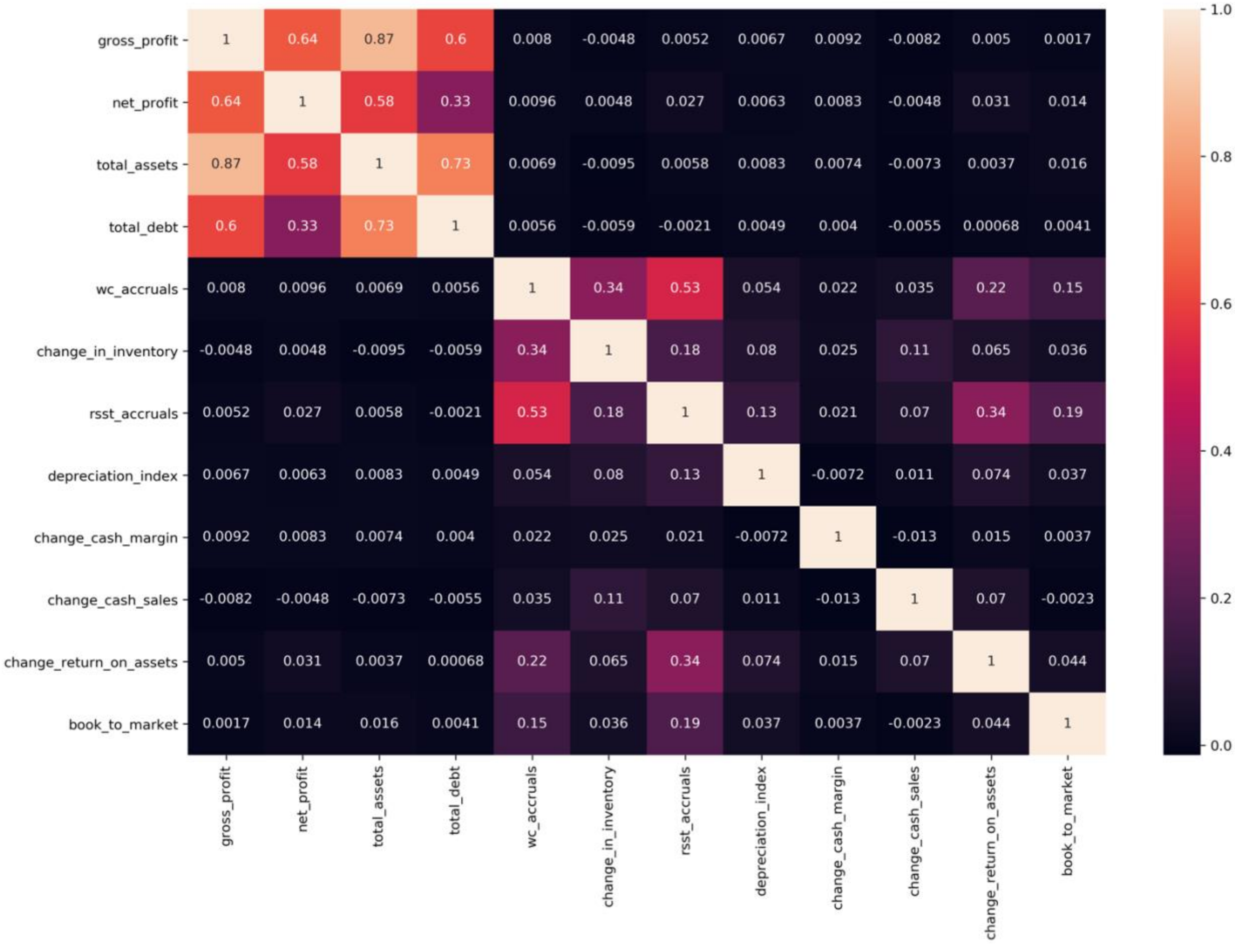


Figure 11 Key Features Correlation

From these results one can see that most of the features have very low correlation and that even fewer of them are negative. The largest correlation visible is between total assets and gross profit followed by total assets and total debt. Total assets appearing in the top two results show the need to pay key attention to it with further experiments.

4.3 Key Financial Statements Features

The main objective of this evaluation was to determine which features obtainable from financial statements are key to indicating the presence of fraud in such financial statements. Evaluation was done through information value which scores from 0.01 to 0.50. The results obtained are as shown below:

Table 3 Features' Information Value Scores

Rank	Feature	Score

1	soft_assets_percentage	0.44
2	non_current_assets/total_assets	0.25
3	gross_profit	0.24
4	current_assets/total_assets	0.23
5	change_in_receivables	0.19
6	change_cash_sales	0.18
7	retained_earnings/total_assets	0.16
8	book_to_market	0.16
9	business_income	0.14
10	total_assets	0.14
11	total_assets/capitals and reserves	0.14
12	rsst_accruals	0.14
13	actual_issuance	0.14
14	wc_accruals	0.12
15	depreciation_index	0.12
16	earnings_before_interest_taxes/total_assets	0.12
17	change_in_inventory	0.11
18	inventory/total_assets	0.10
19	change_return_on_assets	0.10
20	change_free_cash_flows	0.10

These results are shown graphically on the figure below.

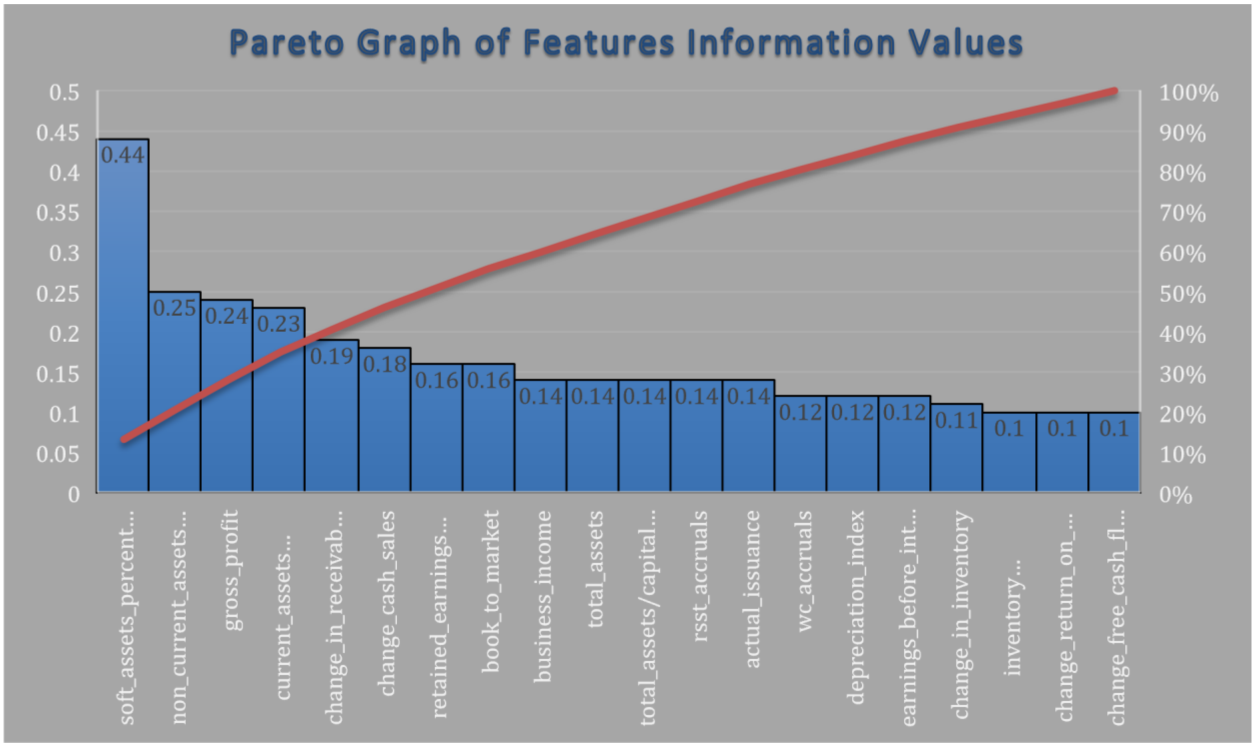


Figure 12 Features Information Value Scores

From the results, the following observations were made, first, the top 7 features make up to 50% of the importance when it comes to detecting the presence of fraud as shown by the pareto line. Out of those 5 features, 3 of them are related to the assets of the business these being: soft assets percentage, non-current assets/total assets, and current assets/total assets. Out of all the 20 features identified, ones dealing with assets were 7 in total.

Second, the most influential feature to detect the occurrence of fraud in financial statements is the percentage of soft assets. This was the only feature that was categorized as a strong predictor. These results show the significance of different features in financial statements when it comes to indicating that fraud has occurred.

Further experiment was done on the dataset being split into 4 bands of 5 years each and the importance of features through the different periods obtained as follows.

First Period (1990 – 1994)

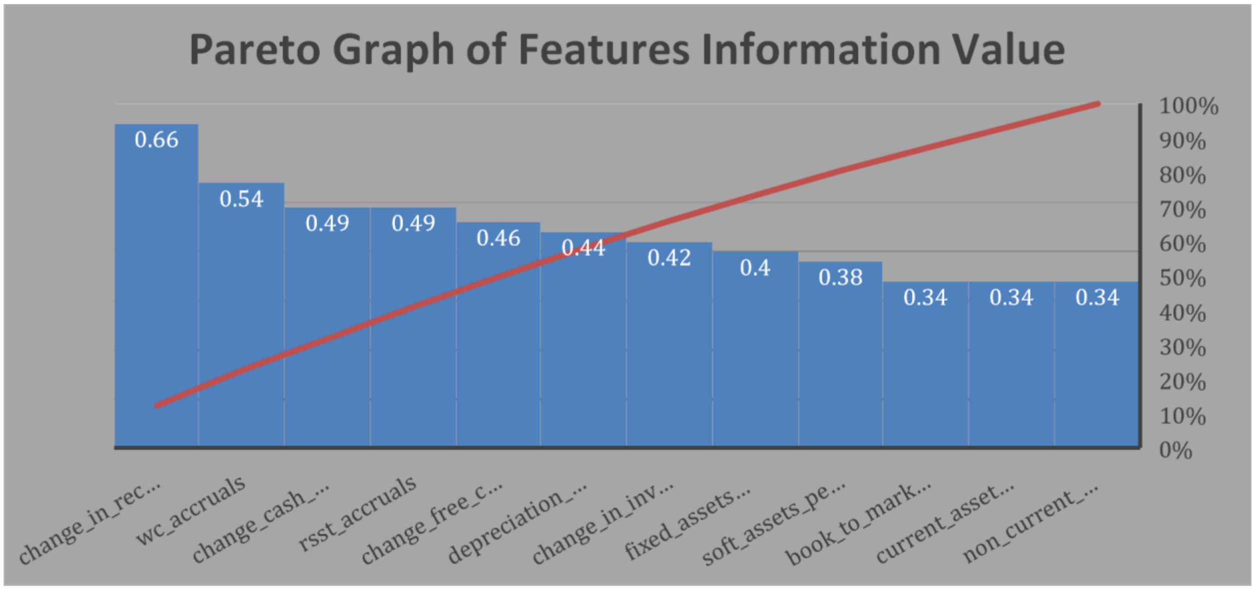


Figure 13 1990 - 1994 Feature's Information Values

From the results, the observations that can be made are that change in receivables and wc accruals were outliers and needed to be treated cautiously as they scored above 0.5. Several features were categorized as strong predictors such as change in cash sales, rsst accruals, change in free cash flows, depreciation index, change in inventory and fixed assets over total assets. These were the features most likely to indicate the presence of fraud during those years.

Second Period (1995 – 1999)

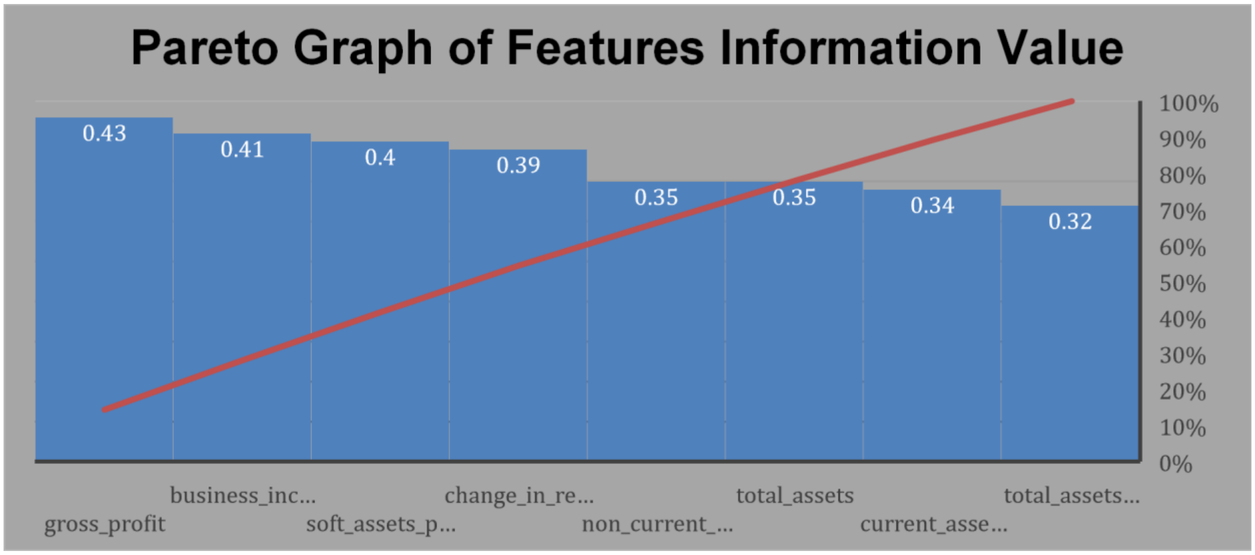


Figure 14 1995 - 1999 Features' Information Values

The results show that the strongest indicators of fraud during the years were gross profit, business income and soft assets percentage. Change in receivables, non-current assets over total assets and total assets were medium predictors during the period tested.

Third Period (2000 – 2004)

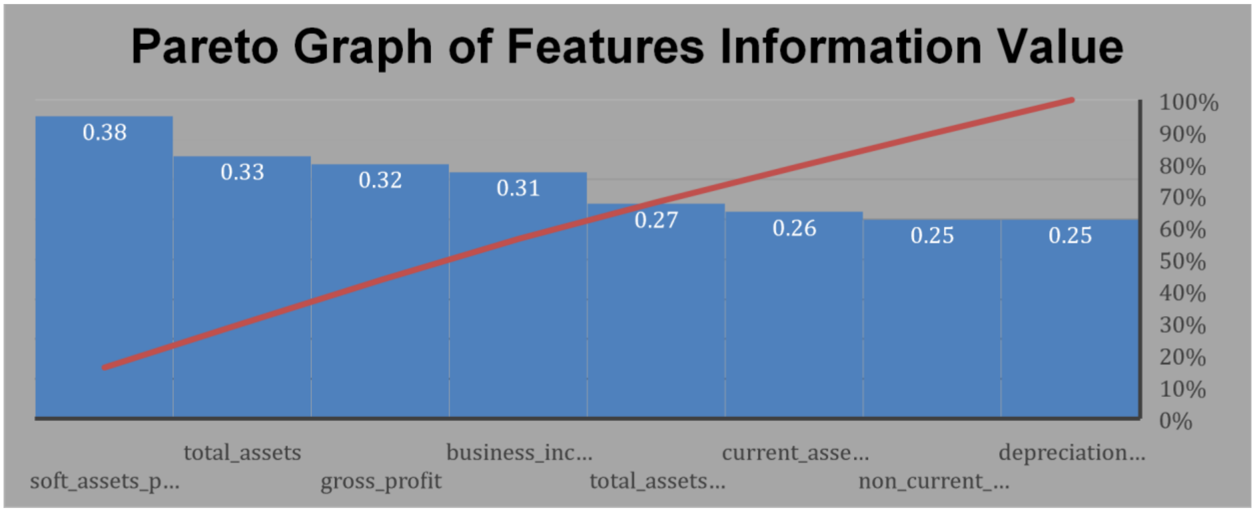


Figure 15 2000 - 2004 Features' Information Values

The observations made from these results are that soft assets percentage, total assets, gross profit, and business income were the best indicators of financial statements fraud during the period. This showed that assets were fairly popular for fraud tampering.

Fourth Period (2005 – 2009)

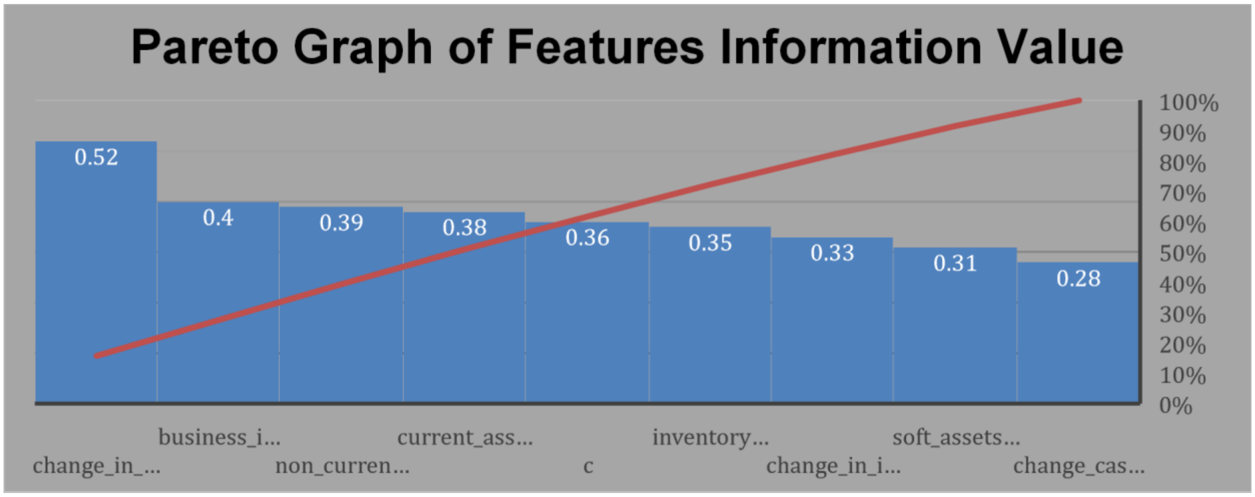


Figure 16 2005 - 2009 Features' Information Values

The results show that change in receivables was an outlier feature during the period as it scored more than 0.5. Business income was the only string predictor of fraud occurrence followed by medium predictors such as non-current assets over total assets, current assets over total assets, depreciation index, inventory over total assets and change in inventory. During this period, assets and inventory showed the highest likelihood of financial statements fraud.

Fifth Period (2010 – 2014)

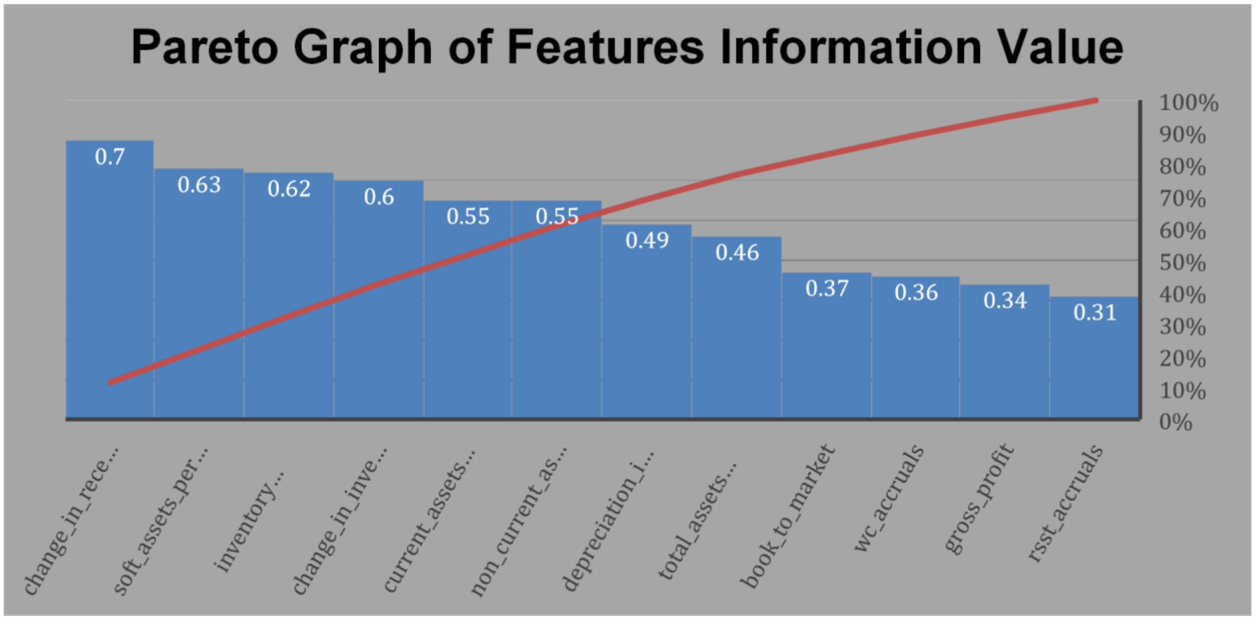


Figure 17 2010 - 2014 Features' Information Values

From the results, the observations made are that first there were about 6 outlier features as their score was above 0.5 hence, they needed to be treated suspiciously. Depreciation index and total assets over capitals and reserves were both strong predictors of fraud in this period.

4.4 Algorithms Evaluation

The main aim of this evaluation was to determine which algorithm would be the best fit to use in detecting financial statements fraud, due to its accuracy in a classification task with many features. The results of this are shown below.

Table 4 Algorithms' Performance

Algorithm	**Accuracy Score (%)**
Random Forest	80.0
AdaBoost	81.08
XGBoost	82.7
Gradient Boosting	82.7
First Order MAML	85.38
Reptile	87.86

These results are shown graphically as below.

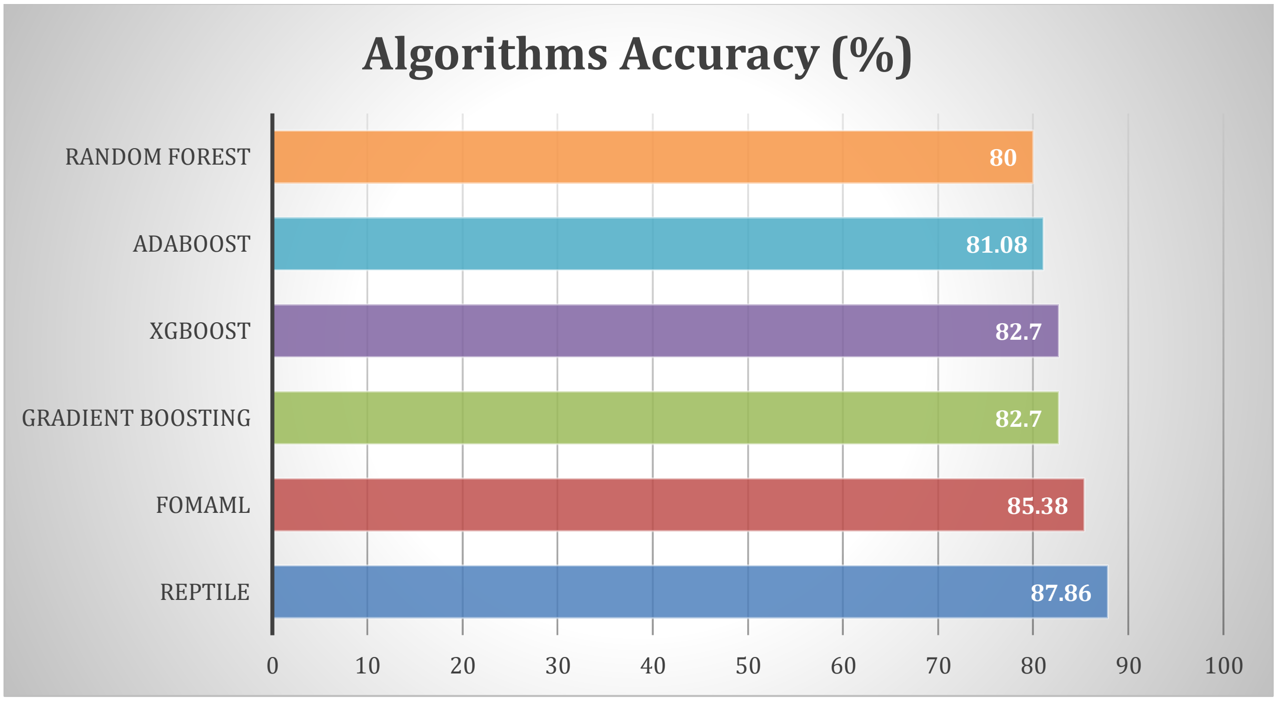


Figure 18 Algorithms' Evaluation Scores

From the results, the observations made are that ensemble methods do offer great accuracy when it comes to classification tasks that have a high dimensionality as all the methods scored 80% and above. It also emerges that metalearning algorithms (Reptile and First Order MAML) performed better than their bagging (Random Forest) and Boosting (AdaBoost, XGBoost and Gradient Boosting) ensemble methods.

Under metalearning algorithms, Reptile performed better of the two with 87.86% accuracy while First Order MAML had an accuracy of 85.86%.

This performance of the two algorithms shows that metalearning algorithms outperform other classification methods in regard to their accuracy and should therefore be considered for classification and detection tasks.

4.5 Model Evaluation

The key purpose of this objective was to train a metalearning model and test its accuracy in detecting fraud in financial statements. Two tests were conducted on the model's accuracy, one using the full dataset and another using dataset after feature selection has been done and only the strongest predictors left. The results of this are as shown below.

Table 5 Model Performance

Features	Accuracy Score (%)
Full Dataset, all the features	80.59
Optimized Dataset, after feature reduction	86.33

These results are further displayed graphically as below:

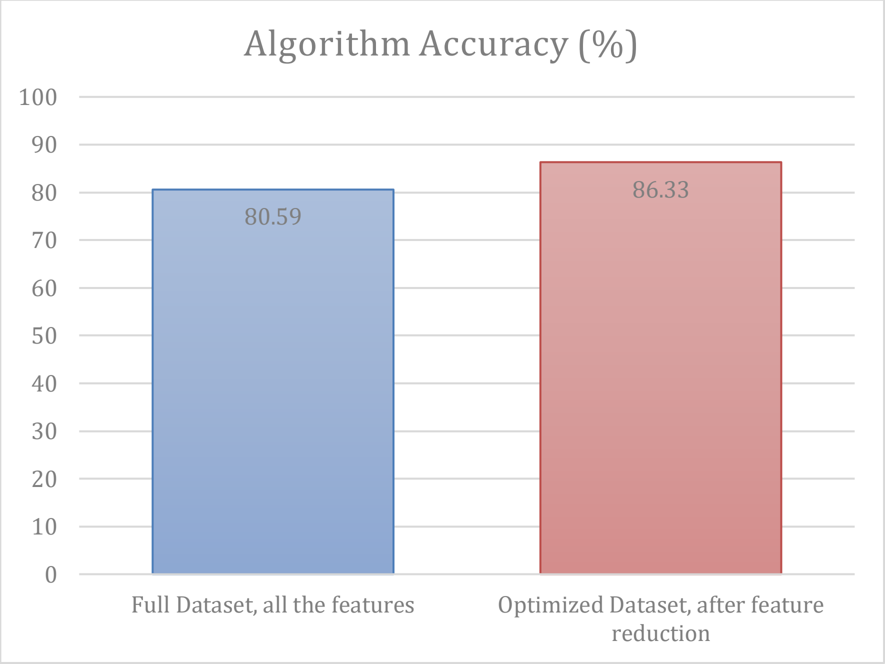


Figure 19 Reptile Model Performance

These results show the performance of the Reptile metalearning algorithm with two different datasets. The first dataset has all the features of the financial statements, and the algorithm had an accuracy of 80.59%. With the optimized dataset, which contained only the features identified as strong fraud predictors the algorithm had an accuracy of 86.33%.

This validates the need for a metalearning model to be used to detect fraud in financial statements and for only the right features to be fed into the model to increase its performance.

5. Conclusion and Recommendations

This chapter concludes the research by showing the achievements of this study, the contributions that this research poses, the challenges that were faced during the duration of the research and finally the recommendations for further work that can be done to build on what has already been done by this research.

5.1 Achievements

The main objective for this research was to find a financial statements fraud detection technique that overcomes the challenge faced in detection of such fraud which is inaccessibility of financial data to train with. This was achieved by meeting the specific objectives that this research had.

The first objective was to identify key features that aid in financial statements fraud detection. This objective was met by first collecting all the features of financial statements that have been used in fraud detection in recent research and then quantitatively determining their importance in detecting the presence of fraud in financial statements. Soft assets percentage was classified as a strong predictor while 19 other statements fell into the medium predictors category.
However, this was not so over all the years covered by the dataset, as the top predictors moved from change in cash sales to gross profit to soft assets percentage to change in receivables to depreciation index over the five periods covered. Assets related features were most dominant in good predictors of fraud over all the years covered. This supports work done by (Yang, et al. 2020) that also saw assets being in the top 5 of the top predictor features.

The second objective was to identify an algorithm to detect financial statements fraud. This was done by first identifying that ensemble methods performed well in classification tasks which led to identifying ensemble algorithms and metalearning algorithms as well. A classification experiment was then conducted on the algorithms identified and metalearning algorithms outperformed the other ensemble algorithms. Reptile algorithm emerged as the most accurate algorithm from those identified.

The third objective was to train a metalearning model to detect financial statements fraud. Achieving this objective relied on the results of the first and second objectives. The features identified in the first objective were used to train the model while the algorithm identified in the second objective was used to build the fraud detection model. The implementation of this objective resulted in a prototype that could be used to detect fraud in financial statements.

The fourth objective was to test the accuracy of the prototype developed because of achieving the third objective. The prototype was given a test dataset to evaluate its accuracy and the result was an 86.33% accuracy. It can therefore be concluded that a metalearning algorithm will outperform other algorithms in detecting financial statements fraud, and the most accurate algorithm for this is Reptile a gradient based metalearning algorithm as had also been pointed out by (Nichol, Achiam and Schulman 2018).

5.2 Contributions

This study contributes to the knowledge of how to accurately detect fraud occurrence in financial statements. It provides an insight as to which features of these statements are more key in indicating the possibility of fraud and even in what order these features are important. This backed up work by (Yang, et al. 2020) and (Kaushik 2014) that also saw improved accuracy in detection after feature selection. This knowledge can then be used by stakeholders in the industry as quick but accurate ways of checking for occurrence of fraud without even needing to use a machine learning algorithm to do so.

The research also shows how key ensemble methods are in fraud detection as also outlined by (Yang, et al. 2020). From these, metalearning algorithms proved to be even more accurate, this being backed up by the accuracy of the two metalearning algorithms as they emerged the top two from all the ensemble algorithms identified. This opens the way for further research to be done on metalearning algorithms in other disciplines as well.

5.3 Challenges

This main challenge faced in this research is the inaccessibility of financial statements data in the public domain. It is even harder to find these statements where fraud has occurred as efforts will have been put into place to conceal the presence of such fraud. This meant that there wasn’t ample data to be used to train our model. The one dataset that was obtained was also hugely imbalanced in regard to its fraud label. This meant that there was a huge emphasis on the model being used to be able to sufficiently learn from a few thousand records as opposed to ‘big data’ and that it also handles the dataset imbalance well.

The other challenge faced was on collecting the features used to identify fraud in financial statements. Detection of financial statements fraud in the real world is conducted by auditors and risk assessors, but the techniques they use is mostly on checking whether records balance. This meant that these features could not be collected from primary data, but from previous research done on financial statements fraud detection.

5.4 Recommendations and future work

The key recommendation for this project is that governments put in place systems to collect and make publicly available financial statements. This can be done in such a way that any confidential information is not exposed, and the advantage is that there will be enough data to train fraud detection models on.

Further research can be conducted on financial statements fraud detection using other metalearning algorithms to benchmark on not only the accuracy of prediction, but other factors such as speed of execution, ability to handle multidimensional data, and ease of implementing into a real time fraud detection system.

Future work on this research could include the functionality to train over data and save the learned model so that any prediction work needed does not need training of the model at the time. This will increase the speed of execution. Further improvements could include a graphical user interface, an explanation facility and even real time fraud detection. This could all be built onto the prototype created in this research to turn it into a system that can be used by the various stakeholders in the sector.

References

weng, lilian. 2018. *meta-learning: learning to learn fast.* november 30. https://lilianweng.github.io/lil-log/2018/11/30/meta-learning.html.

ahmed abbasi, conan albrecht, anthony vance, and james hansen. 2012. "metafraud: a meta-learning framework for detecting financial fraud." *mis quarterly* 1293-1327.

b, adrian. 2015. "detecting and preventing fraud with data analytics." *sciencedirect* 1827-1836.

jidong chen, ye tao, haoran wang, tao chen. 2015. "big data based fraud risk management at alibaba." *sciencedirect* 1-10.

i.sadgali, n.sael , f.benabbou. 2019. "perfomance of machine learning techniques in the detection of financial frauds." *science direct* 45-54.

abbasi, ahmed. 2012. "metafraud: a meta-learning framework for detecting financial fraud." *mis quarterly* 1293-1327.

oates, briony j. 2006. *researching information systems and computing.* london: sage.

tableau. 2020. *what is business intelligence? your guide to bi and why it matters.* https://www.tableau.com/learn/articles/business-intelligence.

kothari, c.r. 2004. *research methodology - methods and techniques.* new delhi: new age international limited.

vicino, franca. 2009. "the probabilistic neural network." *substance use and misuse* 335 - 352.

john. 2019. *icpak list of audit firms [contacts and locations included].* september 18. https://www.wikitionary254.com/icpak-list-of-audit-firms/?fbclid=iwar38ek0csoglp83rcmpxcadcwe_bfioormbmo99vnc5ulk7wwy9y2r09jpw.

bevans, rebecca. 2020. *statistical tests: which one should you use?* january 28. https://www.scribbr.com/statistics/statistical-tests/.

middleton, fiona. 2019. *the four types of validity.* september 6. https://www.scribbr.com/methodology/types-of-validity/.

—. 2019. *types of reliability and how to measure them.* august 8. https://www.scribbr.com/methodology/types-of-reliability/.

mccombes, shona. 2019. *the main types of research compared.* june 20. https://www.scribbr.com/methodology/types-of-research/.

edgar, thomas w, and david o manz. 2017. *research methods for cyber security.* elsevier inc.

pourhabibi, tahereh, kok-leong ong, booi h kam, and yee ling boo. 2020. "fraud detection: a systematic literature review of graph-based anomaly detection approaches." *elsevier.*

bevans, rebecca. 2020. *an introduction to the akaike information criterion.* march 26. https://www.scribbr.com/statistics/akaike-information-criterion/.

stojanovi´c, branka, josip boži´c, katharina hofer-schmitz, kai nahrgang, andreas weber, atta badii, maheshkumar sundaram, elliot jordan, and joel runevic. 2021. "follow the trail: machine learning for fraud detection in fintech applications." *mdpi* 21 (1594).

altexsoft. 2021. *fraud detection: how machine learning systems help reveal scams in fintech, healthcare, and ecommerce.* accessed april 05, 2021. https://www.altexsoft.com/whitepapers/fraud-detection-how-machine-learning-systems-help-reveal-scams-in-fintech-healthcare-and-ecommerce/.

corporate finance institute. 2021. *boosting.* accessed april 06, 2021. https://corporatefinanceinstitute.com/resources/knowledge/other/boosting/.

razak, abdul t, and najeeb g ahmed. 2015. "detecting credit card fraud using data mining techniques - meta-learning ." *indian journal of science and technology* 8 (28).

wenbo, zheng, yan lan, gou chao, and wang fei-yue. 2020. "federated meta-learning for fraudulent credit card detection." international joint conference on artificial intelligence.

nichol, alex, joshua achiam, and john schulman. 2018. "on first-order meta-learning algorithms." *arxiv* abs/1803.02999.

kaushik, nandan. 2014. *fraud detection in financial statements using evolutionary computation based rule miners.* patna: indian institute of technology.

maka, kiran, pazhanirajan s, and mallapur sujata. 2020. "literature review: detection of financial statement fraud." *palarch's journal of archaeology of eegypt/ egyptology* 17 (7).

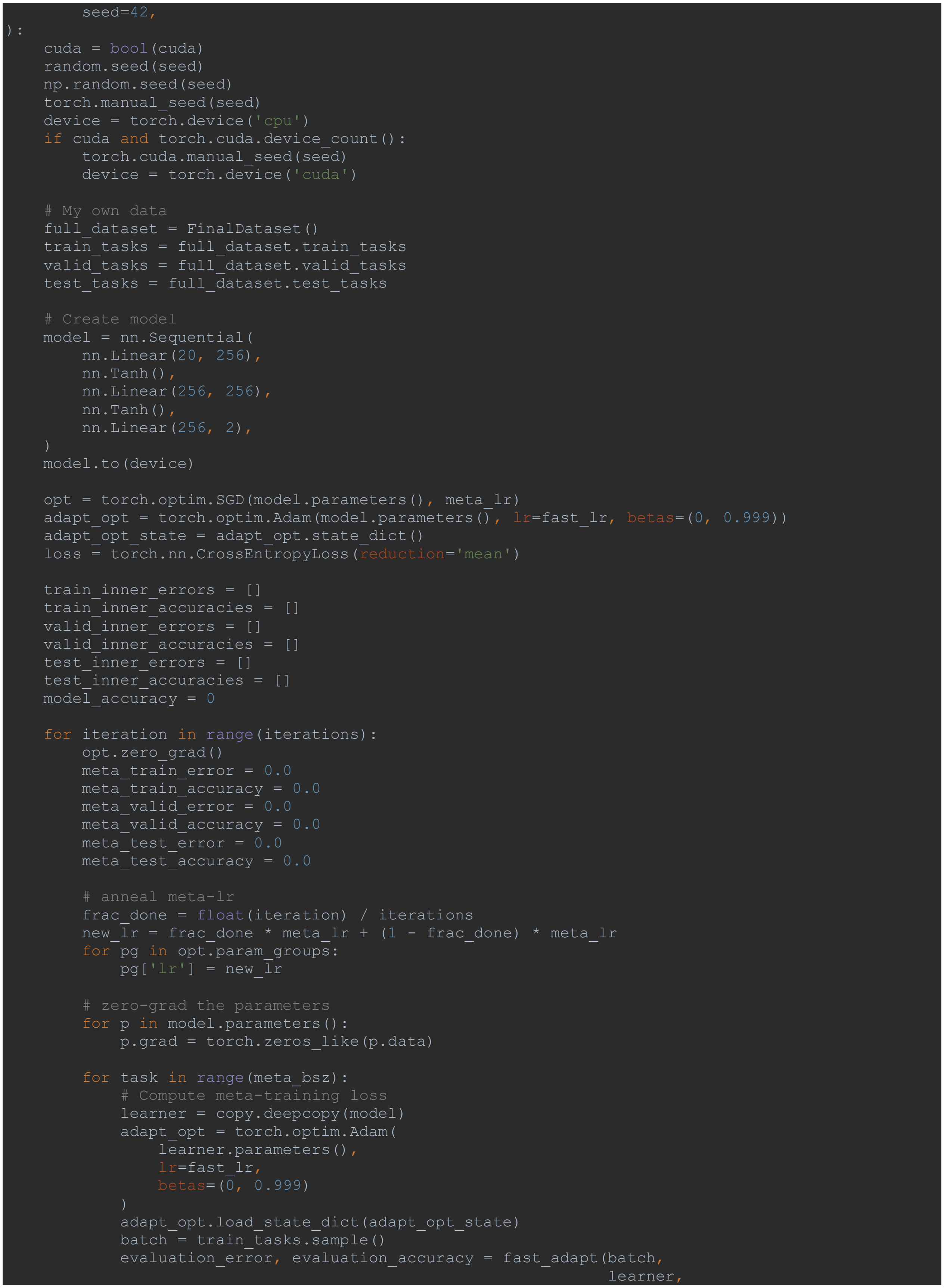
yang bao, bin ke, bin li, julia yu, and jie zhang. 2020. "detecting accounting fraud in publicly traded u.s. firms using a machine learning approach." *journal of accounting research* 58 (1): 199-235.

murphy, chris b. 2020. *financial statements.* accessed april 2021. https://www.investopedia.com/terms/f/financial-statements.asp.

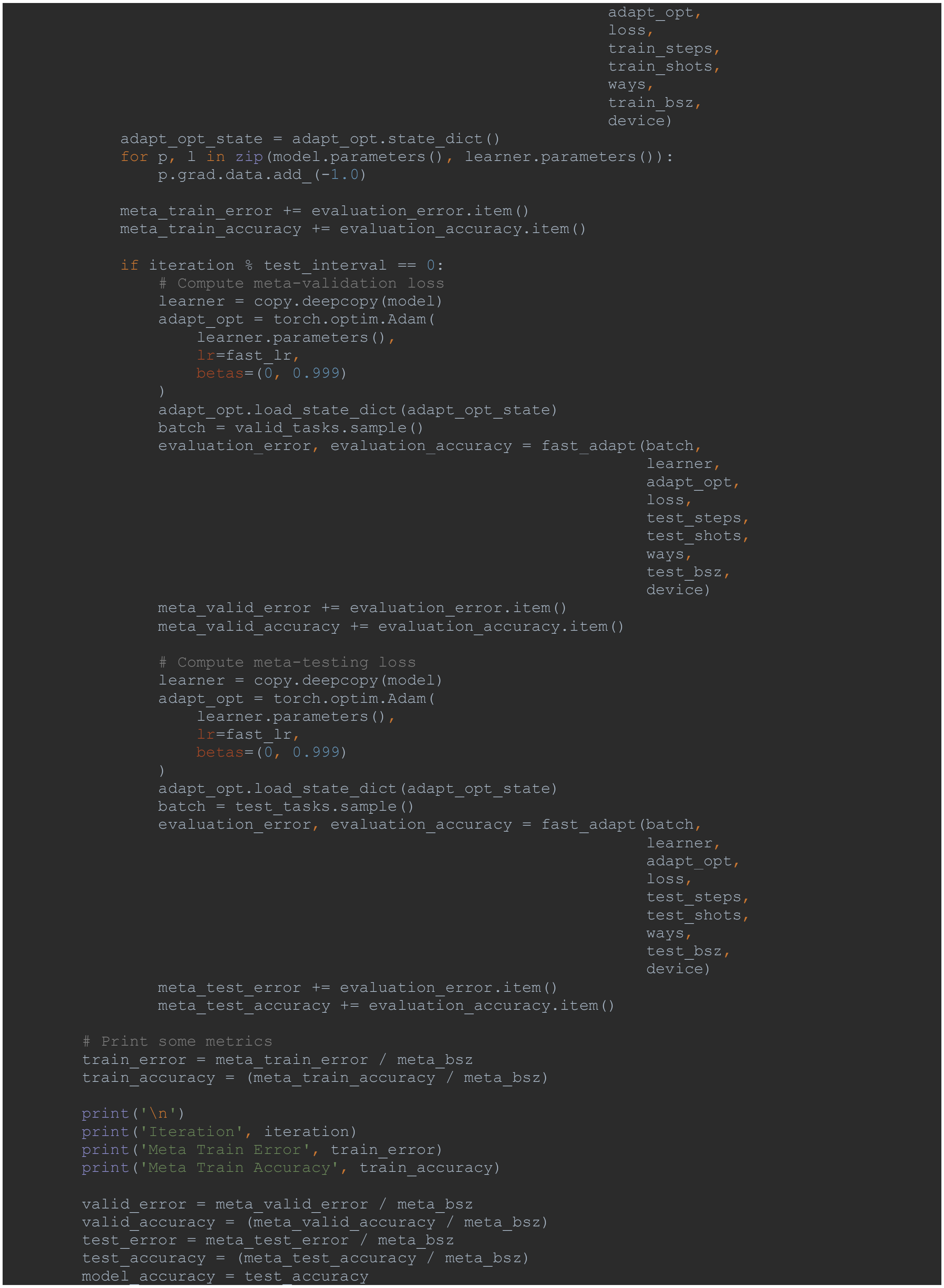
APPENDIX

APPENDIX 1: Source Code for Reptile Algorithm

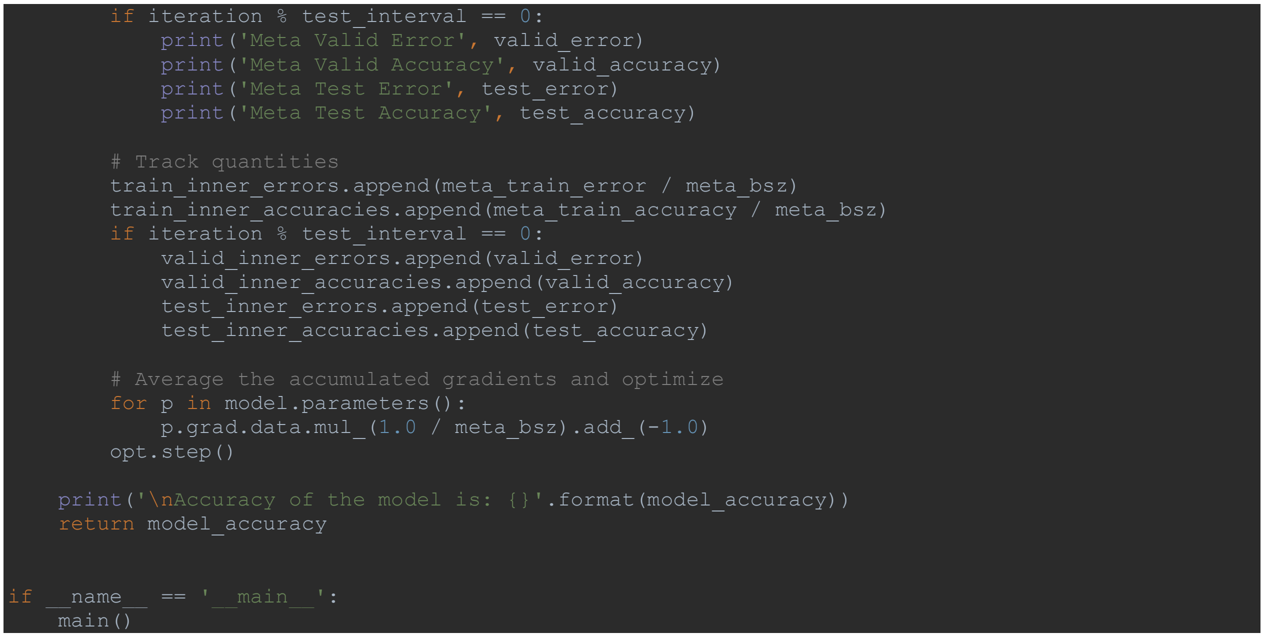
```
import copy
import random

import numpy as np
import torch
from torch import nn
from data.datasets import FinalDataset


def accuracy(predictions, targets):
    predictions = predictions.argmax(dim=1).view(targets.shape)
    return (predictions == targets).sum().float() / targets.size(0)


def fast_adapt(batch, learner, adapt_opt, loss, adaptation_steps, shots, ways, batch_size,
device):
    data, labels = batch
    data, labels = data.to(device), labels.to(device)

    # Separate data into adaptation/evaluation sets
    adaptation_indices = np.zeros(data.size(0), dtype=bool)
    adaptation_indices[np.arange(shots*ways) * 2] = True
    evaluation_indices = torch.from_numpy(~adaptation_indices)
    adaptation_indices = torch.from_numpy(adaptation_indices)
    adaptation_data, adaptation_labels = data[adaptation_indices], labels[adaptation_indices]
    evaluation_data, evaluation_labels = data[evaluation_indices], labels[evaluation_indices]

    # Adapt the model
    for step in range(adaptation_steps):
        idx = torch.randint(
            adaptation_data.size(0),
            size=(batch_size, )
        )
        adapt_X = adaptation_data[idx]
        adapt_y = adaptation_labels[idx]
        adapt_opt.zero_grad()
        error = loss(learner(adapt_X.float()), adapt_y)
        error.backward()
        adapt_opt.step()

    # Evaluate the adapted model
    predictions = learner(evaluation_data.float())
    valid_error = loss(predictions, evaluation_labels)
    valid_error /= len(evaluation_data)
    valid_accuracy = accuracy(predictions, evaluation_labels)
    return valid_error, valid_accuracy


def main(
        ways=2,
        train_shots=9,
        test_shots=5,
        meta_lr=1.0,
        meta_bsz=5,
        fast_lr=0.001,
        train_bsz=10,
        test_bsz=15,
        train_steps=8,
        test_steps=10,
        iterations=51,
        test_interval=10,
        cuda=1,
```



```
        seed=42,
):
    cuda = bool(cuda)
    random.seed(seed)
    np.random.seed(seed)
    torch.manual_seed(seed)
    device = torch.device('cpu')
    if cuda and torch.cuda.device_count():
        torch.cuda.manual_seed(seed)
        device = torch.device('cuda')

    # My own data
    full_dataset = FinalDataset()
    train_tasks = full_dataset.train_tasks
    valid_tasks = full_dataset.valid_tasks
    test_tasks = full_dataset.test_tasks

    # Create model
    model = nn.Sequential(
        nn.Linear(20, 256),
        nn.Tanh(),
        nn.Linear(256, 256),
        nn.Tanh(),
        nn.Linear(256, 2),
    )
    model.to(device)

    opt = torch.optim.SGD(model.parameters(), meta_lr)
    adapt_opt = torch.optim.Adam(model.parameters(), lr=fast_lr, betas=(0, 0.999))
    adapt_opt_state = adapt_opt.state_dict()
    loss = torch.nn.CrossEntropyLoss(reduction='mean')

    train_inner_errors = []
    train_inner_accuracies = []
    valid_inner_errors = []
    valid_inner_accuracies = []
    test_inner_errors = []
    test_inner_accuracies = []
    model_accuracy = 0

    for iteration in range(iterations):
        opt.zero_grad()
        meta_train_error = 0.0
        meta_train_accuracy = 0.0
        meta_valid_error = 0.0
        meta_valid_accuracy = 0.0
        meta_test_error = 0.0
        meta_test_accuracy = 0.0

        # anneal meta-lr
        frac_done = float(iteration) / iterations
        new_lr = frac_done * meta_lr + (1 - frac_done) * meta_lr
        for pg in opt.param_groups:
            pg['lr'] = new_lr

        # zero-grad the parameters
        for p in model.parameters():
            p.grad = torch.zeros_like(p.data)

        for task in range(meta_bsz):
            # Compute meta-training loss
            learner = copy.deepcopy(model)
            adapt_opt = torch.optim.Adam(
                learner.parameters(),
                lr=fast_lr,
                betas=(0, 0.999)
            )
            adapt_opt.load_state_dict(adapt_opt_state)
            batch = train_tasks.sample()
            evaluation_error, evaluation_accuracy = fast_adapt(batch,
                                                               learner,
```



```
                                                           adapt_opt,
                                                           loss,
                                                           train_steps,
                                                           train_shots,
                                                           ways,
                                                           train_bsz,
                                                           device)
        adapt_opt_state = adapt_opt.state_dict()
        for p, l in zip(model.parameters(), learner.parameters()):
            p.grad.data.add_(-1.0)

        meta_train_error += evaluation_error.item()
        meta_train_accuracy += evaluation_accuracy.item()

        if iteration % test_interval == 0:
            # Compute meta-validation loss
            learner = copy.deepcopy(model)
            adapt_opt = torch.optim.Adam(
                learner.parameters(),
                lr=fast_lr,
                betas=(0, 0.999)
            )
            adapt_opt.load_state_dict(adapt_opt_state)
            batch = valid_tasks.sample()
            evaluation_error, evaluation_accuracy = fast_adapt(batch,
                                                               learner,
                                                               adapt_opt,
                                                               loss,
                                                               test_steps,
                                                               test_shots,
                                                               ways,
                                                               test_bsz,
                                                               device)
            meta_valid_error += evaluation_error.item()
            meta_valid_accuracy += evaluation_accuracy.item()

            # Compute meta-testing loss
            learner = copy.deepcopy(model)
            adapt_opt = torch.optim.Adam(
                learner.parameters(),
                lr=fast_lr,
                betas=(0, 0.999)
            )
            adapt_opt.load_state_dict(adapt_opt_state)
            batch = test_tasks.sample()
            evaluation_error, evaluation_accuracy = fast_adapt(batch,
                                                               learner,
                                                               adapt_opt,
                                                               loss,
                                                               test_steps,
                                                               test_shots,
                                                               ways,
                                                               test_bsz,
                                                               device)
            meta_test_error += evaluation_error.item()
            meta_test_accuracy += evaluation_accuracy.item()

    # Print some metrics
    train_error = meta_train_error / meta_bsz
    train_accuracy = (meta_train_accuracy / meta_bsz)

    print('\n')
    print('Iteration', iteration)
    print('Meta Train Error', train_error)
    print('Meta Train Accuracy', train_accuracy)

    valid_error = meta_valid_error / meta_bsz
    valid_accuracy = (meta_valid_accuracy / meta_bsz)
    test_error = meta_test_error / meta_bsz
    test_accuracy = (meta_test_accuracy / meta_bsz)
    model_accuracy = test_accuracy
```



```
        if iteration % test_interval == 0:
            print('Meta Valid Error', valid_error)
            print('Meta Valid Accuracy', valid_accuracy)
            print('Meta Test Error', test_error)
            print('Meta Test Accuracy', test_accuracy)

        # Track quantities
        train_inner_errors.append(meta_train_error / meta_bsz)
        train_inner_accuracies.append(meta_train_accuracy / meta_bsz)
        if iteration % test_interval == 0:
            valid_inner_errors.append(valid_error)
            valid_inner_accuracies.append(valid_accuracy)
            test_inner_errors.append(test_error)
            test_inner_accuracies.append(test_accuracy)

        # Average the accumulated gradients and optimize
        for p in model.parameters():
            p.grad.data.mul_(1.0 / meta_bsz).add_(-1.0)
        opt.step()

    print('\nAccuracy of the model is: {}'.format(model_accuracy))
    return model_accuracy


if __name__ == '__main__':
    main()
```