ALGERIAN DEMOCRATIC AND POPULAR REPUBLIC
Ministry of Higher Education and Scientific Research
Mohamed Khider University – BISKRA
Faculty of Exact Sciences, Natural and Life Sciences
Computer Science Department

Order N°: SIOD32/M2/2021

Dissertation

Presented to obtain the academic master's degree in

Computer Science

Option: Information Systems, Optimization and Decision (SIOD)

Fraud detection using deep learning

Detecting energy consumption fraud using deep learning

By:

OUAMANE FERIAL

Defended on 06/28/2022 before the jury composed of:

Hattab Dalila	MAA	President
Meadi Mohamed Nadjib	MCB	Advisor
Chami Djazia	MAA	Examiner

Academic year 2021-2022

Acknowledgements

Praise be to Allah, Lord of the Worlds, and may prayers and peace be upon his Messenger. First of all, I thank Allah for his facilitation of this work and for giving me the strength, and knowledge to complete this work.

Then, I would like to express my gratitude to my lovely mother D.N for believing in me and supporting me throughout my academic career.

Dad! I have accomplished my goal and am soon to graduate. May Allah have mercy on you and provide you paradise as your home.

I wish to commend my supervisor, Professor ***Meadi Mohamed Nadjib*** *for giving me the fortitude I needed to take the next steps toward my ambition of graduating through his helpful guidance, and beneficial contributions that have left an indelible mark on the pages of this work; That means a lot to me.*

Dear soulmate Kenza, I would not have made it this far without you.

To my grandmother Aicha, you do not have any idea what I'm studying, yet your support never ceased. May Allah continue to bless you.

Dear brothers Ahmed and Ismail, and my uncle Ahmed Rechid, thank you so much for your continuous encouragement in anything I need.

Greetings to everyone from my two families, OUAMANE and DJOUDI. Finally, I would like to express my gratitude to everyone who assisted me, and pushed me ahead to achieve this work, especially Akram, Raid, Tahar, Youcef, Amina, Samira, Rania, Khadedja; I'm very thankful to you all.

OUAMANE FERIAL

ABSTRACT

Energy losses, which are commonly categorized into technical losses (TLs) and non-technical losses (NTLs), cost electric power companies billions of dollars each year. Non-technical losses are those that cannot be explained logically after the TL has been removed. This indicates that in order to ensure sustainable distribution of energy, it is necessary for creating a system to identify electricity thieves.

This theme suggests to establish a system detecting the number of customers who have committed fraud in consumption power using smart grid data provided by the State Grid Corporation of China (SGCC). We offer a fraud detector based on deep learning approaches that use one-dimensional Convolutional Neural Networks (1D-CNNs) and two-dimensional (2D-CNNs) Convolutional Neural Networks models, as well as an evaluation of their efficacy.

Key words: Energy, TLs, NTLs, Fraud, Deep leaning, Convolution neural networks, One-dimensional, Two-dimensional.

Résumé

Les pertes d'énergie, qui sont généralement classées en pertes techniques (TL) et en pertes non techniques (NTL), coûtent aux compagnies d'électricité des milliards de dollars chaque année. Les pertes non techniques sont celles qui ne peuvent pas être expliquées logiquement après la suppression du TL. Cela indique que pour assurer une distribution durable de l'énergie, il est nécessaire de créer un système d'identification des voleurs d'électricité.

Ce thème propose de mettre en place un système détectant le nombre de clients ayant commis une fraude à la consommation électrique en utilisant les données du réseau intelligent fournies par la State Grid Corporation of China (SGCC). Nous proposons un détecteur de fraude basé sur des approches d'apprentissage en profondeur qui utilisent des modèles de réseaux de neurones convolutifs unidimensionnels (1D-CNN) et bidimensionnels (2D-CNN), ainsi qu'une évaluation de leur efficacité.

Mots clés: Energie, TLs, NTLs, Fraude, Apprentissage en profondeur, Réseaux de neurones convolutifs, unidimensionnels, bidimensionnels.

ملخص

خسائر الطاقة، والتي تصنف عادة إلى خسائر فنية (TLs) وخسائر غير فنية(NTLs) ، تكلف شركات الطاقة الكهربائية مليارات الدولارات كل عام. الخسائر غير فنية هي تلك التي لا يمكن تفسيرها منطقيا بعد إزالة الخسائر الفنية. يشير هذا إلى أنه من أجل ضمان التوزيع المستدام للطاقة، من الضروري إنشاء نظام للتعرف على لصوص الكهرباء.

يقترح هذا الموضوع إنشاء نظام للكشف عن عدد العملاء الذين ارتكبوا عمليات احتيال في استهلاك الطاقة باستخدام بيانات الشبكة الذكية المقدمة من State Grid Corporation of China (SGCC) .
نحن نقدم كاشف للاحتيال يعتمد على مناهج التعلم العميق التي تستخدم شبكات عصبية تلافيفيه أحادية البعد (1D-CNNS) ونماذج شبكات عصبية تلافيفيه ثنائية الأبعاد(2D-CNNs) ، بالإضافة إلى تقييم فعاليتهم.

الكلمات المفتاحية: الطاقة، خسائر فنية، خسائر غير فنية، احتيال، التعلم العميق، شبكات عصبية تلافيفيه أحادية البعد، شبكات عصبية تلافيفيه ثنائية البعد.

Summary

List of Figures

List of Tables

General introduction

An energy sector is a group of stocks that deal with the production or distribution of energy. The energy sector companies involve the production of electricity that provides services by generating power, transmission, distribution, and marketing for industrial, commercial, public, and residential customers. It also comprises a slew of government agencies that oversee the industry. Electricity is considered the most prevalent source of power where everything in our world is powered by electricity, and it is extremely significant since it is used in everyday life for light, heat, and medical devices such as ventilators, respirators, and so on. The smart grid, which is a development of the traditional grid, provides the formerly chance to guide the energy sector into a new era of dependability, availability, and efficiency that will benefit our economic and environmental well-being. In a brief, the smart grid is digital technology that allows for two-way communication between the utility and its consumers, and the grid is made smart by sensing along transmission lines. The Smart Grid's two-way interactive capability will enable automated rerouting when equipment malfunctions or outages occur, which improves the electric power system's robustness and readiness to deal with crises such as severe storms, earthquakes, huge solar flares, and especially fraud.
Despite the advancement of the power system, there are still a wide variety of ways to steal electricity, whether through traditional or smart grid power. Researchers can use the data to assist electrical providers in detecting and prosecuting thieves. Through the growth of since and deep learning approaches, it is even possible to reduce fraud techniques. Deep learning is a new machine learning approach that is being used to identify fraud in a variety of ways. In this study, we choose to concentrate on deep learning-based smart grid fraud detection.

Problematic and motivation

Electricity is one of the most common forms of power that a person needs in his everyday life since it runs everything in our society. Furthermore, energy companies have complained about unexplained power losses, commonly known as Non-Technical Losses (NTLs). It occurs more frequently in traditional ones that do not have smart meters. The vast majority of non-technical energy loss occurs as a result of electricity theft, which results in significant financial losses.

Nowadays, machine learning algorithms are widely used and may be used to prevent power theft in a variety of ways, which helps energy companies for minimizing losses.

Aim of the work

The purpose of this research is to present a deep learning-based classification model for detecting fraudulent customers using smart grid data and to evaluate deep learning performance in this domain.
The dissertation also includes comparisons of the proposed models to past research.
In conclusion, the following key goals may be identified:

- Propose deep convolutional neural network models.
- Comparaison the obtained results.
- Comparisons with previous works.
- Conclusion on the use of deep learning in detecting abnormalities.

Dissertation structure

The following is a breakdown of the structure of the dissertation:

- **Chapter 1 Fraud detection:** This chapter discusses energy fraud and its methods, as well as the various fraud detection techniques.
- **Chapter 2 Convolutional Neural Networks:** This chapter provides an overview of machine learning and deep learning approaches, as well as a depth look into convolutional neural networks and related works.
- **Chapter 3 System design:** This chapter describes the datasets and outlines the system with all the phases.
- **Chapter 4 Implementation and results:** This chapter introduces the implementation tools, explains the code, and discusses the end result.

- **Conclusion and future work:** This chapter highlights the key results and offers suggestions for further research.

Chapter 1

Fraud detection

1.1 Introduction

Energy is one of the most important factors that accelerate the economic development of countries and technological production, as it helps communities by providing electricity, gas, and water to homes, companies, and factories. Energy is essential for accomplishing economic, social, and environmental goals for long-term human progress, as the development of various sectors of the economy is impossible without matching the development of the power sector.

One of the most serious issues facing Energy companies is the imbalance between the energy bill and energy supply, which is known as energy losses, with global utility companies losing more than $20 billion each year [1]. Energy losses are caused by a variety of reasons both inside and outside the power system, with Energy Fraud being the most common source.

As a result, a detection mechanism in the form of computer software is critical in avoiding fraud and limiting additional financial losses, where data is stored in computers and used immediately.

The aim of this chapter is to go through Energy fraud, its several varieties, and the ways to detect and prevent fraud and try to eliminate it. So, what does Energy Fraud? What are the many sorts of it? What are the methods for detecting it and protecting companies' utility from it?

1.2 Fraud

Fraud is still one of the most serious crimes, resulting in billions of dollars in damages each year. This section presents some fraud terminology and provides comprehensive coverage of the main types involved in fraud.

1.2.1 Definitions

In the Cambridge dictionary, “fraud is the crime of obtaining money or property by deceiving people” [2].
According to The American Heritage Dictionary, second college edition, fraud is defined as“ a deception practiced in order to induce another to give up possession of property or surrender a right” [3].
Fraud defines as an “intentional perversion of truth in order to induce another to part with something of value or to surrender a legal right” or as ”an act of deceiving or misrepresenting”. Webster’s New World Dictionary states that “fraud is a generic term, and embraces all the multifarious means which human ingenuity can devise, which are resorted to by one individual, to get an advantage over another by false representations” [4].
Van Vlasselaer provides a more comprehensive and detailed description of the phenomenon of fraud [5]: ”fraud is an uncommon, well-considered, time-evolving, carefully organized, and imperceptibly concealed crime that appears in many different types and forms”. This definition involves five characteristics that are associated with particular challenges related to the fraud detection system.
Fraud is a global incident affecting all countries and all sectors of the economy. Fraud includes a wide range of prohibited practices and unlawful acts including intentional deception. The International Professional Practices Framework defines fraud as: “.. any illegal act characterized by deceit, concealment, or violation of trust. These acts are not dependent upon the threat of violence or physical force. Frauds are perpetrated by parties and organizations to obtain money, property, or services; to avoid payment or loss of services, or to secure personal or business advantage” [6].
Fraud is the use of various unfair means including deception, false suggestions, or truth suppression by depriving another person of his or her property or money in return resulting in his or her loss, this involves fraudulent financial or other records to cover up the theft of money or other assets.

1.2.2 Type of Fraud

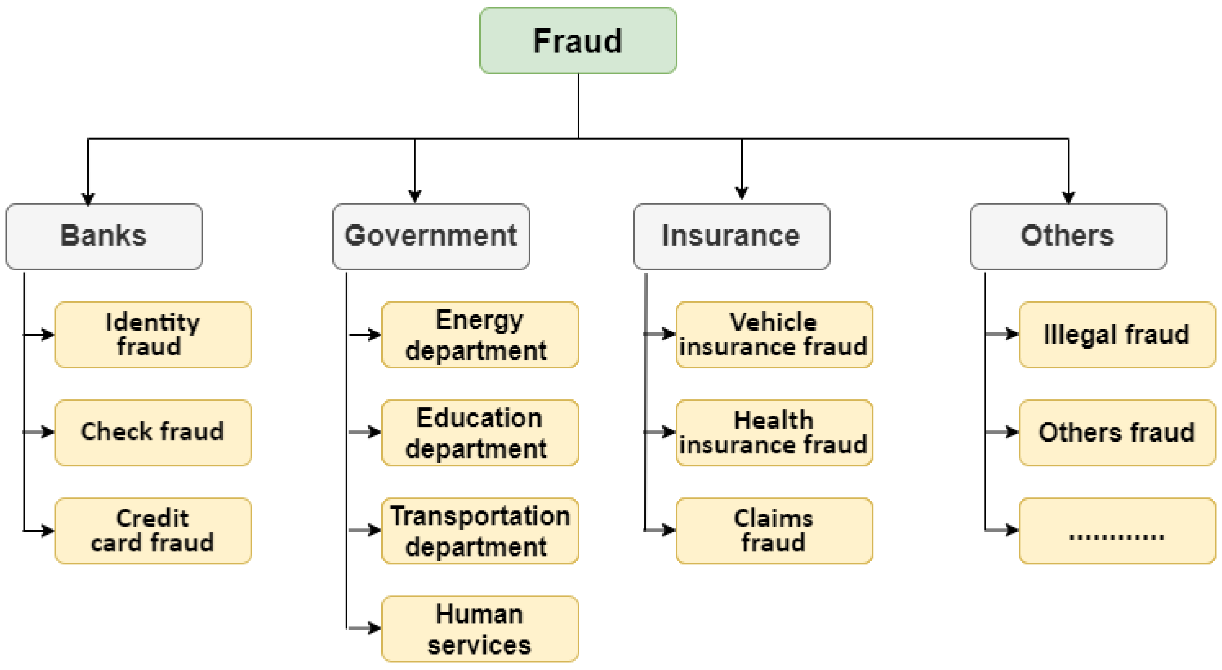


Figure 1.1: The most common sort of fraud.

As we can see in figure 1.1, fraud can be committed in different ways and in different settings such as :

- **Banking fraud:** It is the seizure of the customer's account. This happens when someone illegally gains access to the victim's bank account. Other examples of banking fraud include the use of identity fraud and check and credit card fraud [7].
- **Insurance fraud:** Insurance is a contract in which an individual receives financial protection or compensation for losses from an insurance company. Includes premium transfer fraud, vehicle insurance fraud, claims fraud, and health insurance fraud [8].
- **Government fraud:** Government fraud is committing fraud against federal agencies such as the U.S. Department of Health and Human Services, Department of Transportation, Department of Education, or Department of Energy. Types of government fraud include billing for unnecessary procedures, providing old equipment when billing for new equipment, and reporting hours worked for a worker that does not exist [9]

1.3 Fraud in Energy Consumption and Electricity Theft

Energy is one of the most important variables in accelerating a country's economic and technical progress. The infrastructure needed to ensure services includes kilometers of pipelines or lines that transmit energy to millions of meters that monitor the consumption of individual customers. An important problem faced by these companies is the imbalance in the energy bill with respect to the energy supplied, called energy losses [10].

Electricity theft is one of the causes of energy losses and it is defined as an installation made to bypass the meters that measure energy consumption, also it is an illegal use of electrical equipment for a purpose such as reducing the consumption amount or preventing billing. In common it is known as preventing the electricity meter to record the used amount by installing an invisible system to bypass the meter. Electricity theft causes significant harm to power grids, which influences power supply quality and reduces operating profits.

1.3.1 Energy Losses Types

Energy losses are usually classified into two types which are technical losses(TLs) and non-technical losses(NTLs) where :

TL is caused by the energy dissipation in the electricity transmission system and due to the joule effect on power lines [11] either by copper or iron transformer. Technical losses are legal and independent of the consumer, are approximately constant, are tolerable, and do not have a significant impact on the state economy. Calculation of TL is generally required for correct estimation of NTL but the calculation of TL is very complex, making it difficult to determine the point of loss and to estimate the amount of energy destroyed. TL cannot be completely stopped, but it can be reduced by applying some modulation techniques throughout the system [12, 13].

NTL is the residual loss that cannot be explained theoretically after removing the TL, which is abnormal power consumption behavior. NTL cannot be calculated accurately but can be estimated. Generally, NTL is caused by factors outside the power system, such as electricity theft, which is one of the main sources of NTL. In addition, electricity theft can seriously harm the economic performance of power providers and cause safety problems, such as equipment damage, power outages, and injuries [14].

1.3.2 Energy Consumption Fraud Methods

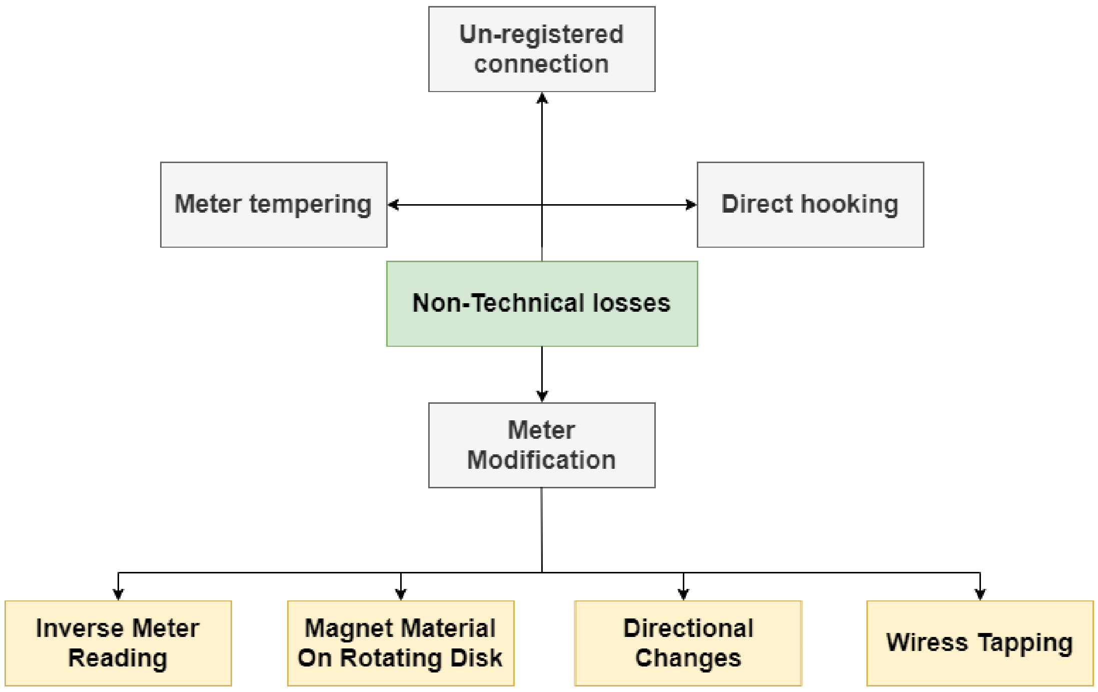


Figure 1.2: Energy Consumption Fraud Methods.

There are many ways to steal electricity, and the most common form of electricity theft is **meter tampering**, a type in which fraudsters tamper with the meter reading on an electromechanical meter device. In addition, there is a theft technique where consumers do not have the recording of their meter device to the respective electric power supply company; this method is called an **unregistered connection** [15].

Moreover, **direct hooking** from the mainline of a high transmission line (HTL) is a commonly used method, 80% of worldwide electricity stealing is by direct tapping from the HTL of which The consumers acquire the electricity without using the electricity meter panel [15, 16].

Furthermore, there are many ways to **modify the meter** such as pressing outer materials into the meter and penetrating holes in the electromechanical meter. In addition, putting a highly viscous liquid or resetting meter reading, destruct the rotating density of coil by meter screws, or using solid neodymium magnets for the disruption of the disk. All of these methods can be tempered by electromechanical meters [15, 16]. There are other methods involved in modifying the meter, some of which are mentioned below [15]:

A thieving technique called **inverse meter Reading** where intruders reverse the actual meter reading, and this is committed by opening the protective shield cover of the electricity meter.

Another one is named **magnet material** on rotating disk Magnet, in which the robber uses the material to stop the rotating disk of an electricity meter by putting a magnet on the upper surface of the rotating disk. This results in a lower energy consumption measurement compared to the normal condition.

Another common electricity theft technique is known as **directional Changes**, where the intruders changed the direction of the energy meter from its actual position. This results slowly down the speed of the rotating disk and by these latter measures a lower amount of energy.

1.4 Smart Grids

To avoid all of the above methods of stealing electricity and to decrease energy transmission and distribution losses, some nations have transformed their traditional grids into smart grids. As a result, we will go further into the smart grid, as well as some of the potential fraud tactics.

1.4.1 Definitions

A smart grid is "an application of digital information technology to optimize electrical power generation, delivery, and use" [17]. The term "smart grid" refers to the communication and control capabilities built into the traditional grid, which is a network of transmission lines, substations, transformers, and other components that transport electricity from a power plant to homes and businesses. The phrase smart grid is relatively new in the twenty-first century [18]. The smart grid is an important link in the process of energy conversion from sources to beneficial activities that fuel economic growth.

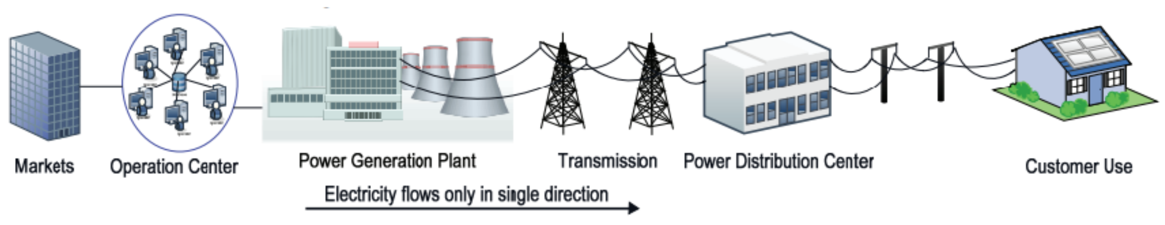


Figure 1.3: Conventional grid structure [19]

There are three reasons to convert the conventional grid to a smart grid [20] :

- **The increased energy demand:** The demand for energy is rapidly increasing due to new technologies like electric cars.

- **Decreasing the losses and illegal usage:** Losses and illegal use appear in transmission and distribution lines due to several reasons such as theft and fraud.

- **The increased producing and carriage capacity in the existing plants:** In order to integrate distributed energy sources such as solar energy and wind in order and meet increased demand for energy.

.

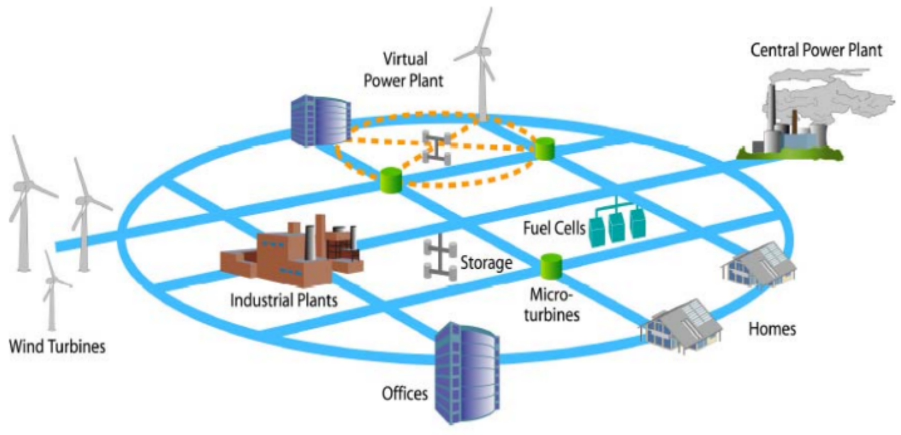


Figure 1.4: Smart grid structure [21]

1.4.2 Smart Grid Fraud Methods

In this section, we will identify some methods of NTL fraud in smart grid [22] :

(a) **Firmware rewrite**

A firmware is a piece of software that controls the functions of a smart meter micro-controller, such as metrology and communication. To gain complete control of the meter, the fraudster might modify or rewrite the firmware. The main motivation for the firmware update is to sell meter hacking kits for profit, which is done by organized crime. The major worry, though, may be taking electricity without being detected.

(b) **Impersonation attack**

An impersonation attack on a Smart Grid is carried out by replacing a legal meter with a fake one, stealing firmware to clone meters, and selling them for profit by a crime organization. Furthermore, a customer with inside knowledge may use a computer and special software to emulate a smart meter, where an adversary impersonates a smart meter and sends billing messages with manipulated time information.

(c) **Password cracking**

Password cracking occurs when a fraudster extracts a password for a harmful purpose such as theft, where the passwords are stored in smart meters and can even be transferred via messages. In this case, the fraudster may obtain the passwords. If the passwords are spoofed, it could directly lead to NTL fraud. For example, if the clock table password is spoofed, the fraudster can change the value of the local clock, causing peak time billing to shift to non-peak time billing.

(d) **Key spoofing**

Key spoofing is happening when a fraudster harvests encryption keys to decode messages or harass authentication procedures, such as extracting keys from smart meters, which relies on various side-channel attacks like channel timing attacks side, and cold boot attacks.

(e) **Man-in-the-middle attack**

Man-in-the-middle (MIM) attacks are common in NTL frauds because they eavesdrop on sensitive information such as passwords and keys in networks. MIM occurs when an attacker intrudes on communication between two smart meters or between a smart meter and the head end in a smart grid to make them believe they are interacting directly.

(f) **Message manipulation attack**

A message manipulation attack occurs in the smart grid when a fraudster modifies messages exchanged between a smart meter and the head end or between two smart meters. When the fraudster adjusts the time-related information of the billing messages, it could result in NTLs fraud.

To be more specific, we show two plots, one with fraud and one without, where the brown plot indicates fraud and the blue plot shows no fraud.

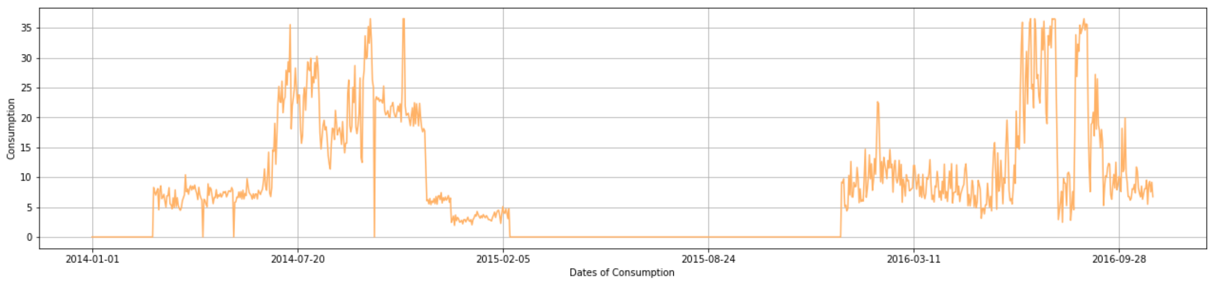


Figure 1.5: Consumer with fraud.

Figure 1.5 depicts an example of a customer's fraudulent electricity consumption over a period of approximately two years. We see that the electricity use data fluctuates month to month, with the value of consumption reaching zero in a period of time, which might be deemed fraud.

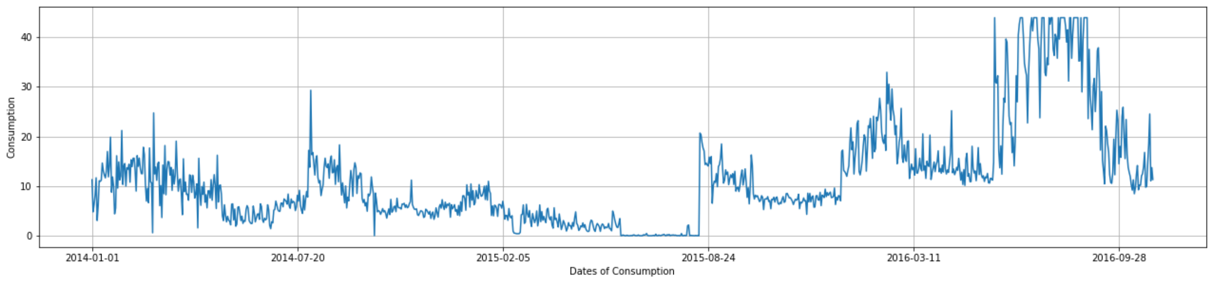


Figure 1.6: Consumer without fraud.

Figure 1.6, on the other hand, depicts an example of a customer's normal electricity consumption over a period of months. We can see that the electricity consumption data has a regular month-to-month variability.

The essential characteristics of electricity thieves and regular consumers are difficult to discern from these graphics. However, there is a slight difference between them that we can see.

Despite the development of the power grid into a smart grid, there is still an increase in energy transmission and distribution line losses, with electricity theft being the primary cause. Smart grids, on the other hand, can help to solve the problem of electricity theft because of the massive data generated, and this through a variety of techniques due to energy thieves' abnormal electricity consumption patterns.

1.5 Fraud Detection

Fraud detection is defined as the recognition of fraud's marks of which imposters do, that no prior suspicion or tendency to fraud exists [23], and that's as quickly as possible once it's detected. Fraud detection is an open field for development continuously to prevent criminals from adapting their methods where once they know that there is only one way to detect, criminals will adapt their strategies and try others. Obviously, new criminals are will not be aware of the fraud detection methods which have been successful in the past and will adopt strategies that lead to identifiable frauds. This means that the earlier detection tools need to be applied as well as the latest developments [24, 25].
Fraud detection is a number of processes and techniques that allow analysis to detect and prevent prohibited activity. This can include fraudulent credit card transactions, identity theft, government fraud, insurance fraud, and more. Businesses can benefit by integrating fraud detection into their websites, company policies, enhanced security features, or even employee training.
In the field of energy, and with the presence of non-technical losses that cannot be explained theoretically, which can be the result of external factors, most of which are fraud,

then fraud detection can help the company to know the problem it faces by detecting fraudsters and preventing their prohibited activities.

1.5.1 NTLs Detection Methods

Researchers are trying different methodologies for efficiently identifying fraudster customers. The existing methods for NTL detection can be categorized to [12]:

1.5.1.1 Studies Based on Social and Economic Conditions

Analysis study of factors and variables that may reveal the presence of NTLs over a specific population, tends to focus on analyzing demographic drivers and social aspects related to electricity fraud. The knowledge of these variables and factors may enable the detection of NTLs in a certain population or area. There are many studies using this method of detection like analysis of variables and factors which is making use of statistical techniques to find the relationships between socio-demographic, economic, and market variables and the amount of theft [26, 27]. Another analysis based on surveys, ethnographic fieldwork, and empirical analysis is used to understand the main factors related to theft [28]. Also an analysis of determinant socio-economic attributes of illegal consumers of electricity through econometric analysis [29].

Advantages:

- Useful in policy design and has a significant impact on decisions to reduce NTL.
- Ease of managing the complexity and amount of data available even if it consists of indicators and variables that cover the entire area.

Disadvantage:

- The main disadvantage of these techniques is that their scope is limited because it usually focuses on a specific country or region.
- Not enough to find specific cases of theft and malfunctions in measurement or billing.

1.5.1.2 Hardware-Based Solutions

The hardware-based solution is defined as a solution that depends on creating and designing a device that can help to allow the identification and estimation of any fraudulent activity [30, 12].

Several studies concentrate on the proposal of metering and sensing hardware of this type. We can mention that this method includes the following types of techniques:

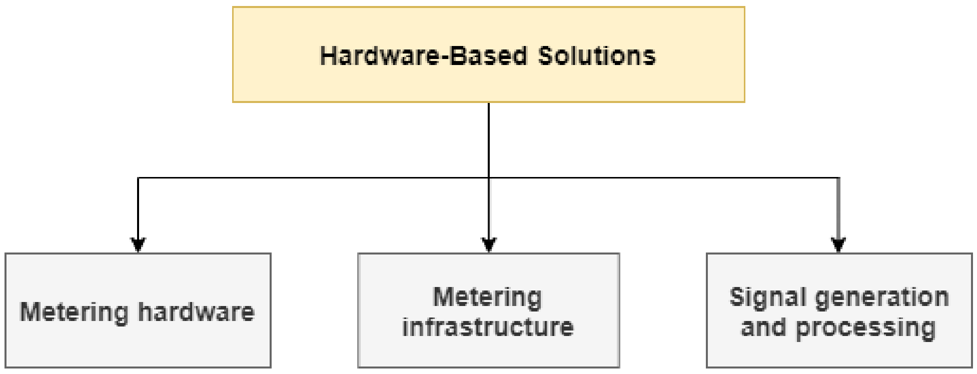


Figure 1.7: Hardware-Based Solutions.

(a) **Metering hardware:** A study in which the measurement of hardware details and specifications is determined, and offers different ways of designing meters or modifying the hardware in existing equipment for ease of theft detection. The problem of NTL detection was addressed by radio frequency identification (RFID) technology that blocks energy meters and speeds the inspection process [31], as well as a system based on specific processor architectures and algorithms [32, 33], which allows protection against tampering by detecting and reporting intrusion events. Metering hardware solutions can only detect NTLs that result from the metering zone [30].

- **Advantage:** Disable theft completely in some options, such as meter reversal and disconnection [30].
- **Disadvantage:** The high cost of installing hardware in a large number of households [30].

(b) **Metering infrastructure:** This technique focuses on infrastructure characteristics and the data collection equipment needed from different locations of the grid, that provide solutions based on a combination of metering assets and/or sensing hardware, such as installation strategies and a number of pieces of equipment needed based on geographical location, and this to effectively calculate NTLs and detect their sources [34, 35]. These studies have important aspects in utilities making decisions to modernize electrical networks [30].

- **Advantage:** The ability to detect any type of NTLs when the source is in the meter or before the meter areas [30].
- **Disadvantage:** High associated equipment costs [30].

(c) **Signal generation and processing:** This study presents solutions that take advantage of signal generation and processing to detect NTLs, and it provides practical ways to control and detect NTLs sources. Where this study is used on the use of a harmonic signal generator, after the separation of the legal consumer meters, and the introduction of a signal to the distribution feeders that affect the electrically connected equipment [36]. In addition, it used to detect illegally connected equipment, consumers were disconnected and a high-frequency signal generator and analysis were used, which contributes to increased loads in the distribution [37].

- **Advantage:** The ability to detect all types of NTLs in the grid [30].
- **Disadvantage:** Rely on having smart metering systems [30].

There are other approaches that do not fit with any type mentioned above, such as the use of a light sensor to collect information about public lighting points and this is done to ensure that the electricity used for municipal lighting is properly reported and paid for. In addition, the use of criminal investigation procedures to find possible causes of collusion and meter fraud [30].

1.5.1.3 Non-Hardware Based Solutions

This method focuses on the resolution of the non-hardware solution, for the hardware-based methods, which is not feasible for several Power Distribution Companies (PDCs) that need to install the new infrastructure, which includes huge amounts, especially those in underdeveloped countries. The software solution depends on describing classification techniques that conclude the presence or estimation of NTLs from consumers' electricity consumption data [12, 30].
The non-hardware-based solutions encompasses the major following types of techniques:

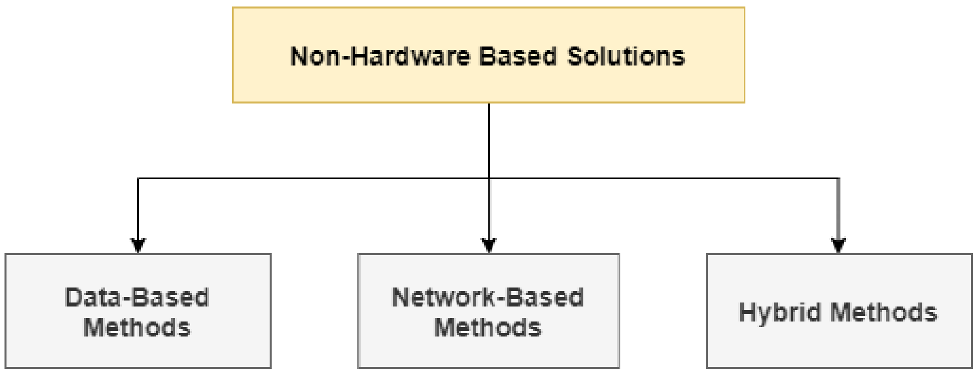


Figure 1.8: Non-Hardware Based Solutions.

(a) **Data-Based Methods:**

Data-based methods are a technique based on machine learning or data analysis, which in turn is divided into two methods that are supervised methods and unsupervised methods [12].

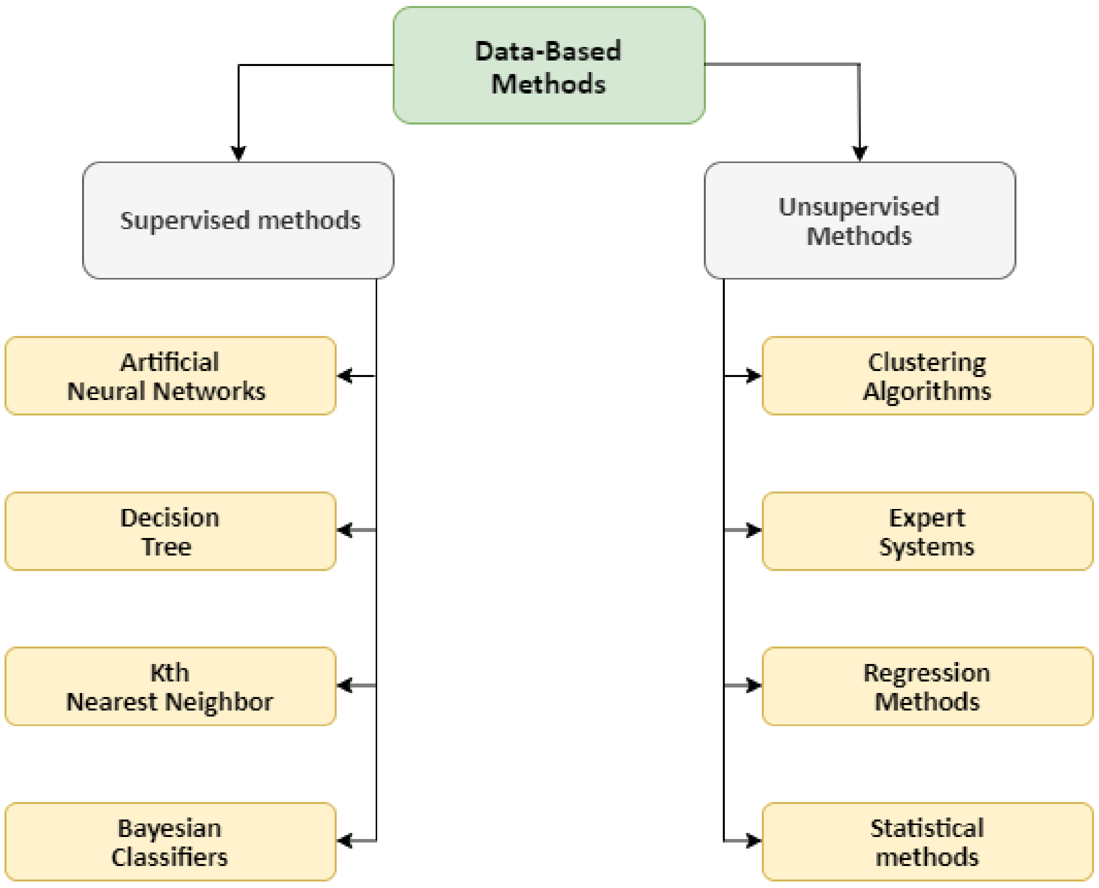


Figure 1.9: Data-Based Methods

As we can see in figure 1.9, the database methods are divided into two types:

The supervised machine learning includes methods that involve both the fraud and honest classes, and this is accomplished by training the classifier with labeled data [10]. The main drawback of this method is that it cannot learn in the absence of fraudsters' labeled data (fraud class). Furthermore, incorrect learning may occur if the number of fraud customers is significantly lower than the number of honest customers [38]. We'll go over some of the supervised methods that have been used to solve the NTL detection problem:
The support vector machine (SVM), which is immune to the class imbalance issue, is one of the main reasons why the researcher uses it to detect NTL. SVMs are sometimes combined with other classifiers such as fuzzy inference systems (FIS) to improve classification performance [39, 40]. ANN was also used to detect the NTLs, the most commonly used version of ANN is Multi-layer perceptron (MLP) [41, 42], which is

used as a binary classifier for NTL detection problems. Also, the use of the Decision Tree(DT) [43, 44] that aids in a better understanding of the characteristics of the NTL, the benefit of these algorithms is their ease of interpretation and transparency to the operatives. In addition, Bayesian classifiers are used to detect NTL by using the previous information probability that can be obtained from the general statistics [45]. As well K-NN [46] is used as a baseline for comparison with other machine learning algorithms for NTL detection.

Unsupervised machine learning refers to methods that do not require labels (Fraud/Non-fraud), implying that supervision is not required [47]. Unsupervised algorithms allow the model to learn the information hidden in the data on its own. Unsupervised methods have the advantage of being able to handle more complex processing tasks [43]. For the NTL detection problem, the following types of unsupervised machine learning algorithms were used:
Clustering algorithms have been used in the NTL detection problem several times [48, 49], mostly in the data processing phase. Clustering algorithms work by gathering similar customers with various power patterns, which are then trained to recognize and classify unlabeled data. Expert systems are methods that have been also used to detect NTLs in both supervised and unsupervised settings [50, 51]. Their main goal is to solve complex problems using logic and analysis, which is usually represented by if-then rules. Also, the use of Regression Methods, which depend on the difference between the expected and the calculated value if the regression model has been trained with one class of consumers' consumption data [52]. As well as Statistical methods, these methods are mostly used for the time-series data, which are working by observing the individual energy consumption and describing the areas, where the consumption pattern may be considered as anomalous for time series data.

(b) **Network-Based Methods**

For efficiently identifying NTL activity, network-based methods rely on information extracted from smart meters and the calculation of different physical parameters of the electrical network [53, 54]. The area where the fraudulent activity occurs is identified by calculating the energy stability and other parameters. While network-based methods are generally more accurate, they are difficult to implement. Three different types of NTL detection are used in network-based methods:

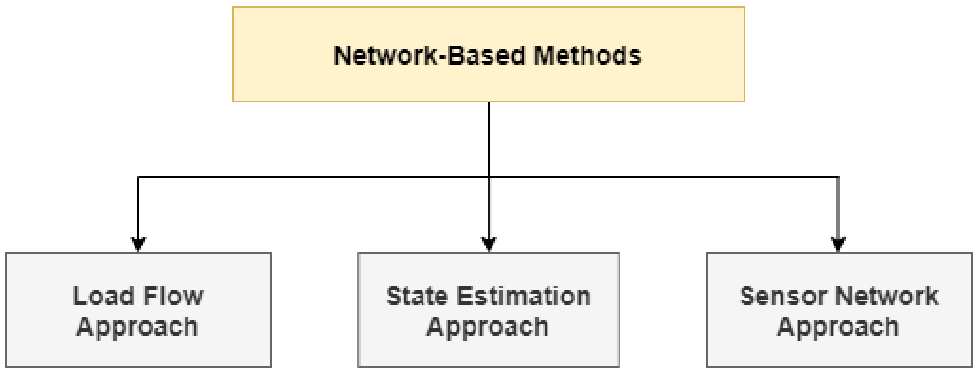


Figure 1.10: Network-Based Methods.

Load Flow Approach: In which a central observer meter is installed to monitor the energy meter in the distribution transformer's low voltage section [48]. The monitoring device's measurement, on the other hand, is compared to the total energy recorded by the smart meters. The total percentage of technical losses is then calculated using the difference between these two readings. The difficult part of these methods is accurately calculating the network's technical losses [43].
State Estimation Approach: Using data from smart meters to observe the grid, distribution state estimation (DSE) methodologies are an excellent scheme for the correct identification of fraudulent activity [55]. It's widely used in medium-voltage networks (MV). Due to the large amount of bad data that is displayed in a random and isolated manner, it is extremely difficult to detect fraud [43].
Sensor Network Approach: In network-oriented methods, the installation of certain sensors in the energy distribution system is a new development [56]. The goal of the latter could be to reduce infrastructure costs as well as calculate the assessment and optimal position as well as the number of sensors needed to locate NTL. These methods typically require precise data and information about the network structure. The majority of these models have the goal of increasing network inspection [43].

(c) **Hybrid Methods**

Hybrid-based approaches depend on both data-oriented and network-based methods by implementing the techniques and algorithms [39], all this for NTL detection with suitable accuracy. Where both SVM algorithms were used in conjunction with the Central observer meter, active power measurements of the network are calculated by checking the counter via SVM output. The SVM algorithm calculates the difference in the active power measurement and the system's technical losses. In addition, the installation of a remote technical unit (RTU) was also proposed to detect NTL [57], by identifying which subnets are vulnerable to electricity theft. With the help of SVM and fuzzy electronic means, the individual customers are identified who commit fraud. Another approach has been calculated for different abnormal situations for each adapter through an unsupervised anomaly algorithm. The weight matrix is then adjusted from the estimated status through that density, which calculates the position of loading the converter using false accounts and forecasting. TL and NTLs can then calculate the individual converter level [12].

1.6 Conclusion

In this chapter, we have covered some of the most essential terminologies in our current work including fraud definitions, energy usage definitions, and so on. Then we have present a broad overview of our selected topic which is smart grid energy usage.
We have also learned a lot about and how to spot energy fraud using a variety of techniques and methods. The goal of this chapter was to lay out the foundational ideas for our theme.

The next chapter will introduce and discuss one of the most useful detection methods used in deep learning.

Chapter 2

Convolutional Neural Networks

2.1 Introduction

The rising prevalence of energy fraud has necessitated the use of more stringent fraud detection methods. Deep learning is said to be one of the strategies that can be successfully used to detect any fraud activity.

In this chapter, we will go over the basics of machine learning and deep learning before diving into Convolutional neural networks (CNNs) and their different variations, and finally, we will go over some earlier relevant work.

2.2 Machine Learning

In 1959, Arthur Samuel described machine learning (ML) as the "field of study that gives computers the ability to learn without being explicitly programmed".

Machine learning (ML) is a rapidly evolving field that describes how computers can learn automatically, recognize complex patterns, and make intelligent decisions by using algorithms to synthesize the underlying relationships between data and information [58]. We can find ML in several uses, such as web search, credit scoring, behavior analysis, smart coupons, drug development, weather forecasting, big data analytics, and many other applications [59].

ML algorithms can be classified into four categories, which are supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning [60]. This is based on the underlying mappings between input data and anticipated output presented during the learning phase of ML.
Supervised learning is a learning mechanism that identifies input data and a targeted variable subject to predict [59]. Supervised learning techniques involve Linear Regression, Logistic Regression, Naïve Bayes, KNN and deep learning, etc...

Unsupervised learning algorithms are designed to discover hidden structures in unlabeled datasets, once the model is given a dataset, it automatically finds patterns and relationships in the datasets, in which the desired output is unknown [59]. Unsupervised learning techniques include Apriori, K-means, SVM, Anomaly Detection, Principal Component Analysis (PCA), etc.
Concerning semi-supervised learning is a class of machine learning techniques that make use of both labeled and unlabeled examples when learning a model. Semi-supervised ML methodology operates between the guidelines of unsupervised learning (unlabeled training data) and supervised learning (labeled training data) and can produce considerable improvement in learning [59].
Last but not least, Reinforcement learning is a type of machine learning which involves the presence of an agent that performs certain actions in an environment so as to maximize the reward in a particular situation. The learning technique synthesizes an adaptation model by training itself for a given set of experimental actions and observed responses to the state of the environment [59].

2.3 Convolutional Neural Networks

ConvNets or CNNs are one of the most common types of image recognition and classification algorithms. CNNs are commonly used in areas such as object detection, face recognition, and so on. But what exactly is it? First and foremost, we must comprehend the origins of CNNs.

2.3.1 Artificial Neural Networks

Artificial neural networks are based on early models of the brain's sensory processing [61]. By simulating a network of model neurons in a computer, an artificial neural network can be created [62]. Following McCulloch and Pitts' introduction of simplified neurons, the first wave of interest in neural networks, known as connectionist models or parallel distributed processing, arose (1943) [63]. Artificial neurons, which are parts of artificial neural networks, are driving deep learning and machine learning capabilities to think more like humans and produce more sophisticated cognitive results. Artificial Neural Networks have a main body called a node and three layers in their structure.

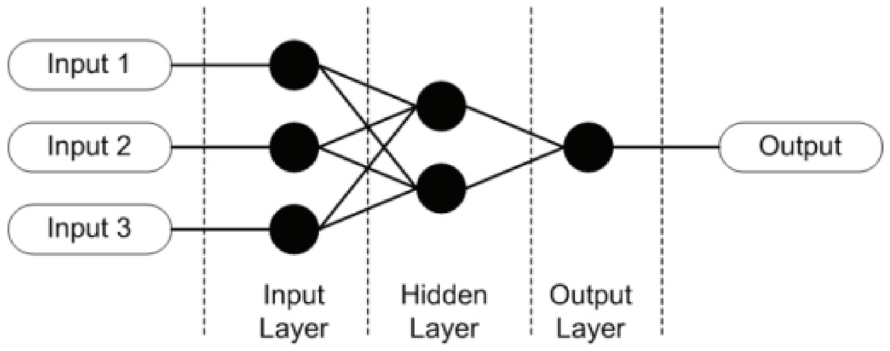


Figure 2.1: Simple artificial neural network [61]

As shown in figure 2.1, the input layer is the first layer and is the only one that is exposed to external signals. The input layer transmits signals to the neurons in the hidden layer, which extracts relevant features or patterns from the received signals. The output layer, which is the network's final layer, receives those features or patterns that are deemed important [64].

2.3.2 Deep Learning

Deep learning is a process of machine learning that allows computers to learn from experience and understand the world by learning to perform tasks that are natural for the brain [65]. Theoretical foundations of deep learning can be found in the literature on neural networks (NNs). Deep learning, on the other hand, accounts for the use of many hidden neurons and layers (typically more than two) as an architectural advantage combined with new training paradigms, as opposed to the more traditional use of NNs [66]. Deep learning is making significant progress in solving problems that have eluded the artificial intelligence community's best efforts for many years [67]. Deep learning (DL) methods have had a profound impact on identifying objects in images [68], transcribing speech into text [69, 70], anomaly detection [71], or products with users' interests [72]. When it comes to deep learning, there are many different types of neural networks. Deep learning architectures are built on top of these networks. RNNs [73] and CNNs are the two most common and widely used neural network architectures.

2.3.3 Convolutional Neural Networks

A convolutional neural network (CNN) was discovered by Hubel and Wiesel (Hubel Wiesel, 1962), which is a specific type of artificial neural network which is inspired by the functionality of the human brain [74], combined with a deep learning process that allows computers to simulate how learning occurs or might occur in the brain [75]. These two later allow CNN to identify automatically the relevant features without any human supervision [76]. CNNs apply to image processing, natural language processing, and other kinds of cognitive tasks.

Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have four main types of layers [76], which are:

Convolution layer: The convolutional layer is the first layer of a CNN, and it is the main building element that determines the output of connected inputs in local subregions. This is accomplished by convolving a group of learnable filters (kernels) across the width and height of the input data, then computing the scalar product between the filter's values and the input, resulting in a two-dimensional activation map for that filter. CNNs can train filters that activate when specific types of features are identified at a specific spatial position in the input [77].

Pooling layer: Additional convolutional layers can follow convolutional layers or pooling layers, which conduct dimensionality reduction, reducing the number of parameters in the input.

- There are two main types of pooling: Max pooling and Average pooling.

Activation function: Mapping the input to the output is the core function of all types of activation functions in all types of neural networks. This layer controls how the signal flows from one layer to the other.

- The most common activation function used in CNNs and other deep neural networks are non-linear activation functions such as Sigmoid ,Tanh and ReLU [78].

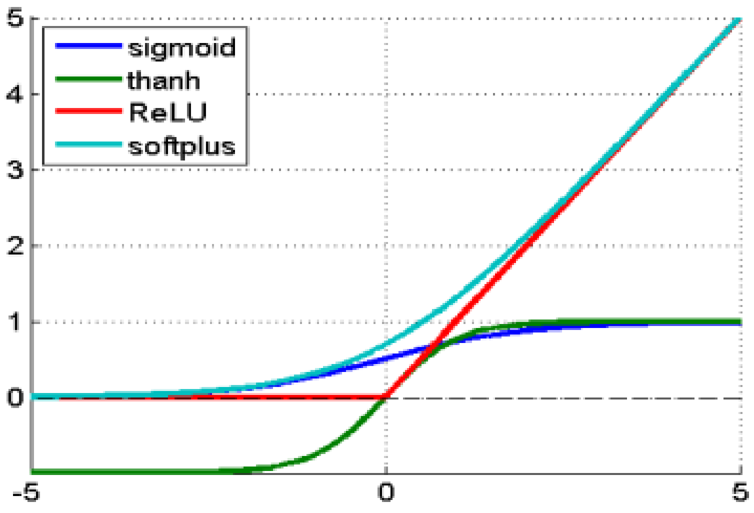


Figure 2.2: Common types of non-linear function [79].

Fully connected: In the fully-connected layer, each node in the output layer connects directly to a node in the previous layer. This layer performs the task of classification based on the features extracted through the previous layers and their different filters.

There are various architectures of CNNs available, the most popular two are [79] :

LeNET: Yan LeCun introduced LeNet for digit recognition, It includes 5 convolutional layers and one fully-connected layer.

AlexNet: AlexNet involves 5 convolutional layers and 2 fully-connected layers for learning features, It has max-pooling after the first, second, and fifth convolutional layer.

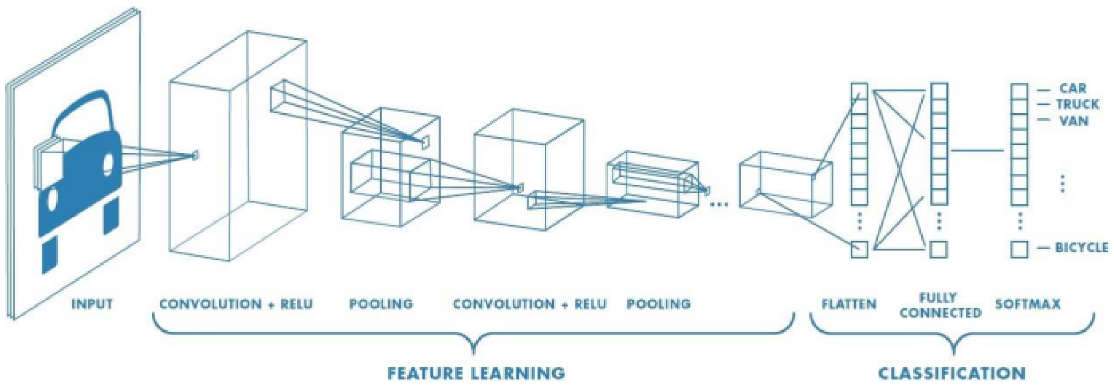


Figure 2.3: An example of CNN architecture for image classification [80].

Advantages of CNN

- CNNs combine the processes of feature extraction and classification into a single learning body. They can learn to optimize the features directly from the raw input during the training phase [81].
- CNNs can process large inputs with greater computational efficiency than traditional fully-connected Multi-Layer Perceptrons (MLP) networks [81].
- CNNs are unaffected by minor data transformations such as translation, scaling, skewing, and distortion [81].
- CNNs can adjust to a variety of input sizes [81].

Disadvantages of CNN

- CNN take sometimes much longer time to train [82].
- Large datasets and proper annotation are required for CNNs [82].

- CNNs are extremely sensitive to noise. [83, 84].

The famous deep CNNs can only process 2D data such as images and videos. This is why they're frequently referred to as 2D CNNs. As an alternative, 1D Convolutional Neural Networks (1D CNNs) have recently been developed as a modified version of 2D CNNs.

2.3.4 1D Convolutional Neural networks

A convolutional neural networks (CNNs) is an artificial neural network used in numerous modern artificial intelligence technologies such as the machine processing of image or audio data. The structure of a classic convolutional neural network consists of one or more convolutional layers, followed by a pooling layer and a fully connected layer [85]. These properties of CNNs are independent of the number of dimensions. One-dimensional CNNs (1D CNNs) work with patterns in one dimension and tend to be useful in signal analysis over fixed-length signals that it is effective at detecting simple patterns in data, which can then be used to create more complex patterns in higher layerst [86, 87]. When deriving features from fixed-length segments of the overall dataset and where the location of the feature within the segment is not important, a 1D-CNNs is very effective [87]. They work well for the analysis of audio signals. Also for some natural language processing(NTL), although recurrent neural networks, which allow for different sequence lengths, may be a better fit there, especially ones with memory gate arrangements such as LSTM or GRU. In short, the number of dimensions is a property of the problem being solved. For example, 1D for audio signals, 2D for images, and 3D for movies.

The different dimensions of CNNs (1D, 2D, or 3D) share the same properties and approach. However, the biggest difference between them is the dimensionality of the input data and the way the filters, also known as convolution kernel or feature detector, used move across the data.
In addition, The computational complexities of 1D and 2D convolutions differ significantly, for example, an image with NxN dimensions convolved with a KxK kernel has a computational complexity of O(N^2K^2), whereas the corresponding 1D convolution (with the same dimensions, N and K) has a computational complexity of O(NK). This means that a 1D-CNNs computational complexity is significantly lower than a 2D-CNNs under similar conditions (configuration, network, and hyperparameters)[81].
Moreover, deep 2D CNN training typically necessitates specialized hardware, such as Cloud computing or GPU farms. For training compact 1D CNNs with few hidden layers and neurons, however, any CPU implementation over a standard computer is feasible and relatively fast. [81].

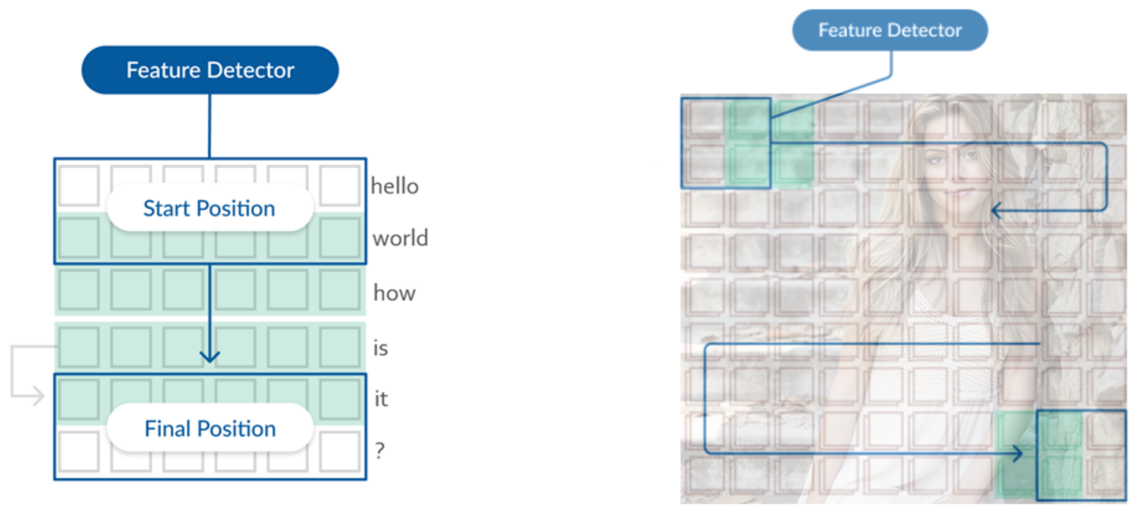


Figure 2.4: 1D CNN vs 2D CNN in feature detection.

As we seen in figure 2.4, an example of a 1D CNN for NLP is shown on the left, with the set consisting of six parts, each represented by a vector. The height of the Detector determines how many components of a sentence are taken into account during training, and the feature detector always covers the entire word. The feature detector would iterate five times over the data in this case because the height is two. On the right, we see an example of a 2D CNN based on computer vision, where each pixel of the image is defined by an x and y position. Additionally, each pixel contains the three RGB values. The dimension of the feature detector, in this case, is 2x2. As shown in the illustration 2.4, the feature detector moves horizontally and vertically across the image.

2.4 Related works

In this part, we shall discuss some of the earlier works that have used various deep and machine learning approaches to detect fraud in the energy sector.

2.4.1 Smart Grid Energy Fraud Detection using Artificial Neural Networks

Vitaly Ford et al. [88] applied Artificial Neural Networks to detect energy smart grid fraud. The researchers used actual smart meter consumption data from approximately 5,000 residential households and 600 businesses for almost two years (2009 to 2011), provided by the Irish Social Science Data Archive Center. For achieving better results in detection, they employed data cleaning, feature selection, and Indexing and compressing for data pre-processing.
The experiment was divided into two parts. The ECB neural network was first trained

by feeding the training set's energy consumption measurements into it, where the input layer contains 48×nodes in total. For simplicity, the hidden layer only has one layer of nodes. There is only one attribute in the output layer, which represents the expected value in the smart meter reading data series.

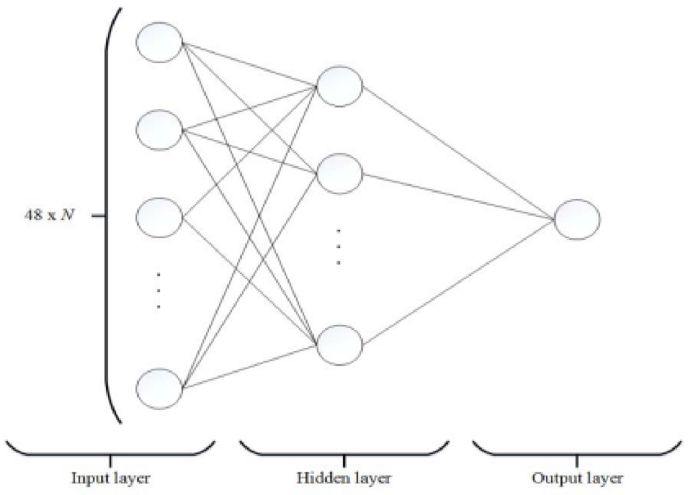


Figure 2.5: ECB Neural network structure [88].

Second, they simulated the scenario in which a malicious actor implants a chip inside a smart meter, slowing down the meter readings for an unknown period of time. They conducted three different types of experiments to gain more information. The first experiment assumed that consumer ECB could be predicted a year in advance, and the second experiment's goal was to see if a neural network trained for three weeks could predict and analyze the following week to detect simulated fraudulent activities. The aim of the third set of experiments was to see if energy fraud could be detected in the same weather season.
They reported confusion matrix reports true positives(93.75%), true negatives(75.00%), false positives(25.00%), and false negatives(6.25%), this means 84.37% of accuracy .

2.4.2 Nontechnical Loss Detection for Metered Customers in Power Utility using Support Vector Machines

Nagi et al [89]. utilized support vector machines for detecting fraudulent activities in a power grid. The researchers used Historical customer data from TNBD's electronic-Customer Information Billing System (e-CIBS) was obtained for the Kuala Lumpur (KL) Barat station, where the data consists of 265 870 customers for a period of 25 months. Additionally, they have High-Risk data containing information on the fraud customers previously detected by TNBD, which lists the detection dates of all the fraud customers detected. The proposed fraud detection framework consists of several steps including data pre-processing, feature extraction, classification, data post-processing, and identifying suspected customers.

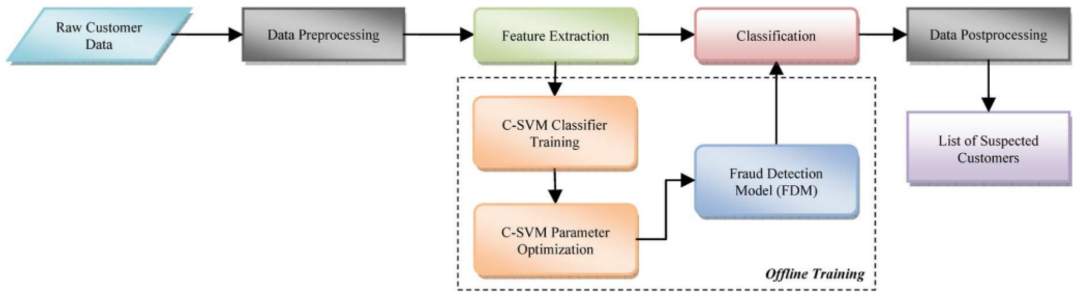

Raw Customer Data
Data Preprocessing
Feature Extraction
Classification
Data Postprocessing
C-SVM Classifier Training
C-SVM Parameter Optimization
Fraud Detection Model (FDM)
List of Suspected Customers
Offline Training

Figure 2.6: Proposed fraud detection framework for the detection of customers with abnormalities and fraud activities [89].

The support vector machine classifier was built using 330 profiles with no fraudulent activities, and 53 samples were flagged as abnormal. The experiments used data from three cities in the Malaysian state of Kelantan, which included energy consumption samples. They pointed out that in the previous 8 years, the percentage of energy fraud detected by the utility company was less than 1% of the total number of customers in each city. However, in those cities, the estimated rate of fraudulent activity was around 35%. They reported a 26% detection hit rate, which corresponds to the rate of detecting fraudulent activities. They training 67 % of the classifier data and testing the remaining 33 % was used to assess validation performance. This procedure was repeated 100 times in a row for tenfold CV, with training and testing data chosen at random each time. The tenfold CV was chosen to ensure that the classifier did not overfit the training data.
The optimal parameters were discovered experimentally, and the highest 10-fold CV training accuracy of 86.43 % was obtained.

2.4.3 Decision Tree Learning for Fraud Detection in Consumer Energy Consumption

C Cody et al. [44] they proposed using Decision Tree Learning to detect energy fraud in consumers. This paper uses the same data as the previous subsection, as well as the preprocessing data, which includes data cleaning, feature selection, indexing, and compression. The M5P decision tree learning algorithm was used in this study. The M5P decision tree algorithm's main goal in this problem space is to learn individual customer behavior in order to create an energy consumption model. The learned energy consumption model is then applied to forecast future measurements. With the ability to forecast future data.

They reported on the success of this method by repeating the previous subsection experiments.

2.4.4 Fraud Detection in Electric Power Distribution Networks using an ANN-Based Knowledge-Discovery Process

Costa et al. [90] proposed an approach based on neural networks and data mining techniques for detecting energy fraud. The researchers used data cleaning and integration, data selection and transformation, data mining, and pattern recognition techniques. They used an artificial neural network/multilayer perceptron (ANN MLP) for the dataset training and classification to avoid the problem of supervised classification one because there are real inspection records to be used for the training step. The researchers used actual data from a Brazilian electric power distribution company with more than seven million consumers. Their method identified 945 fraudulent activities among 1,453 fraudsters (a fraud detection rate of 65%), as well as 2,261 energy fraud cases among 20,130 non-fraudsters.
They reported The accuracy is equal to 87.17%, precision is equal to 65.03% and recall is equal to 29.47%.

2.4.5 Load Profiling and Data Mining Techniques in Electricity Deregulated Market

A. H. Nizar et al [91]. provided a method for handling electrical customer load profiles based on the knowledge discovery in databases (KDD) procedure and data mining techniques. The researchers used data collected from Malaysia from various categories of customer types: residential, commercial, industrial, and mining. Data on 200 customers' usage was collected over a six-month period, encompassing weekdays and weekends, with intervals of 30 minutes recorded by real-time meters on site, yielding 48 values per day for each client on each day. Their framework is divided into 5 phases which are problem statement, data collection, data pre-processing, data mining, and knowledge representation.

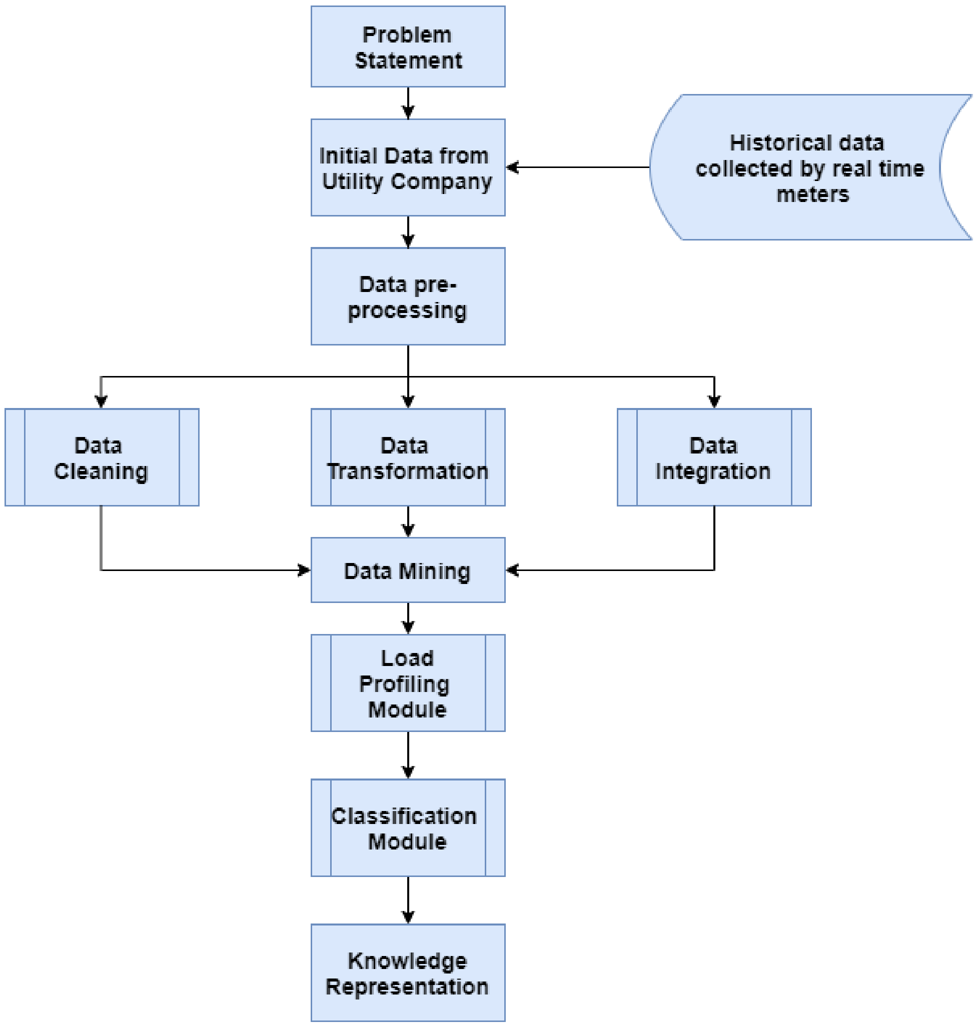


Figure 2.7: Methodology Framework [91].

The researchers compare and contrast three clustering approaches with the aim of categorizing power users into various categories which are the K-means method, EM method, and COBWEB method.

They have the result involves that COBWEB has the longest execution time of all of these approaches. In their datasets, however, the Simple K-means method performs better. The power consumers were grouped according to time of day variables and consumption usage characteristics based on the data set provided.

2.5 Conclusion

In this chapter, we have discussed some of the most important basics of our present work, including machine learning, deep learning, and neural networks. Inasmuch the Convolutional Neural networks is the major topic of this theme and has a connection to deep learning, it was chosen as the chapter's main emphasis. In addition we have presented a broad overview of our selected issue which is smart grid fraud detection, by presenting related work on the subject along with each technique.

The following chapter will cover the system's design, the dataset and its preparation, as well as the potential models.

Chapter 3

System design

3.1 Introduction

Due to the fast growth of machine learning and deep learning algorithms, as well as the researchers' effective applications in this area, the topic of fraud detection has gotten a lot of attentiveness from the recent past to the present. Deep learning convolutional neural networks have been shown to be quite effective in a variety of applications, including fraud detection.

In this theme, we want to develop models based on one-dimensional convolutional neural networks (1D-CNNs), as well as two-dimensional convolutional neural networks (2D-CNNs), to detect and categorize fraudulent customers. This chapter covers the system design, structural datasets, preprocessing phase, and finally our suggested CNNs models.

3.2 Methodology

In general, our system follows certain steps to detect fraud in the smart grid, which are shown in the figure 3.1:

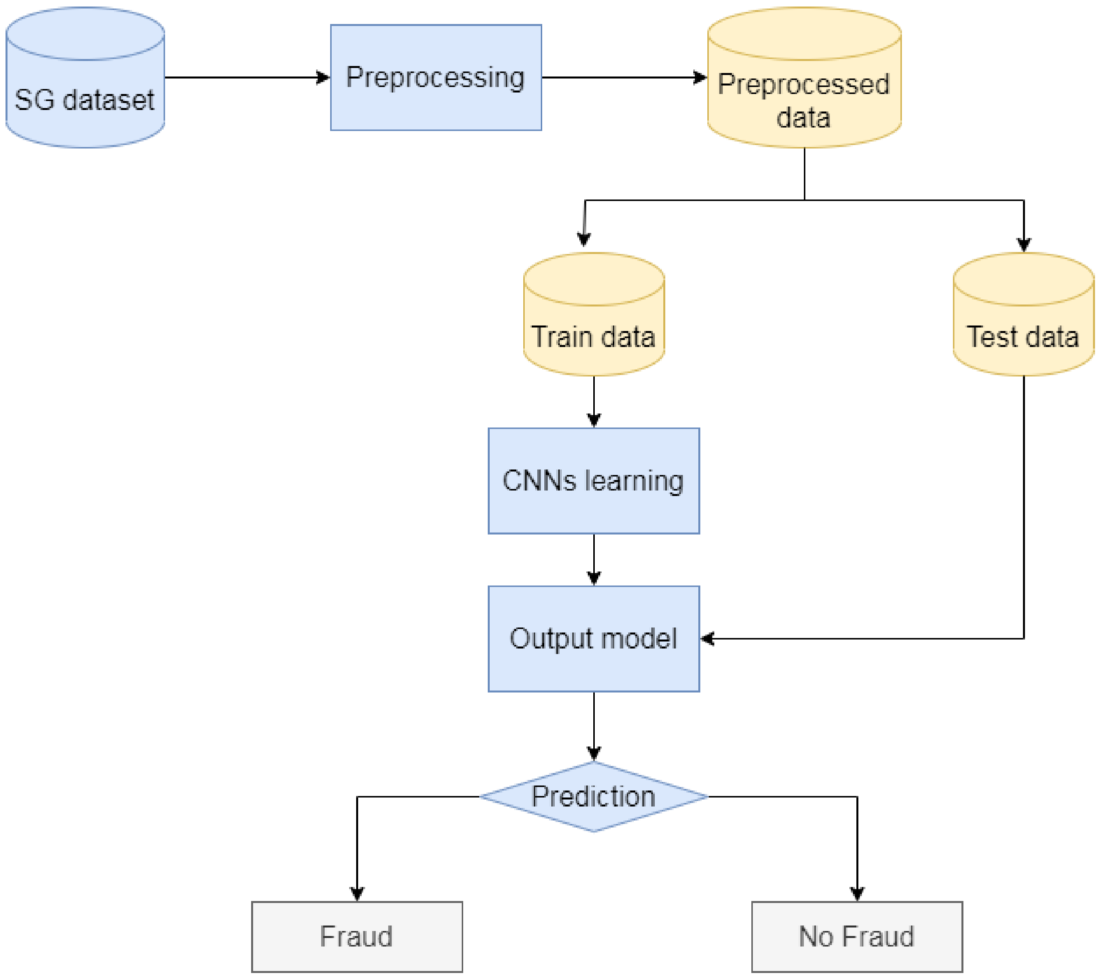


Figure 3.1: Diagram of the proposed system.

The system begins by obtaining the smart grid (SG) dataset of State Grid Corporation of China (SGCC), which is a one-dimensional data collection. After which, we performed preprocessing on the SG dataset, which required cleaning and filtering the database in order to make it suitable for training. A newly processed database will be divided into two sections: training data, whereby will be used to train our models, and testing data, for-which will be used to determine whether or not our models are valid. The next stage is to input a training dataset to our learning models, which are 1D/2D CNNs models, so that it may learn and construct a new predictive model. Once the models are ready, they will then be applied to the test dataset.

3.2.1 Dataset Description

We have used a dataset that is available on GitHub [92], which is a realistic electricity consumption dataset released by the State Grid Corporation of China (SGCC). This dataset contains the electricity consumption data of 42,372 electricity customers within 1034 days (1Jan 2014 to 31Oct 2016). The first 3615 people have been flagged as having committed fraud, while the rest were found to be truthful.

3.2.2 Preprocessing

Data preprocessing is a critical step that improves the quality of data in order to facilitate the extraction of meaningful insights from it. It also refers to the process of cleaning and organizing raw data in order to make it suitable for building and training Machine Learning models.
Preprocessed data is even more crucial than the most sophisticated algorithms, to the point where machine learning models trained on faulty data might be damaging to the analysis, resulting in "garbage" findings.
The real data that we use isn't complete or consistent. We must process this because it is primarily caused by various factors such as smart meter failure, the unreliable transmission of measurement data, and unscheduled system maintenance and storage issues. There are various stages required in this stage to accomplish this, including:

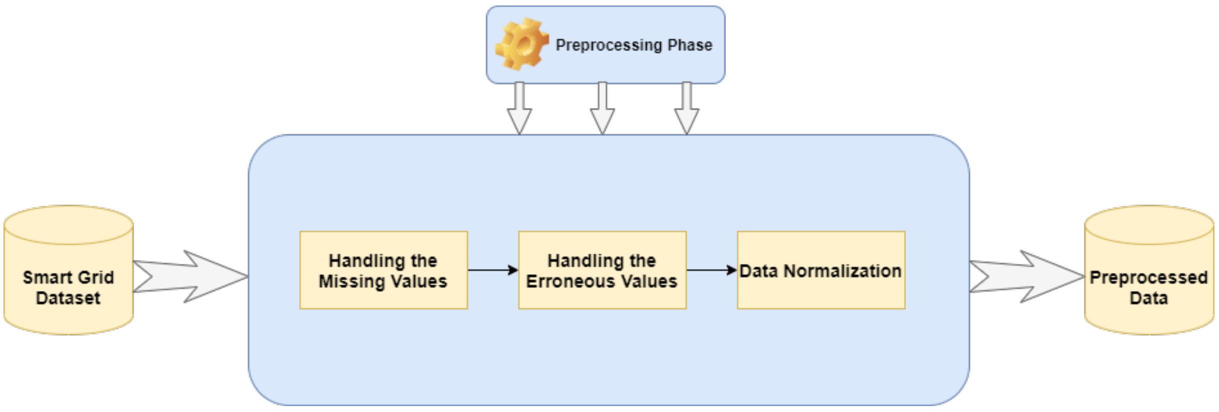


Figure 3.2: Preprocessing phase.

3.2.2.1 Handling the Missing Values

It is important to process missing values for successful data management. If missing values are not handled appropriately, it is possible that an incorrect inference about the data will be drawn as a result of the wrong treatment.
Our type of missing data is numerical data, shown as NaN. We exploit the interpolation method to recover the missing values according to the following equation:

$$F(x_i) = \begin{cases} \frac{x_{i-1}+x_{i+1}}{2} & \text{if } x_i \in NaN, x_{i-1}, x_{i+1} \notin NaN \\ 0 & \text{if } x_i \in NaN, x_{i-1} \textit{ or } x_{i+1} \in NaN \\ x_i & \text{if } x_i \notin Nan. \end{cases} \tag{3.1}$$

Where :

- Xi: The value in the electricity consumption data over a period.
- Xi+1: The next value of Xi.
- Xi-1: The preceding value of Xi.

3.2.2.2 Handling the Erroneous Values

Inputting inaccurate data is one of the most common data entry issues. An unintended error might cause immediate or long-term problems. It might also lead to inaccurate records, erroneous information, and disarray.
Before learning, we must correct some incorrect values in our data. As a result, we use the following equation to restore the value:

$$F(x_i) = \begin{cases} avg(x) + 2.std(x) & \text{if } x_i > avg(x) + 2.std(x) \\ x_i & \text{otherwise.} \end{cases} \tag{3.2}$$

Where:

- X: is a vector that is composed of xi day by day.
- avg(x): is the average value of x.
- std(x): is the standard deviation of x.

3.2.2.3 Data Normalization

Data normalization is a technique for organizing data characteristics to improve the coherence of different types of entities in a data model.
There are numerous goals to achieve throughout the data normalization process. The first step is to eliminate any duplicate data that may exist within the data collection.The second goal is to organize data rationally.
The neural network is sensitive to diverse data. To avoid this problem, we must first normalize the dataset.

We used the MAX-MIN scaling method, which is based on the following equation:

$$F(x_i) = \frac{x_i - min(x)}{max(x) - min(x)} \tag{3.3}$$

Where:

- min(x): is the minimum value in x.
- max(x): is the maximum value in x.

3.2.3 CNN Learning

Convolution layers, pooling layers, and fully connected layers are among the building components of the CNN architecture. A typical architecture comprises of one or more fully connected layers followed by a stack of many convolution layers and a pooling layer. Forward propagation on a training dataset is the step when input data is transformed into output through these layers. A back-propagation algorithm is a popular approach for training neural networks, and it relies heavily on the loss function and the gradient descent optimization process. A loss function calculates a model's performance under specific kernels and weights using forward propagation on a training dataset, and learnable parameters, such as kernels and weights, are changed according to the loss value using a backpropagation and gradient descent optimization procedure [85].

Although these convolution and pooling methods are used to describe 2D-CNNs, they can also be used to describe one-dimensional 1D-CNNs.

Some parameters should be given when training and fitting the model. Here are a few of them, briefly mentioned:

The learning rate: The learning rate is an adjustable hyperparameter that has a modest positive value, usually between 0.0 and 1.0, and is used in the training of neural networks.

The learning rate is a parameter that determines how quickly the model adapts to the situation. It's introduced as a constant (typically very modest) to encourage the weight to update in a steady and gradual manner (to avoid big steps andchaotic behaviour).

Optimizers in CNN: Optimizers are techniques or approaches that adjust the characteristics of neural networks, such as weights and learning rate, to reduce losses.
The optimizer in a CNN learning model can improve the model's outcomes and shorten the time it takes to train it from days to hours or minutes.
A few of the many optimizers available are : Gradient descent optimizer, Stochastic gradient descent, Mini batch gradient descent, RMSprop optimizer and Adam optimizer.

Loss function: To reduce algorithmic error, a loss function is used to calculate this error where it's used to determine how well or poorly the model is performing.

Epoch: The number of epochs is equal to the number of times the full dataset is transmitted forward and backward through the neural networks algorithm. Each epoch is completed when the algorithm has viewed all of the samples in the dataset.

3.2.4 CNNs Models

Deep CNN's high learning capability is due to the use of many feature extraction stages that can automatically learn representations from data.

Since we chose the smart grid field as the focus of our theme, the only data available was one-dimensional (1D) power consumption data, which is sequential data . For that, we are based on the 1D-CNNs model to detect fraudulent customers. As well as, We may also utilize the 2D-CNNs model to transform one-dimensional data into two-dimensional data by converting the input into a 33x33 matrix. In the end, we will compare the results of each model and decide which is better.

Because we have an idea about Convolutional Neural Networks in the previous section, let have a look at our CNN Models.

3.2.4.1 1D CNNs

For this experiment and for this entire chapter, a very simple CNN model is considered to consist of seven layers.

The first two-layer consists of input data with one dimension of 1034×1 with one BatchNormalization layer in between. It is convolved with 100 filters of size 7.
After the flattening step which is usually used as a connection between Convolution and the Dense layers, we end up with a long vector of input data that we then pass through the convolution neural network to further process it.
The fourth layer is a fully connected convolutional layer with 100 neurons. The fifth layer is also a fully-connected layer with 64 neurons. Last but not least the sixth layer is also involved in a fully connected convolutional layer with 32 neurons.

The seventh and final layer will be a sigmoid output layer with one potential class depending on the dataset.

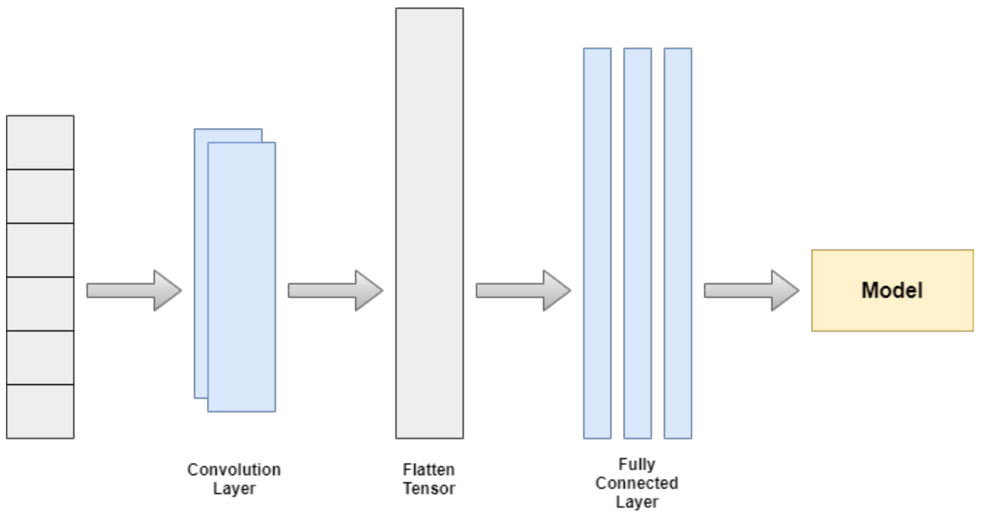


Figure 3.3: Training 1D CNNs model phase.

3.2.4.2 2D CNNs

For the purpose of this experiment, after we have converted our 1-D data into 2-D data, we choose a very simple CNN model that is considered to consist of four layers.

The first layer consists of input data with dimensions of 33×33. It is convolved with 32 filters of size 7×3.

Thus We end up with a lengthy vector of input data after the flattening stage , which we then run through the convolution neural networks to further process.

The second layer is a fully connected convolutional layer with 100 neurons.

Similarly, the third layer is also involved in a fully connected convolutional layer with 64 neurons.

The fourth and final layer will be a sigmoid output layer with one potential class depending on the dataset.

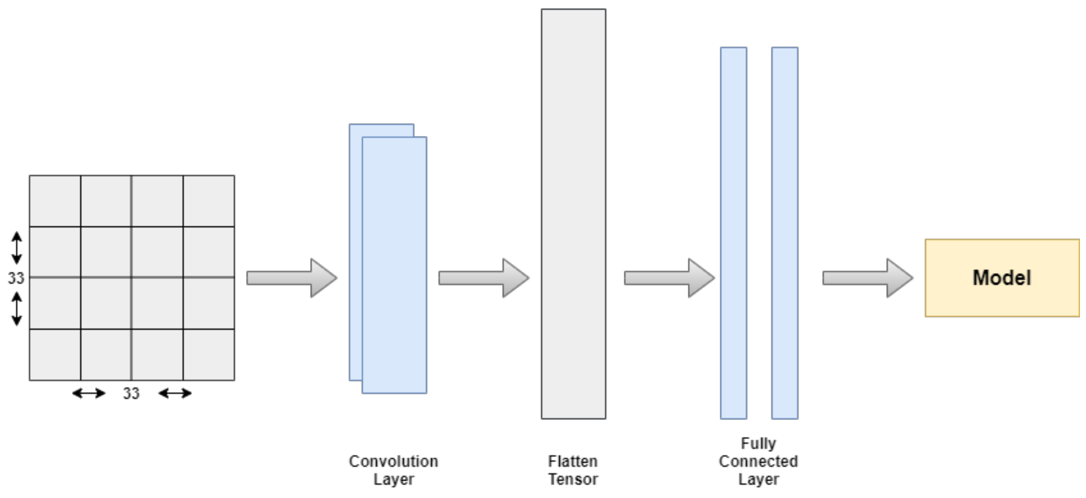


Figure 3.4: Training 2D CNNs model phase.

3.2.5 Prediction

The prediction phase follows the training phase, and it is at this point that a CNN model is ready to categorize the data where the accuracy is recorded. We have the second subset from the dataset splitting, which we will utilize to test our CNN model.

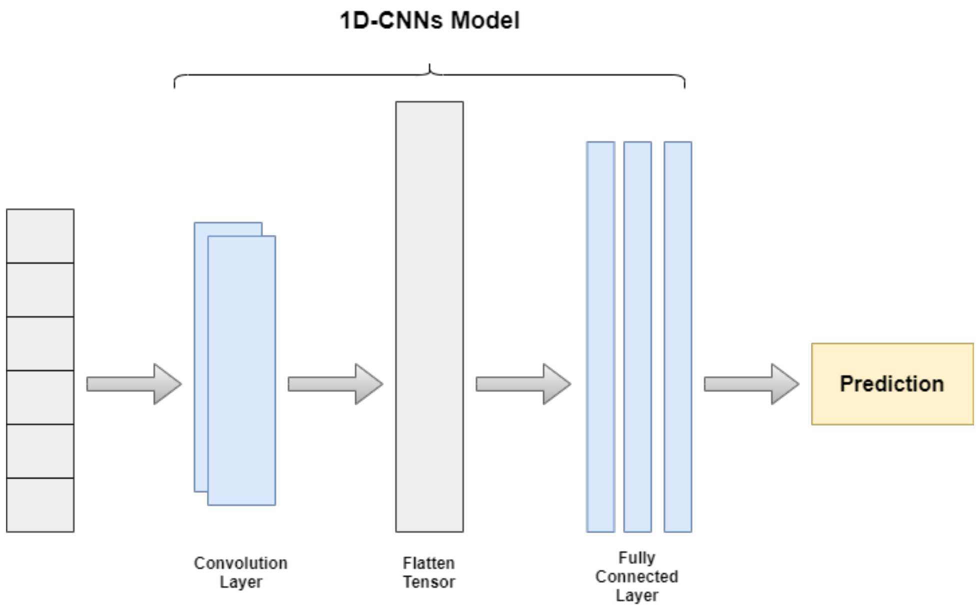


Figure 3.5: Testing 1D CNNs model phase.

Figure 3.5 depicted the architecture of how the CNN 1D model can predict fraud consumers which is applied in the test phase.

The first step contains the test data. Then we move on to the second: the two convolution layers. These last two convolutions contain 100 layers of neurons. The next step is a Flatten operation on a tensor that arrives from the previous step. Once Flatten operation is over, we pass it into the three Fully connected layers and pass it to the output layer that tells us if the consumer is a fraud or not.

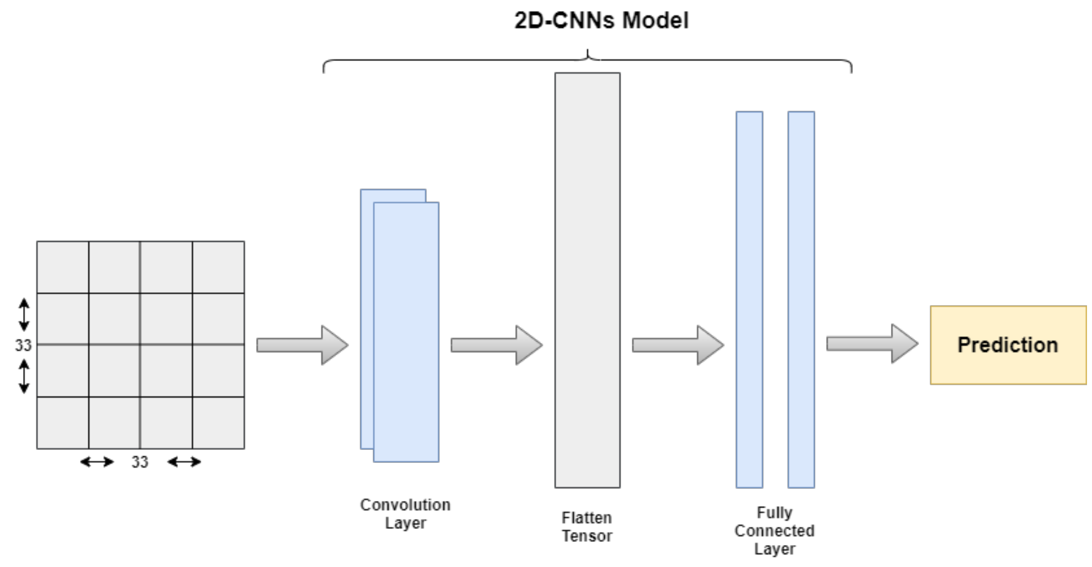


Figure 3.6: Testing 2D CNNs model phase.

Concerning figure 3.6 depicted the architecture of how the CNN 2D model can predict fraud consumers which also is applied in the test phase.

The first step contains the 2D test data. Then we move on to the second: the two convolution layers. The next step is a Flatten operation on a tensor that arrives from the previous step. Once Flatten operation is over, we pass it into the two Fully connected layers and pass it to the output layer that tells us if the consumer is a fraud or not.

3.2.6 Usage

The usage phase depicts the system's actual implementation, from which utility businesses can gain. We will go through this way when we have tested our models and found that they work well by getting better results.

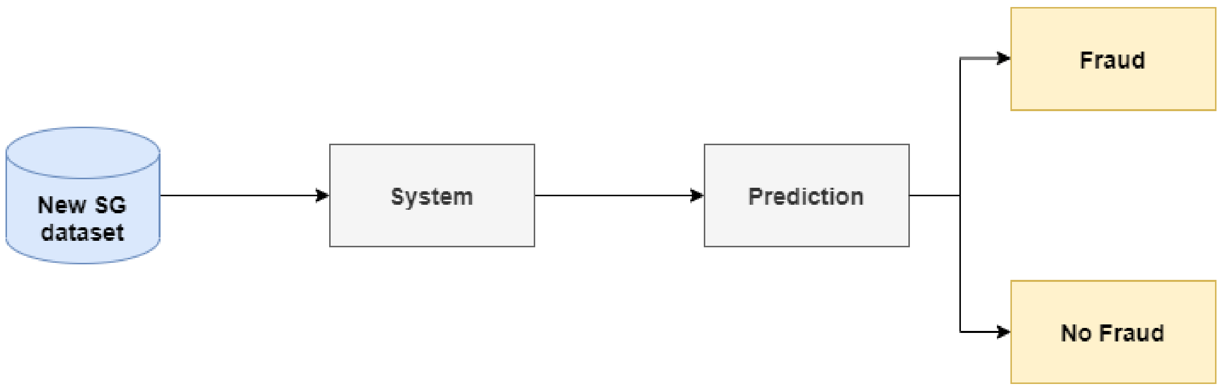


Figure 3.7: System usage.

3.3 Conclusion

The goal of this chapter is to provide a broad and thorough overview of our system architecture, which includes our datasets and what we performed during the preprocessing step, in addition to the CNNs models that we employed.

In the following chapter, we will go through the various tools, frameworks, and libraries that we have utilized, as well as how we have constructed our fraud detection models, and we will talk about the results we obtained and which one is the best based on our data.

Chapter 4

Implementation and results

4.1 Introduction

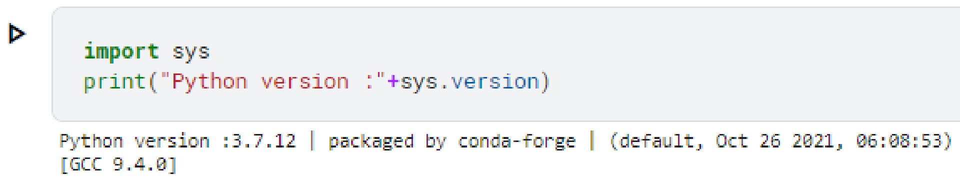
After detailing our system design in the previous chapter, as well as demonstrating the datasets and their preparation, and providing an overview of our models.
We will now move on to the following chapter, which will cover tools and frameworks, implement our system design, and present the coding side. Following that, we will go over into the models we ran and the results we got.

4.2 Implementation Frameworks and Tools

Deep learning can be implemented using a number of different tools. These open-source tools can be accessed on the internet. This section goes through the various development frameworks and tools that were used in this theme.

4.2.1 Python

Python is an object-oriented, high-level programming language with dynamic semantics that is interpreted. Its high-level built-in data structures, paired with dynamic typing and dynamic binding, making it ideal for Rapid Application Development as well as a scripting language for connecting existing components. The Python interpreter and substantial standard library are free to download and distribute in source or binary form for all major platforms.
The python version used in this work is 3.7.12



```
import sys
print("Python version :"+sys.version)
```

```
Python version :3.7.12 | packaged by conda-forge | (default, Oct 26 2021, 06:08:53)
[GCC 9.4.0]
```

Figure 4.1: Python version used in kaggle.

4.2.2 Matplotlib

The Matplotlib library is a Python library that is used to plot data. Matplotlib is an object-oriented API for using GUI toolkits to embedding plots in applications. Michael Droettboom is the creator of the Matplotlib library, which was written by John D. Hunter. Matplotlib is designed to look like MATLAB (Which Means Matrix Laboratory), which was the most popular programming language in academia at the time.
Matplotlib's popularity is due to the fact that it was released at a time when python was gaining popularity among programmers, which aided matplotlib's popularity among big data-loving programmers.

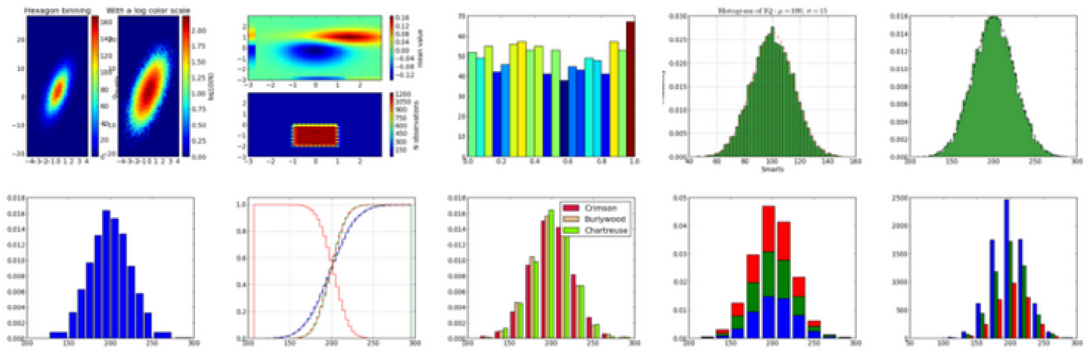


Figure 4.2: Exemples of matplotlib gallery.

4.2.3 Tensorflow

TensorFlow is an open-source end-to-end machine learning platform. It's a symbolic math toolkit that employs dataflow and differentiable programming to handle a variety of tasks related to deep neural network training and inference. It enables programmers to design machine learning applications utilizing a variety of tools, frameworks, and open-source resources.
TensorFlow is the finest library of them all since it is designed to be user-friendly. The Tensorflow library includes a variety of APIs for creating large-scale deep learning architectures such as CNNs and RNNs. This application debugging tool is quite useful. Tensorflow is designed to be used in large-scale deployments. It runs on both the CPU and the GPU programmers.

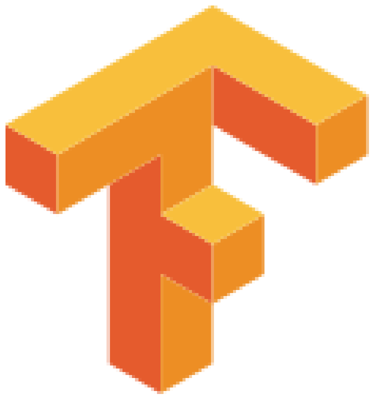


Figure 4.3: Tensorflow logo.

4.2.4 Keras

Keras is a high-level open-source neural network library written in Python that may be used with Theano, CNTK, or TensorFlow. Franco is Chollet, a Google developer, came up with the idea. For speedier neural network testing, it is extendable, user-friendly, and scalable. It supports CNNs both separately and in tandem.

4.2.5 Kaggle

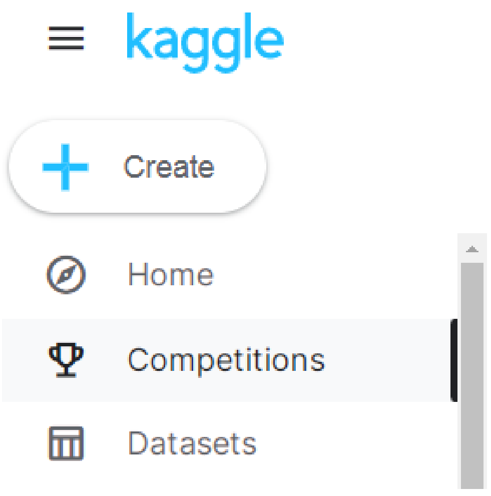
Kaggle is a data scientist and machine learning enthusiast's online community platform. Kaggle allows users to solve data science issues by collaborating with other users, finding and publishing datasets, using GPU integrated notebooks, and competing with other data scientists. With the sophisticated tools and information it provides, this online platform (established in 2010 by Anthony Goldbloom and Jeremy Howard and bought by Google in 2017) aims to assist professionals and learners realize their data science goals. Kaggle has about 8 million registered users as of now (2021).

The use of GPU in deep learning

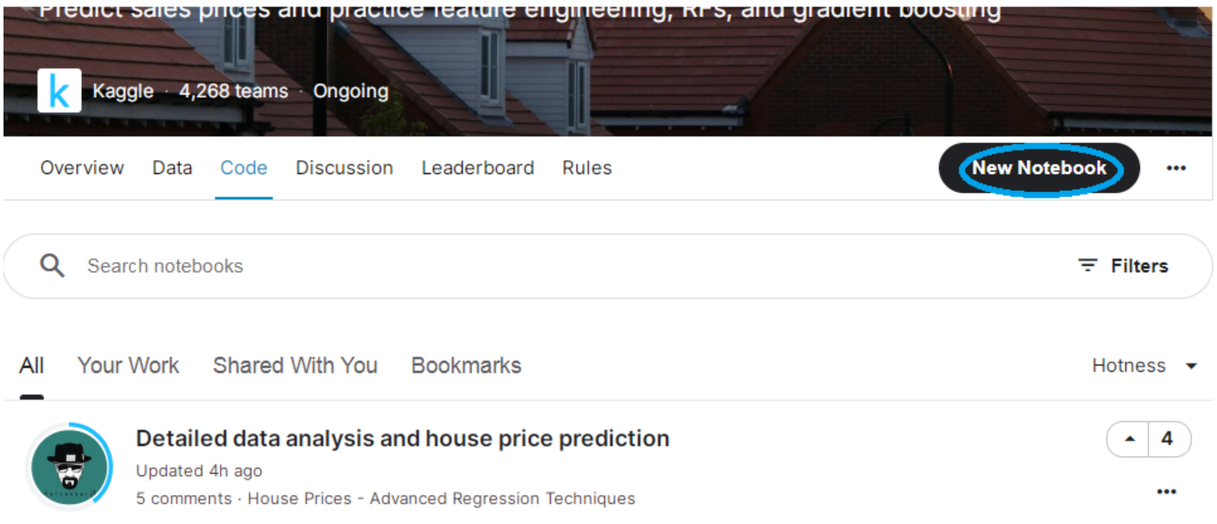
Graphics processing units (GPUs) are specialized processing cores that may be used to accelerate computations. These cores were created with the intention of processing photos and visual data. GPUs, on the other hand, are increasingly being used to improve other computational processes like deep learning. This is due to GPUs' ability to do numerous calculations at the same time. This allows training procedures to be distributed and can considerably speed up machine learning activities. You can have a lot of cores with GPUs and consume less resources without compromising efficiency or power.

How to create a project on Kaggle

First, we need to create an account on Kaggle. Then, on the left side of the page, we tap the competition button and select any competition.



After that, we pick New Notebook from the code button, and with that, we are ready to put our code into action.


Predict sales prices and practice feature engineering, RFs, and gradient boosting
Kaggle · 4,268 teams · Ongoing
Overview
Data
Code
Discussion
Leaderboard
Rules
New Notebook
Search notebooks
Filters
All
Your Work
Shared With You
Bookmarks
Hotness
Detailed data analysis and house price prediction
Updated 4h ago
5 comments · House Prices - Advanced Regression Techniques
4

4.2.6 Pandas

Pandas [93], which gets its name from *panel data*, is a python library built by Wes McKinney, who created it to assist him in working with datasets in Python for his job in finance. Pandas was released as an open-source library in 2009. It is a data analysis and manipulation tool based on the Python programming language that is quick, powerful, flexible, and simple to use.

4.2.7 Numpy

NumPy is the most important Python module for scientific computing. It's a Python library that includes a multidimensional array object, derived objects (such as masked arrays and matrices), and a variety of routines for performing fast array operations, such as mathematical, logical, shape manipulation, sorting, selecting, Input/Output, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation, and more.

Figure 4.4: Numpay logo.

4.3 Evaluation Metrics

Choosing the correct measure is very important when analyzing deep learning models. Deep learning models are evaluated using a variety of metrics in diverse applications. To evaluate our deep learning models, we used the following measures:

4.3.1 Confusion Matrix

A confusion matrix is a method for summarizing a classification algorithm's performance, where it calculates:

True Positive: The TP is when a model accurately predicts the positive class.

False Positive: The FP occurs when a model predicts the positive class incorrectly.

True Negative: The TN is a model that accurately predicts the negative class.

False Negative: The FN occurs when a model predicts the negative class incorrectly.

4.3.2 Accuracy

In machine learning, accuracy is defined as the percentage of right predictions in all predictions made.

It is calculated by dividing the number of true positives and true negatives by the number of true positives, true negatives, false positives, and false negatives.

$$Accuracy = \frac{TN + TP}{TN + TP + FN + FP} \tag{4.1}$$

4.3.3 AUC

The term "AUC" refers to the area under the curve ROC. AUC is a binary classifier's performance statistic. It's a statistical metric that we may use in a probabilistic environment to evaluate model predictions.
In short, the ROC curve depicts the connection between false-positive rate and true positive rate for various model prediction probability levels. In another hand, the AUC is a measurement of the complete two-dimensional area underneath the whole ROC curve where:

$$TPR = \frac{TP}{TP + FN} \qquad FPR = \frac{FP}{FP + TN} \tag{4.2}$$

The AUC or ROC curve plots the proportion of true positives vs the proportion of false positives. Sensitivity is another name for True Positive Rate. The rate of false positives is also known as (1-Specificity). The Y-axis represents sensitivity, whereas the X-axis represents (1-Specificity), as illustrated in figure 4.5

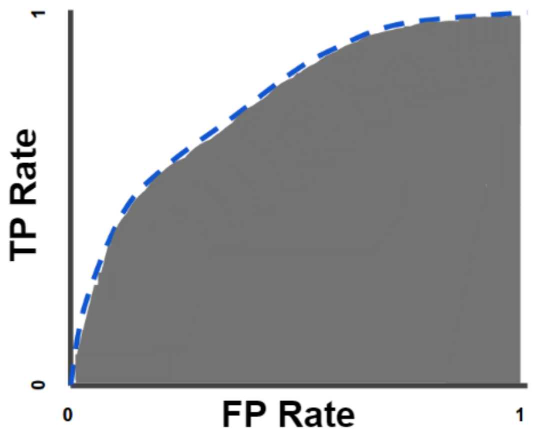


Figure 4.5: Area under the ROC Curve.

4.3.4 Precision

Precision is defined as the ratio of true positives to the sum of true positives and false positives, or how well we predict across all classes. To determine the accuracy of a model, we require the positive and negative integers from the confusion matrix.

$$Precision = \frac{TP}{TP + FP} \tag{4.3}$$

4.3.5 Recall

Recall, also known as Sensitivity, is the proportion of accurately predicted positive observations to all observations in the actual class. In short, Recall measures the model's ability to discover all positive individuals.

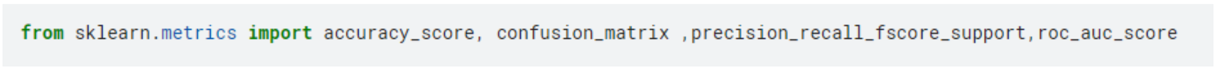
$$Recall = \frac{TP}{TP + FN} \tag{4.4}$$

4.3.6 F1-Score

The F1 score is a metric to evaluate the performance of classification models. The F1 score summarizes the values of precision and recall in a single metric. Mathematically, the F1-score is defined as the harmonic mean of precision and recall, which results in the following equation 4.5:

$$F1 - Score = \frac{TP}{TP + \frac{1}{2}(FN + FP)} \tag{4.5}$$

Note: In our theme, we want to evaluate our models based on accuracy and AUC.



```
from sklearn.metrics import accuracy_score, confusion_matrix ,precision_recall_fscore_support,roc_auc_score
```

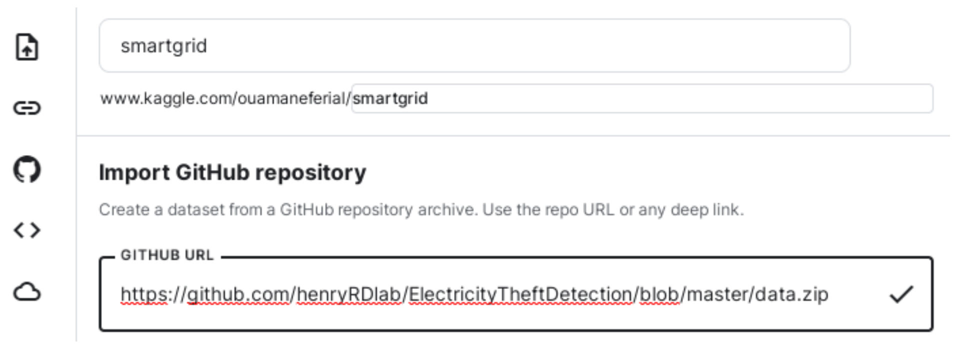
Figure 4.6: Metrics importation.

4.4 Implementation Phases

In this subsection, we will go over our implementation phases, in which we built and evaluated our system utilizing Python scripts and the Kaggle environment.

4.4.1 Loading and Preprocessing the Datasets

First, we used the dataset url to download the datasets from GitHub to our Kaggle account.



smartgrid

www.kaggle.com/ouamaneferial/smartgrid

Import GitHub repository

Create a dataset from a GitHub repository archive. Use the repo URL or any deep link.

GITHUB URL

https://github.com/henryRDlab/ElectricityTheftDetection/blob/master/data.zip ✓

Figure 4.7: Datasets importation.

Second, we read our dataset after downloading it and giving it a name of ”smartgrid”.

```
Data = pd.read_csv('../input/smartgrid/data/data.csv')
```

Figure 4.8: Read the data.

Concerning the preprocessing phase, the data that we use is not complete and consistent, so we will have to go through the following procedures to get our data ready for the training phase:

(a) First of all, we need separate the first two columns, which are “FLAG” (the class that indicates whether or not a customer is a fraudster) and “CONS_NO” (the consumer ID), as it is shown in figure 4.9.

```
infoData = pd.DataFrame()
infoData['FLAG'] = Data['FLAG']
infoData['CONS_NO'] = Data['CONS_NO']
data = Data.drop(['FLAG', 'CONS_NO'], axis=1)
```

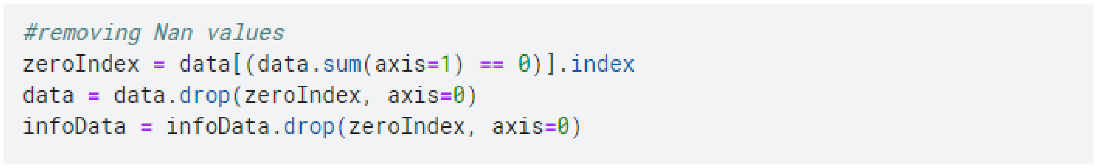
Figure 4.9: Separate the data.

(b) After we have separated the first two columns, we will go on to the first step of our preprocessing which requires dropping duplicate values/index that exist when two rows are same.

```
#droping duplicate row
dropIndex = data[data.duplicated()].index
data = data.drop(dropIndex, axis=0)   #dropping duplicate value
infoData = infoData.drop(dropIndex, axis=0) #dropping duplicate index
```

Figure 4.10: Dropping duplicate row.

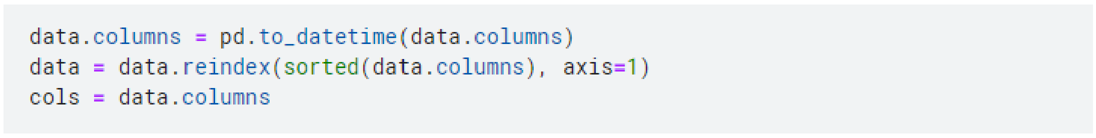
(c) The third step requires removal the row with containing all zero (Nan) values.



```
#removing Nan values
zeroIndex = data[(data.sum(axis=1) == 0)].index
data = data.drop(zeroIndex, axis=0)
infoData = infoData.drop(zeroIndex, axis=0)
```

Figure 4.11: Removing Nan values.

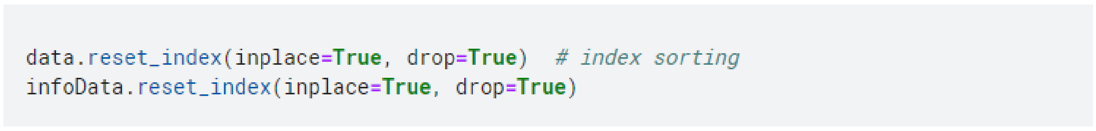
(d) After, we transform column names to dates (YYYY/M/D to YYYY-M-D) for the purpose of sorting data by date where certain columns are unsorted.



```
data.columns = pd.to_datetime(data.columns)
data = data.reindex(sorted(data.columns), axis=1)
cols = data.columns
```

Figure 4.12: Sort data according to date.

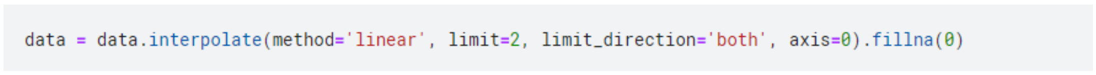
(e) After removing duplicate rows, and rows with Nan values we have to reindex row name.



```
data.reset_index(inplace=True, drop=True)  # index sorting
infoData.reset_index(inplace=True, drop=True)
```

Figure 4.13: Reindex row name.

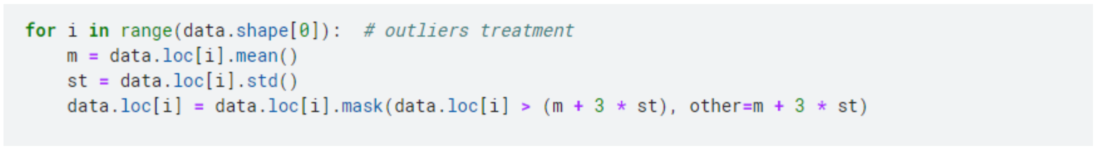
(f) Then, we select to fill Nan values with neighboring values, with the middle missing value replaced by average and the other by the maximum 2 distance element.



```
data = data.interpolate(method='linear', limit=2, limit_direction='both', axis=0).fillna(0)
```

Figure 4.14: Filling Nan values.

(j) In addition, we have to correct outliers by deleting erroneous data as illustrated in the figure 4.15.

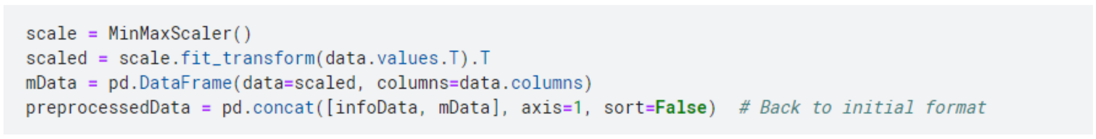


```
for i in range(data.shape[0]):  # outliers treatment
    m = data.loc[i].mean()
    st = data.loc[i].std()
    data.loc[i] = data.loc[i].mask(data.loc[i] > (m + 3 * st), other=m + 3 * st)
```

Figure 4.15: Removing erroneous values.

(h) Finally, since the neural network is sensitive to a wide range of data, we employed the MAX-MIN scaling approach to normalize the input.

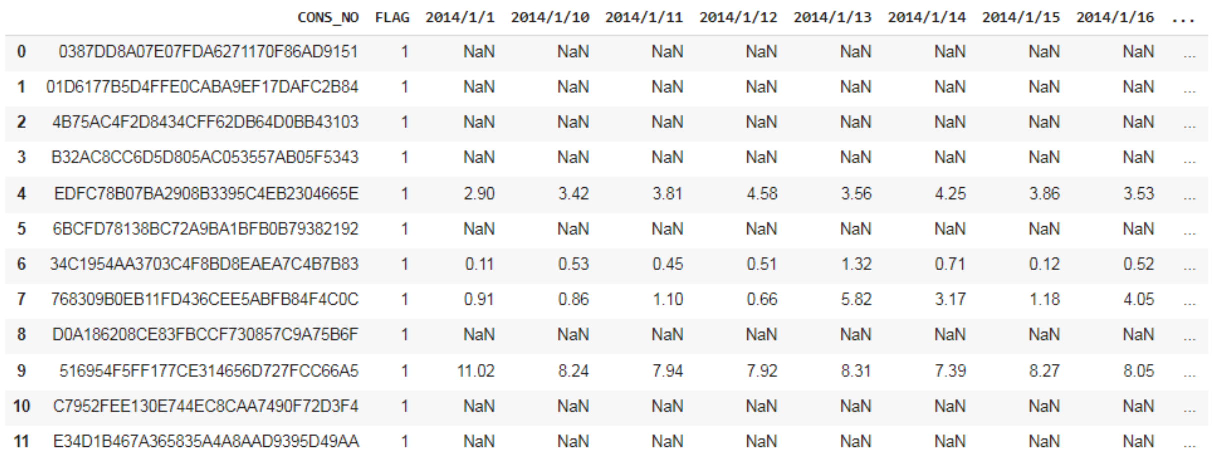
Once this process is over we will back to the initial format by concatenating the first two-row with other data as illustrated in the figure 4.16.



```
scale = MinMaxScaler()
scaled = scale.fit_transform(data.values.T).T
mData = pd.DataFrame(data=scaled, columns=data.columns)
preprocessedData = pd.concat([infoData, mData], axis=1, sort=False)  # Back to initial format
```

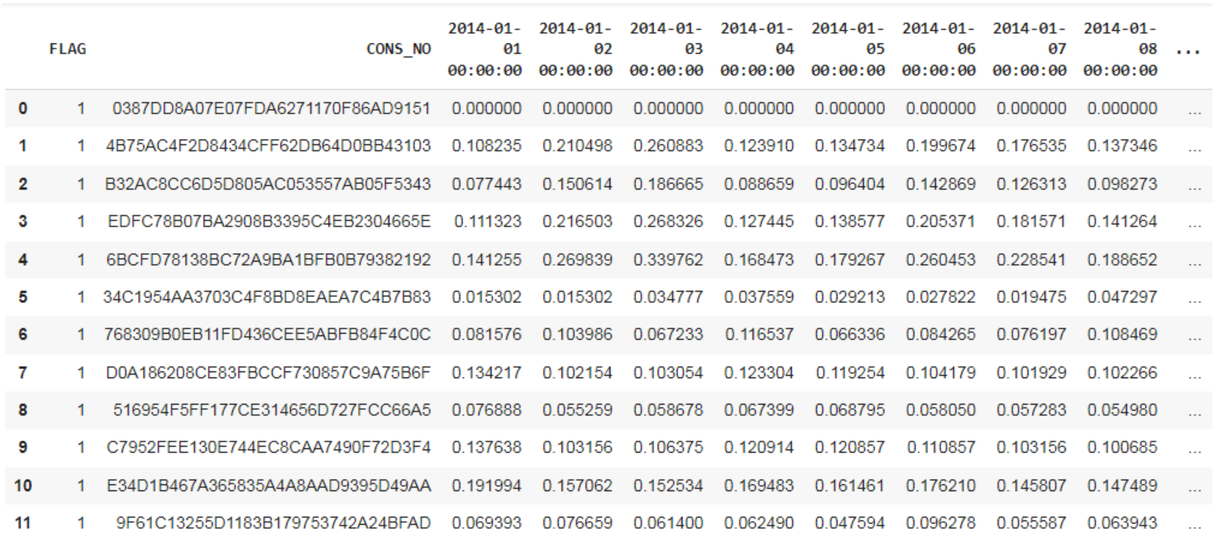
Figure 4.16: Normalization data process.

As a consequence, we will show how the data looked before and after the preprocessing process.



	CONS_NO	FLAG	2014/1/1	2014/1/10	2014/1/11	2014/1/12	2014/1/13	2014/1/14	2014/1/15	2014/1/16	...
0	0387DD8A07E07FDA6271170F86AD9151	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
1	01D6177B5D4FFE0CABA9EF17DAFC2B84	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
2	4B75AC4F2D8434CFF62DB64D0BB43103	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
3	B32AC8CC6D5D805AC053557AB05F5343	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
4	EDFC78B07BA2908B3395C4EB2304665E	1	2.90	3.42	3.81	4.58	3.56	4.25	3.86	3.53	...
5	6BCFD78138BC72A9BA1BFB0B79382192	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
6	34C1954AA3703C4F8BD8EAEA7C4B7B83	1	0.11	0.53	0.45	0.51	1.32	0.71	0.12	0.52	...
7	768309B0EB11FD436CEE5ABFB84F4C0C	1	0.91	0.86	1.10	0.66	5.82	3.17	1.18	4.05	...
8	D0A186208CE83FBCCF730857C9A75B6F	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
9	516954F5FF177CE314656D727FCC66A5	1	11.02	8.24	7.94	7.92	8.31	7.39	8.27	8.05	...
10	C7952FEE130E744EC8CAA7490F72D3F4	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...
11	E34D1B467A365835A4A8AAD9395D49AA	1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...

Figure 4.17: Smart Grid dataset befor preprocessing.



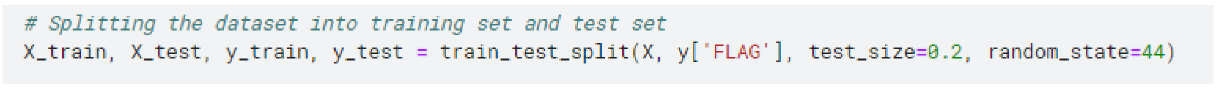
	FLAG	CONS_NO	2014-01-01 00:00:00	2014-01-02 00:00:00	2014-01-03 00:00:00	2014-01-04 00:00:00	2014-01-05 00:00:00	2014-01-06 00:00:00	2014-01-07 00:00:00	2014-01-08 00:00:00	...
0	1	0387DD8A07E07FDA6271170F86AD9151	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
1	1	4B75AC4F2D8434CFF62DB64D0BB43103	0.108235	0.210498	0.260883	0.123910	0.134734	0.199674	0.176535	0.137346	...
2	1	B32AC8CC6D5D805AC053557AB05F5343	0.077443	0.150614	0.186665	0.088659	0.096404	0.142869	0.126313	0.098273	...
3	1	EDFC78B07BA2908B3395C4EB2304665E	0.111323	0.216503	0.268326	0.127445	0.138577	0.205371	0.181571	0.141264	...
4	1	6BCFD78138BC72A9BA1BFB0B79382192	0.141255	0.269839	0.339762	0.168473	0.179267	0.260453	0.228541	0.188652	...
5	1	34C1954AA3703C4F8BD8EAEA7C4B7B83	0.015302	0.015302	0.034777	0.037559	0.029213	0.027822	0.019475	0.047297	...
6	1	768309B0EB11FD436CEE5ABFB84F4C0C	0.081576	0.103986	0.067233	0.116537	0.066336	0.084265	0.076197	0.108469	...
7	1	D0A186208CE83FBCCF730857C9A75B6F	0.134217	0.102154	0.103054	0.123304	0.119254	0.104179	0.101929	0.102266	...
8	1	516954F5FF177CE314656D727FCC66A5	0.076888	0.055259	0.058678	0.067399	0.068795	0.058050	0.057283	0.054980	...
9	1	C7952FEE130E744EC8CAA7490F72D3F4	0.137638	0.103156	0.106375	0.120914	0.120857	0.110857	0.103156	0.100685	...
10	1	E34D1B467A365835A4A8AAD9395D49AA	0.191994	0.157062	0.152534	0.169483	0.161461	0.176210	0.145807	0.147489	...
11	1	9F61C13255D1183B179753742A24BFAD	0.069393	0.076659	0.061400	0.062490	0.047594	0.096278	0.055587	0.063943	...

Figure 4.18: Smart Grid dataset after preprocessing.

4.4.2 Data Splitting

Following the pretreatment of the data that we use indicated in the previous subsection, we should split our dataset.

Splitting data is a crucial part of data science. Splitting refers to the process of dividing data into two or more parts. Our dataset is split into two-part, the first portion of our dataset is used to train the model with 80% of the data, and the second part is utilized to evaluate or test the data with 20%.



```
# Splitting the dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y['FLAG'], test_size=0.2, random_state=44)
```

Figure 4.19: Split the data.

Arguments:

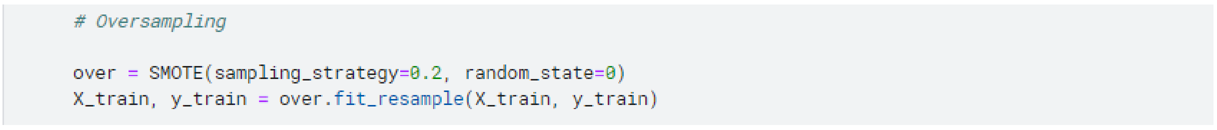
- **FLAG:** The name of our data's class.
- **test_size:** Percentage of the test size when compared to the entire dataset.
- **random_state:** The random state is the lot number of the set created randomly. We may use this lot number to order the same set again at any time. If we don't provide random state in the code, a new random value is created each time we run the code, and the train and test datasets will have different values each time.

	Training	**Test**	**Total**
Normal Consumer	29316	7361	36677
Fraud Consumer	2888	691	3579
Total	32204	8052	40256

Table 4.1: Dataset structure.

This table 4.1 shows the structure of the data that will be used to train and test fraud detection models.

To overcome the imbalanced learning, we choose to use oversampling of the minority class for a better outcome. Data oversampling is a technique for generating data that closely reflects the underlying distribution of real data. To balance out our dataset, we used an oversampling approach called Synthetic Minority Over-Sampling Technique or SMOTE.



```
# Oversampling

over = SMOTE(sampling_strategy=0.2, random_state=0)
X_train, y_train = over.fit_resample(X_train, y_train)
```

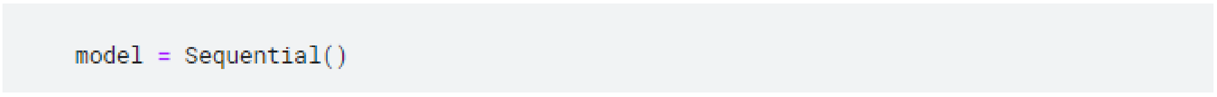
Figure 4.20: Oversampling the train data.

The input data is now available for our CNN models after loading and preparing it.

4.4.3 Creating the Models

In this section, we will utilize 1D CNNs and 2D CNNs to explain how we used deep learning to identify fraudulent clients.

First and foremost, we will utilize the Sequential API model, which is a method of building deep learning models that allows us to build them layer by layer in a sequential way.



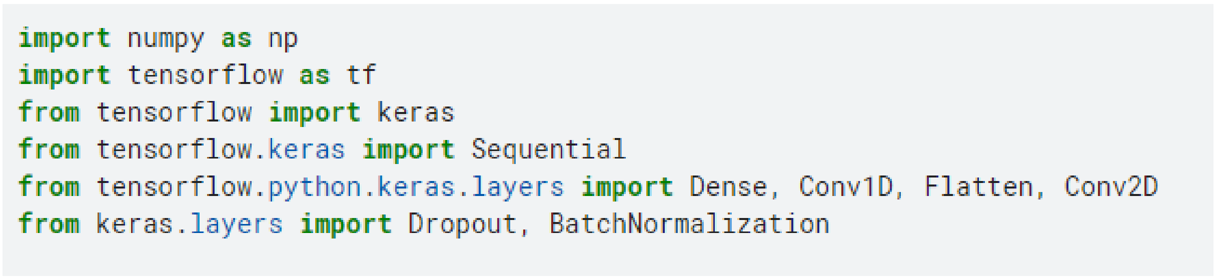
```
model = Sequential()
```

Figure 4.21: Sequential model.

4.4.3.1 1D CNNs model

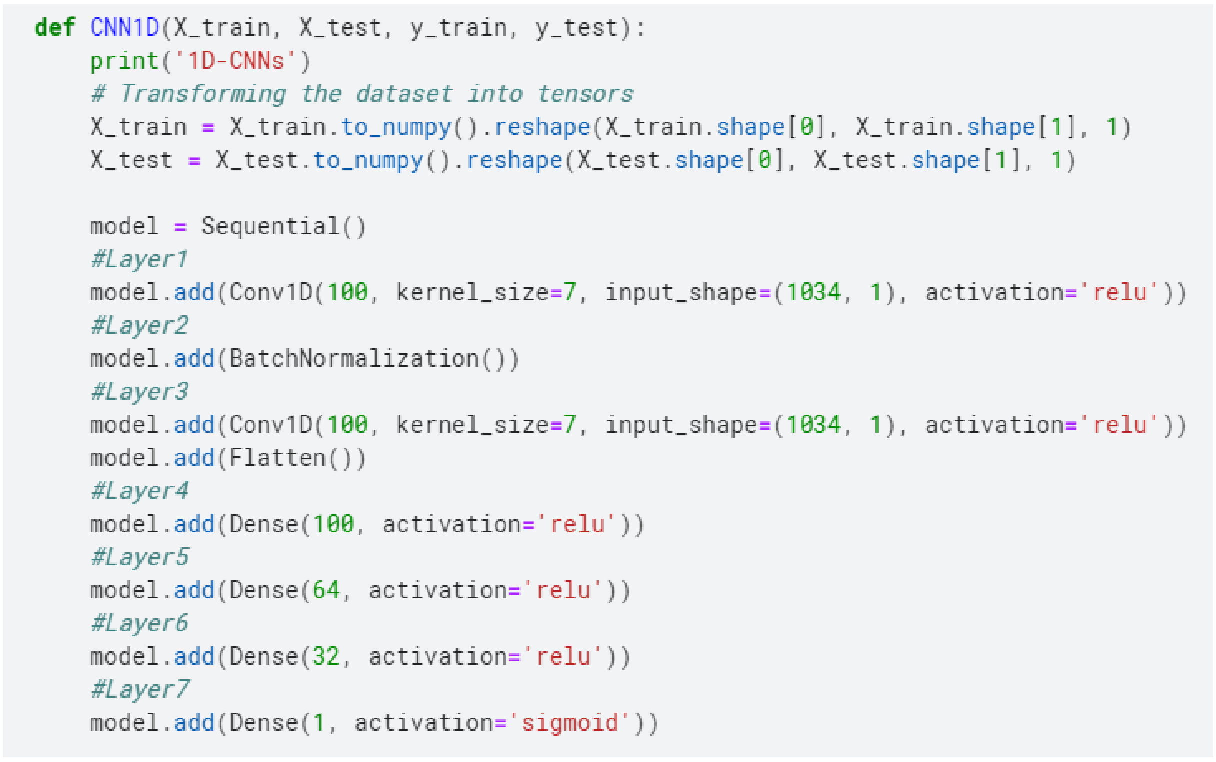
The first CNN experiment architecture is a 1D CNN, which is made up of two Convolution layers, one BatchNormalization layer, one Fully connected tensor, and three Dense layers.

Firstly, we must convert our data into tensors using the NumPy library's reshape () function, which is used to modify the shape of an array without affecting its original data and depicts tensors as n-dimensional arrays of a given use case. In addition, we import the libraries that are utilized in this model.



```
import numpy as np
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import Sequential
from tensorflow.python.keras.layers import Dense, Conv1D, Flatten, Conv2D
from keras.layers import Dropout, BatchNormalization
```

Figure 4.22: Importation of libraries.



```python
def CNN1D(X_train, X_test, y_train, y_test):
    print('1D-CNNs')
    # Transforming the dataset into tensors
    X_train = X_train.to_numpy().reshape(X_train.shape[0], X_train.shape[1], 1)
    X_test = X_test.to_numpy().reshape(X_test.shape[0], X_test.shape[1], 1)

    model = Sequential()
    #Layer1
    model.add(Conv1D(100, kernel_size=7, input_shape=(1034, 1), activation='relu'))
    #Layer2
    model.add(BatchNormalization())
    #Layer3
    model.add(Conv1D(100, kernel_size=7, input_shape=(1034, 1), activation='relu'))
    model.add(Flatten())
    #Layer4
    model.add(Dense(100, activation='relu'))
    #Layer5
    model.add(Dense(64, activation='relu'))
    #Layer6
    model.add(Dense(32, activation='relu'))
    #Layer7
    model.add(Dense(1, activation='sigmoid'))
```

Figure 4.23: 1D-CNNs implementation code.

Layers:

- **Conv1D:** 1D convolution layer, 100 is the number of filters, 7 is the kernel size that specifies the length of the 1D convolution window, input shape is the shape of the input data and activation is the activation function we employ.

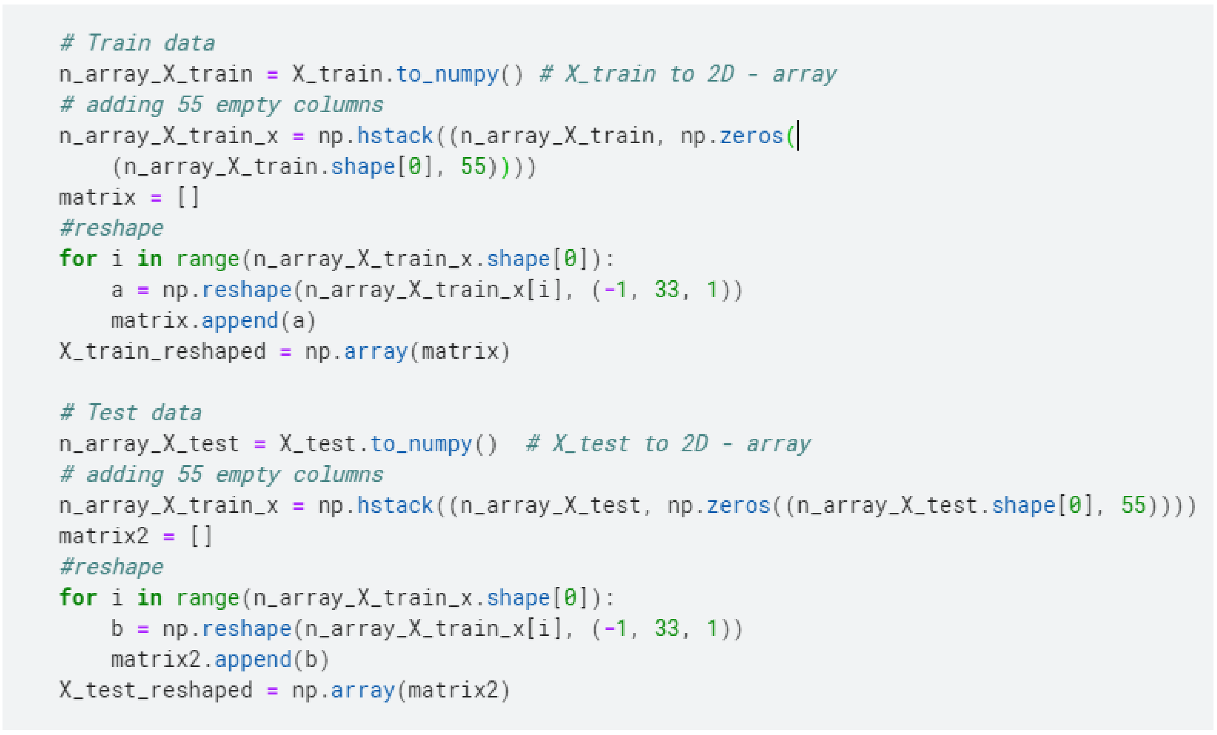
- **BatchNormalization:** Layer that normalizes and scales inputs or activations to improve the speed, performance, and stability of neural networks.

- **Flatten:** In Keras, a Flatten layer reshapes the tensor to have the same shape as the number of elements it contains.

- **Dense:** The dense layer, which is deeply connected to the preceding layer, works to change the output dimension by conducting matrix-vector multiplication.

4.4.3.2 2D CNNs model

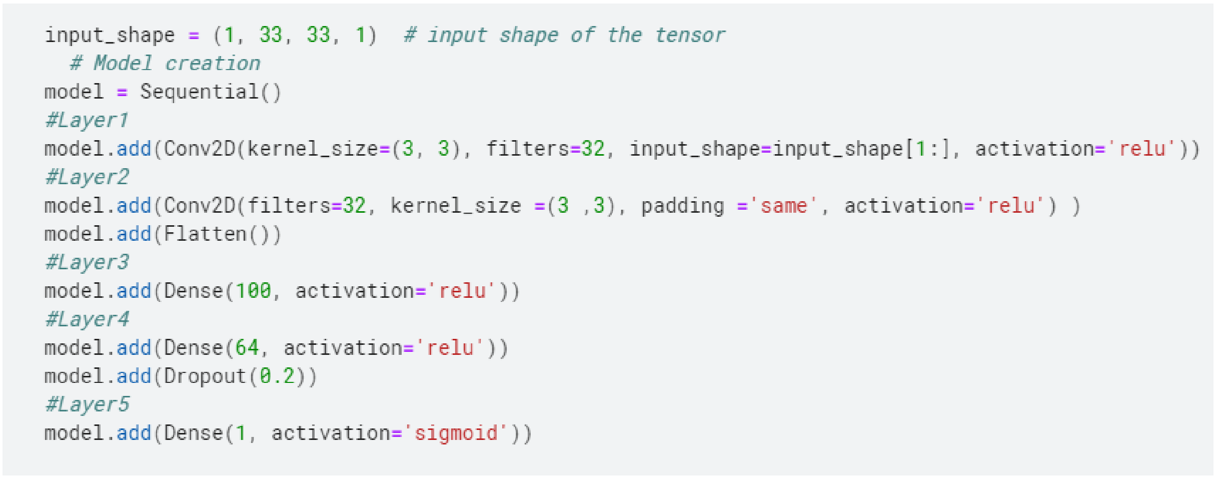
The second CNN experiment model is a 2D CNN, in which we transform our 1D data into 2D data to obtain a 33x33 matrix.
In order to make the number of columns an exact multiple of 33, we add 55 empty columns. Then, using the NumPy library's reshape () function, we turn our 2D data into tensors.



```
# Train data
n_array_X_train = X_train.to_numpy() # X_train to 2D - array
# adding 55 empty columns
n_array_X_train_x = np.hstack((n_array_X_train, np.zeros(
    (n_array_X_train.shape[0], 55))))
matrix = []
#reshape
for i in range(n_array_X_train_x.shape[0]):
    a = np.reshape(n_array_X_train_x[i], (-1, 33, 1))
    matrix.append(a)
X_train_reshaped = np.array(matrix)

# Test data
n_array_X_test = X_test.to_numpy()  # X_test to 2D - array
# adding 55 empty columns
n_array_X_train_x = np.hstack((n_array_X_test, np.zeros((n_array_X_test.shape[0], 55))))
matrix2 = []
#reshape
for i in range(n_array_X_train_x.shape[0]):
    b = np.reshape(n_array_X_train_x[i], (-1, 33, 1))
    matrix2.append(b)
X_test_reshaped = np.array(matrix2)
```

Figure 4.24: Transforming 1D data into 2D data.

Our 2D CNNs model is made up of two Convolution layers, one Fully connected tensor, and three Dense layers.



```
input_shape = (1, 33, 33, 1)  # input shape of the tensor
  # Model creation
model = Sequential()
#Layer1
model.add(Conv2D(kernel_size=(3, 3), filters=32, input_shape=input_shape[1:], activation='relu'))
#Layer2
model.add(Conv2D(filters=32, kernel_size =(3 ,3), padding ='same', activation='relu') )
model.add(Flatten())
#Layer3
model.add(Dense(100, activation='relu'))
#Layer4
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.2))
#Layer5
model.add(Dense(1, activation='sigmoid'))
```

Figure 4.25: 2D-CNNs implementation code.

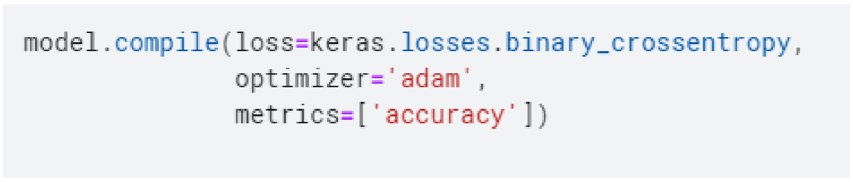
2D-CNNs arguments:

- **input shape:** Is the shape of the input data (batch, dim 1, dim 2, channels).

- **filters:** Integer, the dimensionality of the output space that represents the number of output filters in the convolution.

- **kernel_size:** The height and width of the 2D convolution window, as an integer or a tuple/list of two integers.

Note: We utilized this compiled model in both of our experiments.



```
model.compile(loss=keras.losses.binary_crossentropy,
              optimizer='adam',
              metrics=['accuracy'])
```

Figure 4.26: Compiling Model.

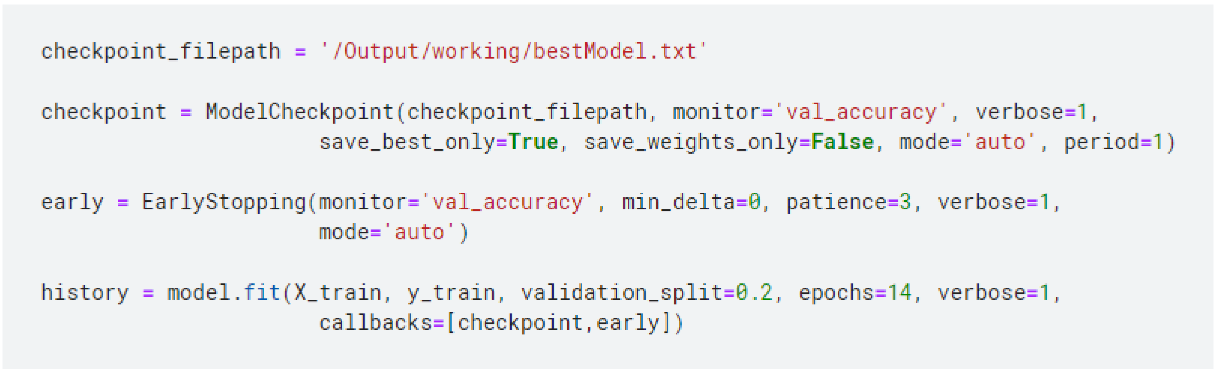
Model compile arguments:

- **loss:** Binary Crossentropy is employed as the loss function when there is simply a classification problem between two categories.
- **optimizer:** We employed Adam optimization, which is a stochastic gradient descent approach based on adaptive first-order and second-order moment estimation.
- **metrics:** List of metrics that the model will evaluate during training and testing.

Once our model is complete, we can use the summary() function to view its contents.

4.4.4 Training the Models

To speed up the training process, we have used a Kaggle GPU, which is provided for 33h for each week. The number of epochs was set to 14. Then, depending on the structure, we utilized ModelCheckpoint and EarlyStopping with various arguments. We will go through these functions and the significance of each of their arguments in detail.



```
checkpoint_filepath = '/Output/working/bestModel.txt'

checkpoint = ModelCheckpoint(checkpoint_filepath, monitor='val_accuracy', verbose=1,
                             save_best_only=True, save_weights_only=False, mode='auto', period=1)

early = EarlyStopping(monitor='val_accuracy', min_delta=0, patience=3, verbose=1,
                             mode='auto')

history = model.fit(X_train, y_train, validation_split=0.2, epochs=14, verbose=1,
                             callbacks=[checkpoint,early])
```

Figure 4.27: Training code.

ModelCheckpoint: The model is saved using a model checkpoint using model.fit(). The monitor parameter allows us to set a metric that we need to keep an eye on. We only save the model in our scenario when the validation accuracy is at its maximum.

ModelCheckpoint arguments:

- **filepath:** the path where the model file should be saved.
- **monitor:** The name of the metric to monitor.
- **verbose:** Verbose mode 1 displays messages when the callback takes an action.
- **save_best_only=True:** The model is only saved when it is deemed the "best".
- **save_weights_only:** If True, just the weights of the model will be saved. otherwise, the entire model will be saved.

EarlyStopping: The EarlyStopping approach is used to cease training when a monitored measure stops improving, its main goal is to minimize overfitting.

EarlyStoppingt arguments:

- **monitor:** The name of the metric to be monitored.
- **min_delta:** There must be a minimum change in the monitored quantity to qualify as an improvement, which means that no improvement if the absolute change is smaller than the min delta.
- **patience:** Training will be stopped after a certain number of epochs with no improvement.

Our model is trained using a single approach termed fit, which requires only a few arguments.

Model fit arguments:

- **X_train:** List of the training data.
- **y_train:** List of the target data.
- **validation_data:** Data on which to evaluate the loss and any model metrics at the end of each epoch.
- **epochs:** The number of epochs to train the model.

In the next section, we will go through the results of our training model, as well as visualize our results using the history variable and the Matpolib library.

4.5 Experiments and Results

This section presents the findings of the models used to identify consumer fraud in the smart grid section.

The experimental findings acquired after using 1D-CNNs are shown in the first half of the section, as well as an analysis of both accuracy and loss obtained. In the second part of the section, we described and studied the results of the 2D-CNNs model.

Finally, a brief summary of both outcomes is provided, along with a comparison of the models.

4.5.1 1D-CNNs model

For the first experiment, we used 1D-CNNs model and received these results after more than 3 minutes of training with Kaggle GPU. We evaluated the model using training accuracy, validation accuracy, training loss, and validation loss.

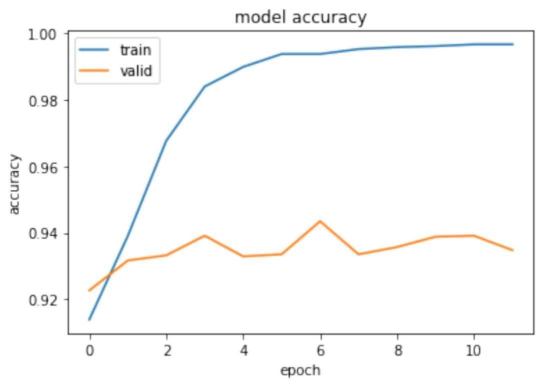


Figure 4.28: The model accuracy obtained by the 1D-CNNs.

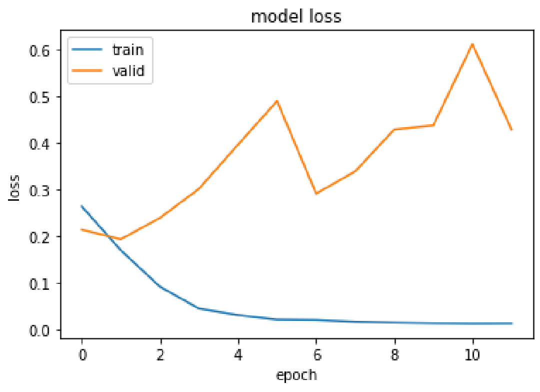
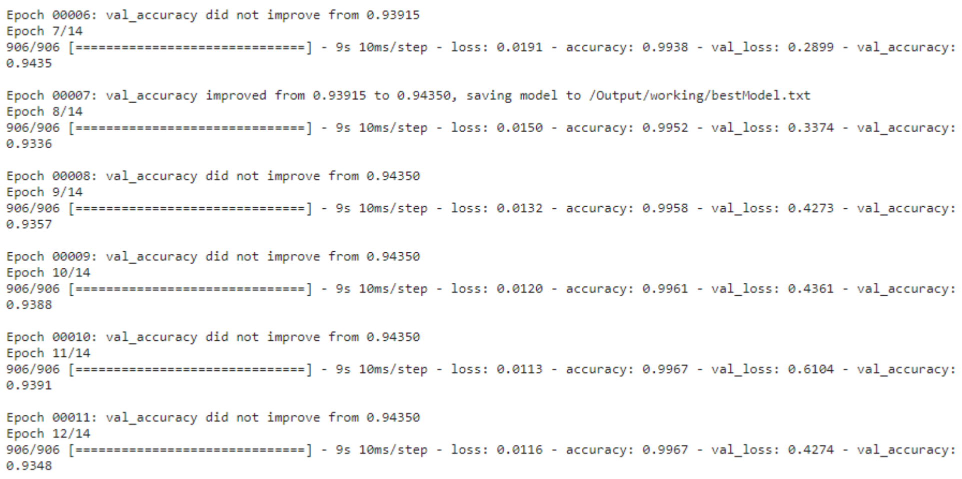


Figure 4.29: The model loss obtained by the 1D-CNNs.

The blue line depicts how well the model learns by each epoch, while the orange line depicts the learning curve calculated from a hold-out validation dataset, which indicates how well the model generalises. In this case, the model is performing well, with a training accuracy of 99.38% and a validation accuracy of 94.35%.

The best model was preserved in epoch 7, while the earliest stoppage occurred in epoch 12. As it will show in figure 4.30.



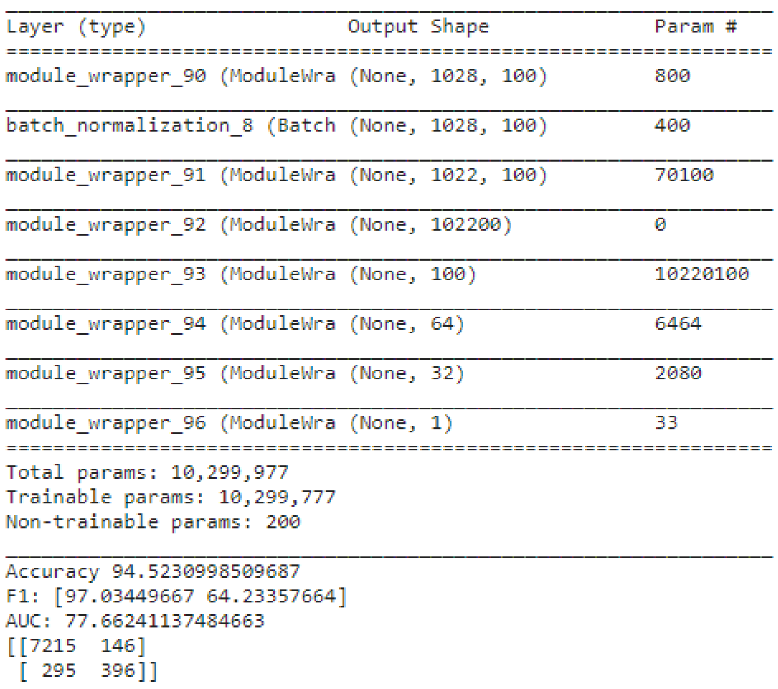
```
Epoch 00006: val_accuracy did not improve from 0.93915
Epoch 7/14
906/906 [==============================] - 9s 10ms/step - loss: 0.0191 - accuracy: 0.9938 - val_loss: 0.2899 - val_accuracy:
0.9435

Epoch 00007: val_accuracy improved from 0.93915 to 0.94350, saving model to /Output/working/bestModel.txt
Epoch 8/14
906/906 [==============================] - 9s 10ms/step - loss: 0.0150 - accuracy: 0.9952 - val_loss: 0.3374 - val_accuracy:
0.9336

Epoch 00008: val_accuracy did not improve from 0.94350
Epoch 9/14
906/906 [==============================] - 9s 10ms/step - loss: 0.0132 - accuracy: 0.9958 - val_loss: 0.4273 - val_accuracy:
0.9357

Epoch 00009: val_accuracy did not improve from 0.94350
Epoch 10/14
906/906 [==============================] - 9s 10ms/step - loss: 0.0120 - accuracy: 0.9961 - val_loss: 0.4361 - val_accuracy:
0.9388

Epoch 00010: val_accuracy did not improve from 0.94350
Epoch 11/14
906/906 [==============================] - 9s 10ms/step - loss: 0.0113 - accuracy: 0.9967 - val_loss: 0.6104 - val_accuracy:
0.9391

Epoch 00011: val_accuracy did not improve from 0.94350
Epoch 12/14
906/906 [==============================] - 9s 10ms/step - loss: 0.0116 - accuracy: 0.9967 - val_loss: 0.4274 - val_accuracy:
0.9348
```

Figure 4.30: Training process for 1D-CNNs.

	Accuracy	**Validation accuracy**	**Loss**	**Validation Loss**
1D-CNNs Model	99.38%	94.35%	1.91%	28.99%

Table 4.2: Table of results of 1D-CNNs.

We found the following metrics after evaluating our model on the second subset (test data).



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
module_wrapper_90 (ModuleWra (None, 1028, 100)         800
_________________________________________________________________
batch_normalization_8 (Batch (None, 1028, 100)         400
_________________________________________________________________
module_wrapper_91 (ModuleWra (None, 1022, 100)         70100
_________________________________________________________________
module_wrapper_92 (ModuleWra (None, 102200)            0
_________________________________________________________________
module_wrapper_93 (ModuleWra (None, 100)               10220100
_________________________________________________________________
module_wrapper_94 (ModuleWra (None, 64)                6464
_________________________________________________________________
module_wrapper_95 (ModuleWra (None, 32)                2080
_________________________________________________________________
module_wrapper_96 (ModuleWra (None, 1)                 33
=================================================================
Total params: 10,299,977
Trainable params: 10,299,777
Non-trainable params: 200
_________________________________________________________________
Accuracy 94.5230998509687
F1: [97.03449667 64.23357664]
AUC: 77.66241137484663
[[7215  146]
 [ 295  396]]
```

Figure 4.31: Testing results for 1D-CNNs.

For the 1D-CNNs model, we found that accuracy was 94.52%, and AUC(Area under the curve) was 77.66%. The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve. AUC is a better measure of classifier performance than accuracy.

Concerning the confusion matrix:

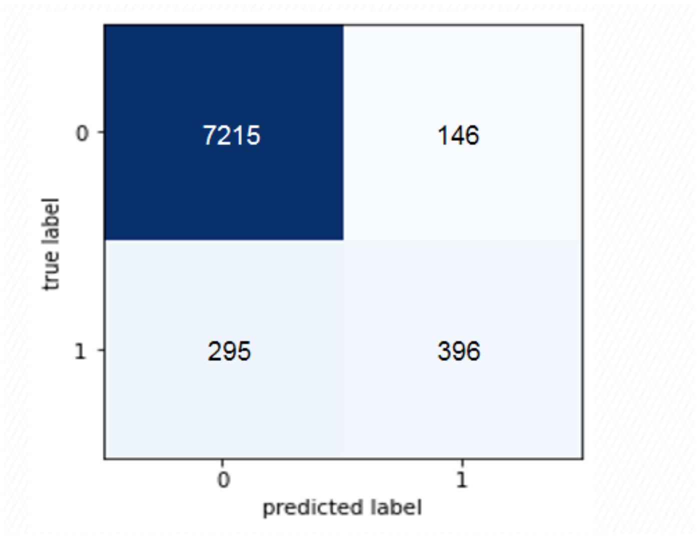
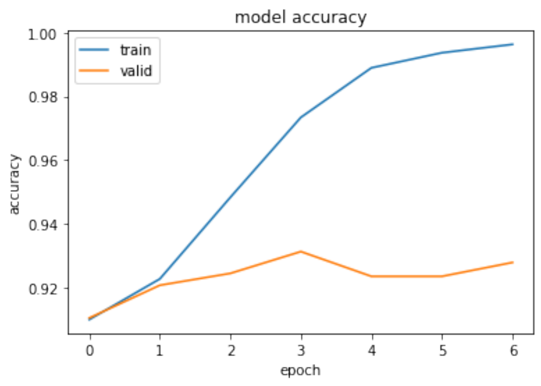


Figure 4.32: 1D-CNNs confusion matrix.

- The model could predict 7215 data right out of 7361 from the No fraud consumers.
- The model could predict 396 data right out of 691 from the fraud consumers.

4.5.2 2D-CNNs model

For the second experiment, we used 2D-CNNs model and received these results after 42 second of training.



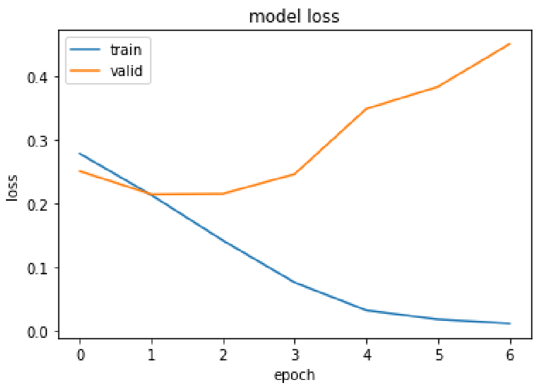
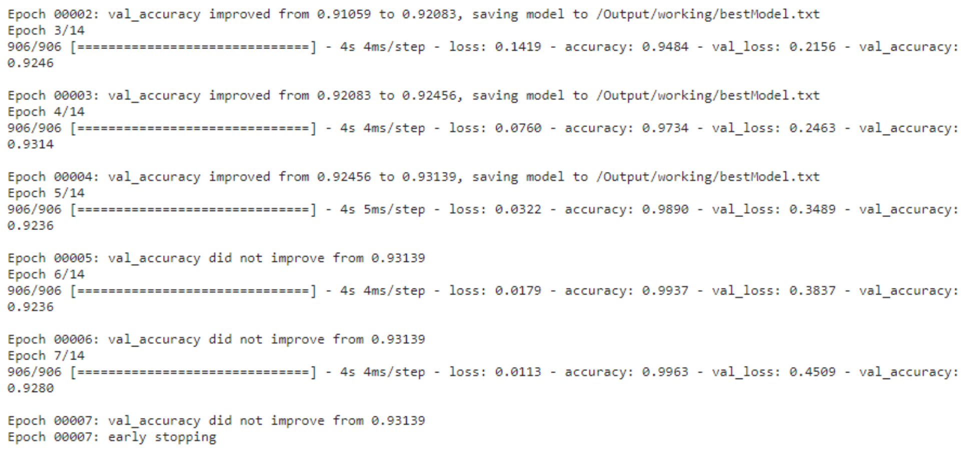


Figure 4.33: The model accuracy obtained by the 2D-CNNs.

Figure 4.34: The model loss obtained by the 2D-CNNs.

In this case, the 2D-CNNs model performs worse than the 1D-CNNs model, with a training accuracy of 97.34% and a validation accuracy of 93.14%. Figure 4.35 will show when the best model is saved.



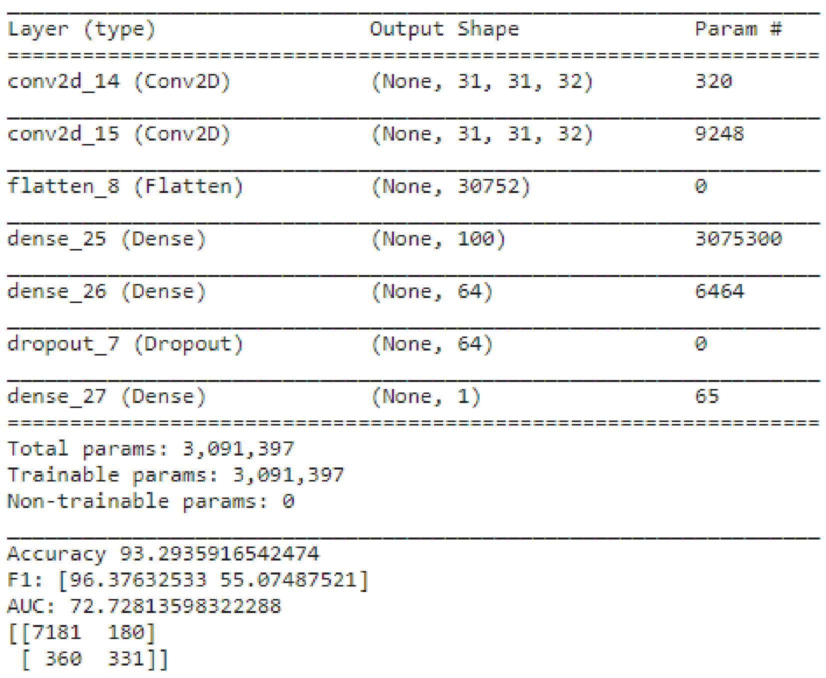
```
Epoch 00002: val_accuracy improved from 0.91059 to 0.92083, saving model to /Output/working/bestModel.txt
Epoch 3/14
906/906 [==============================] - 4s 4ms/step - loss: 0.1419 - accuracy: 0.9484 - val_loss: 0.2156 - val_accuracy:
0.9246

Epoch 00003: val_accuracy improved from 0.92083 to 0.92456, saving model to /Output/working/bestModel.txt
Epoch 4/14
906/906 [==============================] - 4s 4ms/step - loss: 0.0760 - accuracy: 0.9734 - val_loss: 0.2463 - val_accuracy:
0.9314

Epoch 00004: val_accuracy improved from 0.92456 to 0.93139, saving model to /Output/working/bestModel.txt
Epoch 5/14
906/906 [==============================] - 4s 5ms/step - loss: 0.0322 - accuracy: 0.9890 - val_loss: 0.3489 - val_accuracy:
0.9236

Epoch 00005: val_accuracy did not improve from 0.93139
Epoch 6/14
906/906 [==============================] - 4s 4ms/step - loss: 0.0179 - accuracy: 0.9937 - val_loss: 0.3837 - val_accuracy:
0.9236

Epoch 00006: val_accuracy did not improve from 0.93139
Epoch 7/14
906/906 [==============================] - 4s 4ms/step - loss: 0.0113 - accuracy: 0.9963 - val_loss: 0.4509 - val_accuracy:
0.9280

Epoch 00007: val_accuracy did not improve from 0.93139
Epoch 00007: early stopping
```

Figure 4.35: Training process for 2D-CNNs.

	Accuracy	**Validation accuracy**	**Loss**	**Validation Loss**
2D-CNNs Model	97.34%	93.35%	7.60%	24.63%

Table 4.3: Table of results of 2D-CNNs.

We found the following metrics after evaluating our model on the second subset (test data).



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_14 (Conv2D)           (None, 31, 31, 32)        320
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 31, 31, 32)        9248
_________________________________________________________________
flatten_8 (Flatten)          (None, 30752)             0
_________________________________________________________________
dense_25 (Dense)             (None, 100)               3075300
_________________________________________________________________
dense_26 (Dense)             (None, 64)                6464
_________________________________________________________________
dropout_7 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_27 (Dense)             (None, 1)                 65
=================================================================
Total params: 3,091,397
Trainable params: 3,091,397
Non-trainable params: 0

_________________________________________________________________
Accuracy 93.2935916542474
F1: [96.37632533 55.07487521]
AUC: 72.72813598322288
[[7181  180]
 [ 360  331]]
```

Figure 4.36: Testing results for 2D-CNNs.

For the 2D-CNNs model, we found that accuracy was 93.29%, and AUC was 72.72%.

As for the confusion matrix:

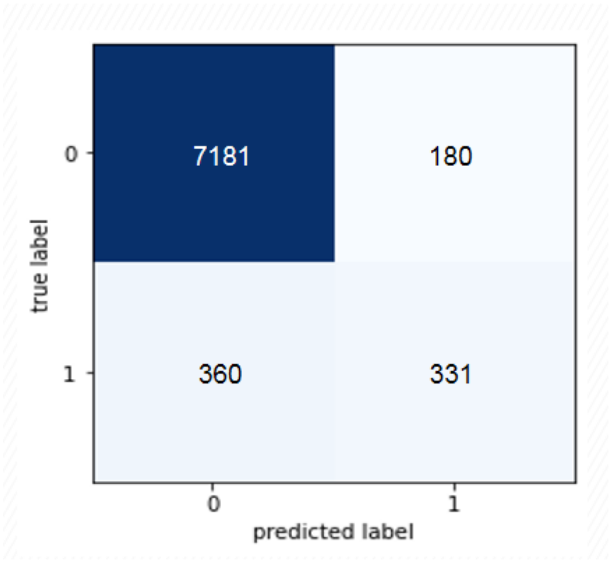


Figure 4.37: 2D-CNNs confusion matrix.

- The model could predict 7181 data right out of 7361 from the No fraud consumers.
- The model could predict 331 data right out of 691 from the fraud consumers.

4.5.3 Comparisons of our Experiments

Following our experiments with the two models, we find that 1D-CNNs are the best model for our smart grid data.
The comparisons of our experiments are shown in Table 4.4.

	Accuracy	**AUC**	**Precision**	**Recall**	**Time**
1D-CNNs	**94.52%**	**77.66%**	**73.06%**	**57.30%**	**3.24 min**
2D-CNNs	93.29%	72.72%	64.77%	47.90%	42s

Table 4.4: Comparisons of our models results.

4.6 Comparisons our Work and the Previous Works

In this section, we will compare the various achieved outcomes from our proposed deep convolutional networks with previous studies. To make the comparison, we choose accuracy.

	ANN	**1D-CNNs**	**2D-CNNs**	SVM	ANN-KDD
Accuracy	84.37%	**94.52%**	**93.29%**	86.43%	87.17%

Table 4.5: Accuracies comparisons of our experiments and previous works.

Table 4.5 shows that when compared to standard Machine learning techniques, convolutional neural networks provide higher outcomes in terms of accuracy. The findings from CNN models are good, and 1D-CNNs outperform them all.

4.7 Conclusion

This chapter included the theme primary findings, the implementation and experiments were discussed and explained, in addition to tables with annotations, and the results were presented with plots and the confusion matrix for the test phase, and comparisons of the acquired results.

The acquired findings show very strong performance, encouraging us to strengthen our model design.

Conclusion and future work

This study intends to establish and provide a deep learning-based classification model for detecting fraudulent behavior in power usage, where State Grid Corporation of China (SGCC) is considered as the case study of our theme. It's difficult for energy providers to examine the No Technical Losses (NTLs) because most device operators disregard them and do not collect any data. To reduce the NTLs, we conducted models using deep convolutional neural networks (CNNs).

As it turned out later, our CNNs models outperformed all of the earlier studies that we discussed. In particular, 1D-CNNs produced more accurate results than 2D-CNNs. As a result, Convolution Neural Networks have a variety of interesting implications, where CNNs may be used to detect abnormalities in a wide range of situations.

Generally, our models produce good classification results. however, due to hardware restrictions, we were unable to train our models for extended epochs, and we did not test our models on other datasets to determine whether they were accurate as well as they did on our dataset.

For the future, this theme attempts to address the way for further research into the model's improvement, the use of another dataset, as well as a variety of deep learning approaches, and other preprocessing methods.

Bibliography

[1] Shuan Li et al. “Electricity theft detection in power grids with deep learning and random forests”. In: *Journal of Electrical and Computer Engineering* 2019 (2019).

[2] *fraud.* URL: `https://dictionary.cambridge.org/fr/dictionnaire/anglais/fraud`.

[3] *fraud.* URL: `https://ahdictionary.com/word/search.html?q=fraud`.

[4] Michael D Akers and Jodi L Gissel. “What is fraud and who is responsible?” In: *Journal of Forensic Accounting* (2006), pp. 3–4.

[5] Véronique Van Vlasselaer et al. “Gotcha! Network-based fraud detection for social security fraud”. In: *Management Science* 63.9 (2017), pp. 3090–3110.

[6] Madan Lal Bhasin. “Menace of frauds in the Indian banking industry: an empirical study”. In: *Australian Journal of Business and Management Research* 4.12 (2015).

[7] Alan Doig. *Fraud.* Willan, 2013.

[8] H Lookman Sithic and T Balasubramanian. “Survey of insurance fraud detection using data mining techniques”. In: *arXiv preprint arXiv:1309.0806* (2013).

[9] *fraud detection.* URL: `https://www.techtarget.com/searchsecurity/definition/fraud-detection`.

[10] Bernat Coma-Puig et al. “Fraud detection in energy consumption: A supervised approach”. In: *2016 IEEE international conference on data science and advanced analytics (DSAA).* IEEE. 2016, pp. 120–129.

[11] Zibin Zheng et al. “Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids”. In: *IEEE Transactions on Industrial Informatics* 14.4 (2017), pp. 1606–1615.

[12] Muhammad Salman Saeed et al. “Detection of non-technical losses in power utilities—A comprehensive systematic review”. In: *Energies* 13.18 (2020), p. 4727.

[13] Md Hasan et al. “Electricity theft detection in smart grid systems: A CNN-LSTM based approach”. In: *Energies* 12.17 (2019), p. 3310.

[14] Yining Yang et al. "A Detection Method for Group Fixed Ratio Electricity Thieves Based on Correlation Analysis of Non-technical Loss". In: *IEEE Access* (2022).

[15] Z Hussain et al. "Analysis of non-technical electrical power losses and their economic impact on Pakistan". In: *Sindh University Research Journal-SURJ (Science Series)* 49.2 (2017).

[16] Zahoor Hussain et al. "Methods and techniques of electricity thieving in pakistan". In: *Journal of Power and Energy Engineering* 4.09 (2016), p. 1.

[17] Ramazan Bayindir et al. "Smart grid technologies and applications". In: *Renewable and Sustainable Energy Reviews* 66 (2016), pp. 499–516.

[18] Yasin Kabalci. "A survey on smart metering and smart grid communication". In: *Renewable and Sustainable Energy Reviews* 57 (2016), pp. 302–318.

[19] Yu Cunjiang, Zhang Huaxun, and Zhao Lei. "Architecture design for smart grid". In: *Energy Procedia* 17 (2012), pp. 1524–1528.

[20] Ilhami Colak. "Introduction to smart grid". In: *2016 International Smart Grid Workshop and Certificate Program (ISGWCP)*. IEEE. 2016, pp. 1–5.

[21] Chengbing Wei. "A conceptual framework for smart grid". In: *2010 Asia-Pacific Power and Energy Engineering Conference*. IEEE. 2010, pp. 1–4.

[22] Wenlin Han and Yang Xiao. "Non-technical loss fraud in advanced metering infrastructure in smart grid". In: *International Conference on Cloud Computing and Security*. Springer. 2016, pp. 163–172.

[23] Vikas Jayasree and Rethnamoney Vijayalakshmi Siva Balan. "A review on data mining in banking sector". In: *American Journal of Applied Sciences* 10.10 (2013), p. 1160.

[24] Richard J Bolton and David J Hand. "Statistical fraud detection: A review". In: *Statistical science* 17.3 (2002), pp. 235–255.

[25] Yufeng Kou et al. "Survey of fraud detection techniques". In: *IEEE international conference on networking, sensing and control, 2004*. Vol. 2. IEEE. 2004, pp. 749–754.

[26] Luisa M Mimmi and Sencer Ecer. "An econometric study of illegal electricity connections in the urban favelas of Belo Horizonte, Brazil". In: *Energy Policy* 38.9 (2010), pp. 5081–5097.

[27] Faisal Jamil. "On the electricity shortage, price and electricity theft nexus". In: *Energy policy* 54 (2013), pp. 267–272.

[28] Tanja Winther. "Electricity theft as a relational issue: A comparative look at Zanzibar, Tanzania, and the Sunderban Islands, India". In: *Energy for Sustainable Development* 16.1 (2012), pp. 111–119.

[29] Çağlar Yurtseven. “The causes of electricity theft: An econometric analysis of the case of Turkey”. In: *Utilities Policy* 37 (2015), pp. 70–78.

[30] Joaquim L Viegas et al. “Solutions for detection of non-technical losses in the electricity grid: A review”. In: *Renewable and Sustainable Energy Reviews* 80 (2017), pp. 1256–1268.

[31] Benjamin Khoo and Ye Cheng. “Using RFID for anti-theft in a Chinese electrical supply company: A cost-benefit analysis”. In: *2011 Wireless Telecommunications Symposium (WTS)*. IEEE. 2011, pp. 1–6.

[32] K Dineshkumar, Prabhu Ramanathan, and Sudha Ramasamy. “Development of ARM processor based electricity theft control system using GSM network”. In: *2015 International Conference on Circuits, Power and Computing Technologies [ICCPCT-2015]*. IEEE. 2015, pp. 1–6.

[33] Suchat Ngamchuen and Chaiyod Pirak. “Smart anti-tampering algorithm design for single phase smart meter applied to AMI systems”. In: *2013 10th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*. IEEE. 2013, pp. 1–6.

[34] Aryadevi Remanidevi Devidas and Maneesha Vinodini Ramesh. “Wireless smart grid design for monitoring and optimizing electric transmission in India”. In: *2010 Fourth International Conference on Sensor Technologies and Applications*. IEEE. 2010, pp. 637–640.

[35] Petr Kadurek et al. “Theft detection and smart metering practices and expectations in the Netherlands”. In: *2010 IEEE PES innovative smart grid technologies conference Europe (ISGT Europe)*. IEEE. 2010, pp. 1–6.

[36] Soma Shekara Sreenadh Reddy Depuru, Lingfeng Wang, and Vijay Devabhaktuni. “A conceptual design using harmonics to reduce pilfering of electricity”. In: *IEEE PES General Meeting*. IEEE. 2010, pp. 1–7.

[37] A Pasdar and S Mirzakuchaki. “A solution to remote detecting of illegal electricity usage based on smart metering”. In: *2007 2nd International Workshop on Soft Computing Applications*. IEEE. 2007, pp. 163–167.

[38] Madalina Mihaela Buzau et al. “Detection of non-technical losses using smart meter data and supervised learning”. In: *IEEE Transactions on Smart Grid* 10.3 (2018), pp. 2661–2670.

[39] Anish Jindal et al. “Decision tree and SVM-based data analytics for theft detection in smart grid”. In: *IEEE Transactions on Industrial Informatics* 12.3 (2016), pp. 1005–1016.

[40] Muhammad Salman Saeed et al. “An efficient boosted C5. 0 decision-tree-based classification approach for detecting non-technical losses in power utilities”. In: *Energies* 13.12 (2020), p. 3242.

[41] M Anas et al. “Minimizing electricity theft using smart meters in AMI”. In: *2012 Seventh International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*. IEEE. 2012, pp. 176–182.

[42] Madalina-Mihaela Buzau et al. “Hybrid deep neural networks for detection of non-technical losses in electricity smart meters”. In: *IEEE Transactions on Power Systems* 35.2 (2019), pp. 1254–1263.

[43] Muhammad Salman Saeed et al. “Fraud detection for metered costumers in power distribution companies using C5. 0 decision tree algorithm”. In: *Journal of Computational and Theoretical Nanoscience* 17.2-3 (2020), pp. 1318–1325.

[44] Christa Cody, Vitaly Ford, and Ambareen Siraj. “Decision tree learning for fraud detection in consumer energy consumption”. In: *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*. IEEE. 2015, pp. 1175–1179.

[45] AH Nizar et al. “A data mining based NTL analysis method”. In: *2007 IEEE power engineering society general meeting*. IEEE. 2007, pp. 1–8.

[46] Jacob Sakhnini, Hadis Karimipour, and Ali Dehghantanha. “Smart grid cyber attacks detection using supervised learning and heuristic feature selection”. In: *2019 IEEE 7th International Conference on Smart Energy Grid Engineering (SEGE)*. IEEE. 2019, pp. 108–112.

[47] Leandro Aparecido Passos Júnior et al. “Unsupervised non-technical losses identification through optimum-path forest”. In: *Electric Power Systems Research* 140 (2016), pp. 413–423.

[48] Kedi Zheng et al. “A novel combined data-driven approach for electricity theft detection”. In: *IEEE Transactions on Industrial Informatics* 15.3 (2018), pp. 1809–1819.

[49] Paria Jokar, Nasim Arianpoo, and Victor CM Leung. “Electricity theft detection in AMI using customers’ consumption patterns”. In: *IEEE Transactions on Smart Grid* 7.1 (2015), pp. 216–226.

[50] Patrick Glauner et al. “The challenge of non-technical loss detection using artificial intelligence: A survey”. In: *arXiv preprint arXiv:1606.00626* (2016).

[51] Juan I Guerrero et al. “Improving knowledge-based systems with statistical techniques, text mining, and neural networks for non-technical loss detection”. In: *Knowledge-Based Systems* 71 (2014), pp. 376–388.

[52] Sook-Chin Yip et al. “Energy theft and defective meters detection in AMI using linear regression”. In: *2017 IEEE International Conference on Environment and Electrical Engineering and 2017 IEEE Industrial and Commercial Power Systems Europe (EEEIC/I&CPS Europe)*. IEEE. 2017, pp. 1–6.

[53] Abdulrahaman Okino Otuoze et al. “Electricity theft detection by sources of threats for smart city planning”. In: *IET Smart Cities* 1.2 (2019), pp. 52–60.

[54] George M Messinis and Nikos D Hatziargyriou. “Review of non-technical loss detection methods”. In: *Electric Power Systems Research* 158 (2018), pp. 250–266.

[55] Ting Liu et al. “A novel method to detect bad data injection attack in smart grid”. In: *2013 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE. 2013, pp. 49–54.

[56] G de O Lucimário, Albérico AP da Silva, and Adiel T de Almeida-Filho. “Allocation of power-quality monitors using the P-median to identify nontechnical losses”. In: *IEEE Transactions on Power Delivery* 31.5 (2016), pp. 2242–2249.

[57] Yonghe Guo, Chee-Wooi Ten, and Panida Jirutitijaroen. “Online data validation for distribution operations against cybertampering”. In: *IEEE Transactions on Power Systems* 29.2 (2013), pp. 550–560.

[58] Jiawei Han, Micheline Kamber, and Jian Pei. “Data Mining: Concepts and Techniques Third Edition [M]”. In: *The Morgan Kaufmann Series in Data Management Systems* 5.4 (2011), pp. 61, 62.

[59] Mariette Awad and Rahul Khanna. *Efficient learning machines: theories, concepts, and applications for engineers and system designers*. Springer nature, 2015.

[60] Myeongsu Kang and Noel Jordan Jameson. “Machine Learning: Fundamentals”. In: *Prognostics and Health Management of Electronics: Fundamentals, Machine Learning, and the Internet of Things* (2018), pp. 85–109.

[61] Kenji Suzuki. *Artificial neural networks: methodological advances and biomedical applications*. BoD–Books on Demand, 2011.

[62] Anders Krogh. “What are artificial neural networks?” In: *Nature biotechnology* 26.2 (2008), p. 1.

[63] Ajith Abraham. “Artificial neural networks”. In: *Handbook of measuring system design* (2005).

[64] OS Eluyode and Dipo Theophilus Akomolafe. “Comparative study of biological and artificial neural networks”. In: *European Journal of Applied Engineering and Scientific Research* 2.1 (2013), pp. 7–9.

[65] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[66] Daniele Ravı et al. “Deep learning for health informatics”. In: *IEEE journal of biomedical and health informatics* 21.1 (2016), p. 1.

[67] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *nature* 521.7553 (2015), pp. 436–444.

[68] Ajeet Ram Pathak, Manjusha Pandey, and Siddharth Rautaray. “Deep learning approaches for detecting objects from images: a review”. In: *Progress in Computing, Analytics and Networking* (2018), pp. 491–499.

[69] Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. “Listening while speaking: Speech chain by deep learning”. In: *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE. 2017, pp. 301–308.

[70] Frank Seide, Gang Li, and Dong Yu. “Conversational speech transcription using context-dependent deep neural networks”. In: *Twelfth annual conference of the international speech communication association*. 2011.

[71] Guansong Pang et al. “Deep learning for anomaly detection: A review”. In: *ACM Computing Surveys (CSUR)* 54.2 (2021), pp. 1–38.

[72] Abubakr H Ombabi et al. “Deep learning framework based on Word2Vec and CNNfor users interests classification”. In: *2017 Sudan conference on computer science and information technology (SCCSIT)*. IEEE. 2017, pp. 1–7.

[73] Larry R Medsker and LC Jain. “Recurrent neural networks”. In: *Design and Applications* 5 (2001), pp. 64–67.

[74] Jonghong Kim et al. “Convolutional neural network with biologically inspired retinal structure”. In: *Procedia Computer Science* 88 (2016), p. 2.

[75] Yoshua Bengio, Ian Goodfellow, and Aaron Courville. *Deep learning*. Vol. 1. MIT press Cambridge, MA, USA, 2017.

[76] Laith Alzubaidi et al. “Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions”. In: *Journal of big Data* 8.1 (2021), pp. 1–74.

[77] Eiman Kanjo, Eman MG Younis, and Chee Siang Ang. “Deep learning analysis of mobile physiological, environmental and location sensor data for emotion detection”. In: *Information Fusion* 49 (2019), p. 12.

[78] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. “Activation functions in neural networks”. In: *towards data science* 6.12 (2017), p. 1.

[79] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. “Understanding of a convolutional neural network”. In: *2017 international conference on engineering and technology (ICET)*. Ieee. 2017, p. 4.

[80] Mahbub Hussain, Jordan J Bird, and Diego R Faria. “A study on cnn transfer learning for image classification”. In: *UK Workshop on computational Intelligence*. Springer. 2018, p. 3.

[81] Serkan Kiranyaz et al. “1D convolutional neural networks and applications: A survey”. In: *Mechanical systems and signal processing* 151 (2021), p. 107398.

[82] Andreas Kamilaris and Francesc X Prenafeta-Boldú. “A review of the use of convolutional neural networks in agriculture”. In: *The Journal of Agricultural Science* 156.3 (2018), p. 15.

[83] Manjunath Jogin et al. “Feature extraction using convolution neural networks (CNN) and deep learning”. In: *2018 3rd IEEE international conference on recent trends in electronics, information & communication technology (RTEICT)*. IEEE. 2018, pp. 2319–2323.

[84] Ping-Rong Chen et al. “DSNet: An efficient CNN for road scene segmentation”. In: *APSIPA Transactions on Signal and Information Processing* 9 (2020).

[85] Rikiya Yamashita et al. “Convolutional neural networks: an overview and application in radiology”. In: *Insights into imaging* 9.4 (2018), pp. 611–629.

[86] Sangyeon Kim and Myungjoo Kang. “Financial series prediction using Attention LSTM”. In: *arXiv preprint arXiv:1902.10877* (2019).

[87] Banee Bandana Das et al. “A spatio-temporal model for EEG-based person identification”. In: *Multimedia Tools and Applications* 78.19 (2019), pp. 28157–28177.

[88] Vitaly Ford, Ambareen Siraj, and William Eberle. “Smart grid energy fraud detection using artificial neural networks”. In: *2014 IEEE symposium on computational intelligence applications in smart grid (CIASG)*. IEEE. 2014, pp. 1–6.

[89] Jawad Nagi et al. “Nontechnical loss detection for metered customers in power utility using support vector machines”. In: *IEEE transactions on Power Delivery* 25.2 (2009), pp. 1162–1171.

[90] Breno C Costa et al. “Fraud detection in electric power distribution networks using an ann-based knowledge-discovery process”. In: *International Journal of Artificial Intelligence & Applications* 4.6 (2013), p. 17.

[91] Anisah H Nizar, Zhao Y Dong, and JH Zhao. “Load profiling and data mining techniques in electricity deregulated market”. In: *2006 IEEE Power Engineering Society General Meeting*. IEEE. 2006, 7–pp.

[92] *ElectricityTheftDetection*. URL: `https://github.com/henryRDlab/ElectricityTheftDetectio`

[93] Wes McKinney et al. “Data structures for statistical computing in python”. In: *Proceedings of the 9th Python in Science Conference*. Vol. 445. 1. Austin, TX. 2010, pp. 51–56.