A Framework for the Analysis of Discrete Irregular Patterned Sequential Datasets

A thesis submitted in fulfillment of the requirements for the Doctor of Philosophy in Computer Science in the Department of Computer Science, Faculty of Natural Sciences.

Submitted on October 18, 2023

by

Kudakwashe Dandajena
Student Number : 3986658

Supervisor : Dr Mehrdad Ghaziasgar

Co-Supervisor : Professor Isabella M. Venter

Mentor : Reg Dodds

Abstract

Trial-and-error is often used for developing new deep learning models. There is no systematic procedure for improving existing models for predicting irregular sequential time-series. This research developed a systematic framework to improve state-of-the-art deep learning models for financial time-series prediction. The framework was used to create an enhanced model. A design science research methodology was used to design this framework with customised multidimensional evaluation criteria and metrics. The model was applied to predict currency exchange rates more accurately than existing state-of-the-art models found in the literature. The main contribution of this work is a framework that provides a procedure for improving deep learning models. As a proof of its usefulness, the framework was applied to develop an improved model.

Keywords

Discrete, sequential, time-series, machine, deep, learning, model and prediction.

Declaration of Authorship

I Kudakwashe Dandajena, declare that the thesis which is hereby submitted to the Department of Computer Science, Faculty of Natural Sciences of the University of the Western Cape is my independent work and has not been handed in before for a qualification at or in another University or Faculty or Department or school. I further declare that all sources cited or quoted are indicated and acknowledged by a complete and comprehensive list of references. I further cede copyright of the thesis to the University of the Western Cape.

[illegible]

Student's signature......................................: Date: October 18, 2023

Student : Kudakwashe Dandajena

Dedication

Without the help of these beloved people, this thesis would not have been completed, and I owe them a debt of gratitude that's why I heartily and proudly dedicated this work to them. I am grateful for the support, strength, courage, patience, wisdom, time, and guidance provided by my supervisors Prof. Isabella M. Venter and Dr. Mehrdad Ghaziasgar, and my mentor Reg Dodds in realizing this work, which has been tough, painful, and exciting in some parts. The faculty and staff of the University of the Western Cape supported me in unique ways. Dr. Panashe Chiurunge and Dr. Eng. Willie D. Ganda encouraged me all the way and ensured that I give everything I have to finish what I have started.

My mother, mother-in-law, father-in-law, siblings and grandparents for being my source of inspiration. Even though my late father is no longer with us, his virtues of perseverance, hard work and respect continue to guide me in my life.

A special loving thanks to my dear wife Rutendo Shiloh for your encouragement, prayers and support especially in balancing competing demands of study, work and family. My children Kudakwashe Ethan, Kupakwashe Kayla and Kunashe Seth have been affected in every way possible by this quest. Finally, God the Divine who continues to make the impossible possible.

Thank you all. My love for you all can never be quantified. God bless you richly.

Acknowledgments

This work received funding from the Research Committee of the University of the Western Cape and Dr Panashe Chiurunge.

Contents

List of Figures

List of Tables

Abbreviations

1D	One-dimensional
3D	Three-dimensional
4IR	Fourth industrial revolution
5G	Fifth generation
ACL	Anthology Reference Corpus
ACM	Association for Computing Machinery
AI	Artificial intelligence
ANN	Artificial neural network
API	Application programming interface
AQR	Air quality prediction
AR	Annualized return
AR	Auto regression
ARIMA	Auto Regressive Integrated Moving Average
ASIC	Application-specific integrated circuits
At-LSTM	Attention Long short-term memory
AUD	Australian dollar
AV	Annualized volatility
BERT	Bidirectional encoder representations from transformers
BFGS	Broyden-Fletcher-Goldfarb-Shanno
Bi-BloSAN	Bidirectional block self-attention network
BiDLSTM	Bidirectional long short-term memory
BJ	Beijing
BP	Back-propagation
BPNN	Back-propagation neural networks

BPTT	Back-propagation through time
CAD	Canadian dollar
CapsNet	Capsule neural network
CG	Conjugate gradient
CHPC	Centre for High-Performance Computing
CIFAR-10	Canadian Institute for Advanced Research
CNN	Convolutional neural network
CNY	Chinese Yuan
COCO	Common objects in context
COVID-19	Corona virus pandemic of 2019
CPU	Central processing unit
CR	Cumulative return
DAE	Denoising autoencoder
DA-RNN	Dual-stage attention based recurrent neural network
DARTS	Differentiable architecture
DD	Draw-down
DEA	Denoising autoencoders
DJI	Dow Jones Industrial
DJIA	Dow Jones Industrial Average
DNN	Deep neural network
DSANet	Dual Self-Attention Network
DNN	Deep neural network
DP-LSTM	Deep differential privacy-inspired Long short-term memory
DSANet	Dual self-attention network
DSP	Digital signal processing
DSR	Design science research
DTW	Dynamic time warping
EA	Evolutionary algorithms
ECORR	Empirical correlation coefficient
ELU	Exponential linear unit
ERNN	Elmann recurrent neural networks

FFNN	Feed forward neural network
FLOPs	Floating-point operations per second
FPGA	Field programmable gate array
FS-RNN	Fast-slow recurrent neural network
GAN	Generative adversarial network
GARCH	Generalized autoregressive conditional heteroscedasticity
GB	Gigabytes
GBP	Pound Sterling
GP	Gaussian models
GPU	Graphical processing unit
GRNN	General regression neural network
HDD	Hard disk drive
HMM	Hidden Markov model
HM-RNN	Hierarchical multi-scale recurrent neural network
IMDbD	Internet movie database dataset
IndRNN	Independent recurrent neural network
IoT	Internet of things
IPPPD	Irregular pattern peak period detection
IQR	Inter-quartile range
JPY	Japanese yen
KDD	Knowledge discovery and data mining
L-BFGS	Limited-Memory Broyden-Fletcher-Goldfarb-Shanno algorithm
LFNN	Large feedforward neural network
LIME	Local interpretable model-agnostic explanations
LR	Linear regression
LSTM	Long short-term memory
LU	Linear unit
MAE	Mean absolute error
MAPE	Mean absolute percentage error
MDA	Mean directional accuracy
ME	Mean error

MEP	Mean error percent
MGU	Minimal gated unit
ML	Machine learning
MLP	Multi-layer perceptron
MNIST	Modified National Institute of Standards and Technology
MOM	Momentum models
MOrdReD	Memory-based ordinal regression deep neural networks
MRE	Mean relative error
MSE	Mean squared error
MSLE	Mean squared log error
MSPE	Mean squared percentage error
NASDAQ	National Association of Securities Dealers Automated Quotations
ND	Normalized deviation
NFL	No free lunch
NLL	Negative log-likelihood
NLP	Natural language processing
NPU	Neural processing unit
NZD	New Zealand dollar
PCA	Principal component analysis
PDA	Prediction difference analysis
PReLU	Parametric rectified linear unit
PRISMA	Preferred reporting items for systematic reviews and meta-analysis
PTB	Standard Penn Treebank
QRNN	Quasi-recurrent neural network
RAM	Random access memory
RBFNN	Radial basis neural networks
ReLU	Rectified linear unit
RHN	Recurrent highway network
RMN	Recurrent memory network
RMSE	Root mean square error
RMSLE	Root mean squared logarithmic error

RNN	Recurrent neural network
RRSE	Root relative squared error
SA	Sensitivity analysis
SAICSIT	South African Institute of Computer Scientists and Information Technologists
SAN	Self-attention network
SATNAC	South African Telecommunications Networks and Applications Conference
SeLFISA	Systematic enhanced Deep Learning Framework for Irregular Sequential Analysis
SELU	Scaled exponential linear unit
SEN	Algorithm sensitivity
SFNN	Small feedforward neural network
SGD	Stochastic gradient descent
SkipRNN	Skip recurrent neural network
SLR	Systematic literature review
SQuAD	Stanford question answering dataset
SR	Sharpe ratio
ST-ResNet	Spatio-temporal residual neural network
SVHN	Street view house number
SVM	Support vector machine
TaxiBJ	Trajectory data from taxicab GPS data and meteorology data in Beijing
TB	Terabytes
TCN	Temporal convolutional network
TPU	Tensor processing unit
TQDM	Training models with a progress bar
UCI	University of California, Irvine, School of Information and Computer Science
USD	United States Dollar
VAE	Variational autoencoder
WT	Wiki text

Research Outputs

The peer reviewed scientific research outputs from this thesis:

1. **K. Dandajena**, I. M. Venter, M. Ghaziasgar, and R. Dodds, (2023), "*A comprehensive explainable framework for designing enhanced deep learning models*," Journal of Information and Technology, 7(1), 47–57. https://doi.org/10.53819/81018102t3086.

2. **K. Dandajena**, I. M. Venter, M. Ghaziasgar, and R. Dodds, (2021), "*Selecting datasets for evaluating an enhanced deep learning framework*," SATNAC 21: Proceedings of the Annual Conference of the Southern African Telecommunication Networks and Applications Conference, Drakensberg, KwaZulu-Natal, South Africa, 21–23 November 2021.

3. **K. Dandajena**, I. M. Venter, M. Ghaziasgar, and R. Dodds, (2020), "*Complex Sequential Data Analysis: A Systematic Literature Review of Existing Algorithms*," SAICSIT '20: Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists, Cape Town, South Africa, 14–16 September 2020.

4. **K. Dandajena**, I. M. Venter, M. Ghaziasgar, and R. Dodds, (2019), "*An enhanced deep learning algorithm towards the accurate prediction of irregular sequential patterns*," SATNAC 19: Proceedings of the Annual Conference of the Southern African Telecommunication Networks and Applications Conference, Fairmont Resort, KwaZulu-Natal, South Africa 1–4 September 2019.

Chapter 1

Introduction

1.1 Background and introduction

When one set of technology is replaced by a new set in a relatively short period, it is considered to be a technological revolution [123]. Steam-powered engines influenced the development of the first industrial revolution, the use of electricity in mass manufacturing as well as automation pioneered the second industrial revolution, the third industrial revolution was driven by computerisation and the inter-connectivity of various technological devices. The *fourth industrial revolution* (4IR) will amalgamate modern physical spaces with cyberspace. It is characterised by smaller, cheaper and exponentially more powerful devices [118, p. 7]. According to Lee et al. [77], this revolution will impact the global society by its improved connectivity enabled by *fifth-generation* (5G) mobile technology, the abundance of data characterised by big data, automation, intelligent agents, robotics, *artificial intelligence* (AI), cybersecurity, hyper automation, edge computing, quantum computing, the *Internet of things* (IoT), nanotechnology, biotechnology, blockchain, vitality sensors, *three dimensional*

(3D) printing, augmented reality and virtual reality as shown in Figure 1.1.

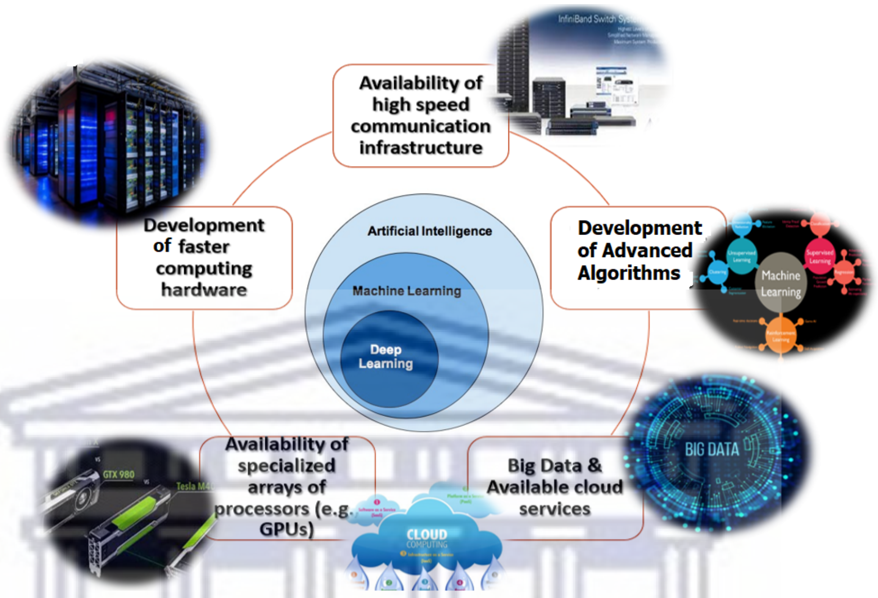

Availability of high speed communication infrastructure
Development of faster computing hardware
Development of Advanced Algorithms
Artificial Intelligence
Machine Learning
Deep Learning
Availability of specialized arrays of processors (e.g. GPUs)
Big Data & Available cloud services
BIG DATA
CLOUD COMPUTING
Machine Learning

Figure 1.1: Some of the major 4IR components that influenced the growth of AI [118]

Big data is considered as the liquid capital of this revolution and AI, the golden commodity for manipulating, analysing, modelling and forecasting these massive datasets [77]. The growth of data is phenomenal with more than 2.5 exabytes of data being created per day [52]. Hence, data "lakes" are created from real time interconnected portable digital systems, processes, gadgets and sensors such as our mobile phones, social media [13], location-based services, security cameras, financial transactions, news feeds and weather systems [16]. This poses several problems such as the storage of data, the security of the data based on secure encryption methods and the efficient analysis of the data [11, 16]. It is important to find ways to capitalise on this deluge of data to make informed decisions to improve and secure the global community [67].

Deep learning models are currently the state-of-the-art in machine learning.

These models have been successful in learning patterns and trends in massive datasets in a variety of domains. These models can carry out accurate prediction, ranging from mimicking human cognition, to predicting complex sequences such as sentences, domestic and industrial water/electricity usage. Deep learning models take the form of deep multi-layered neural networks, where each layer can consist of one or more algorithms such as a: *gated recurrent unit* (GRU), *long short-term memory* (LSTM) [58], *bidirectional mechanism* (Bi) and *attention mechanism* (Att). There is great flexibility in the architecture [103], i.e. the design and arrangement of layers, in any given deep learning model, in addition to a range of hyper-parameters that can be optimized [8, 10, 22, 41].

1.2 Problem statement

A review of deep learning literature revealed that a process of trial and error is used when designing the architecture of deep learning models [29]. At most, various rules of thumb and guidelines for best practices may be applied, but the overall process used to arrive at the design of each deep learning model is ultimately one of trial and error [5, 94]. In fact, in the majority of deep learning research, a discussion of the process used to arrive at the final model design is usually omitted, and focus on the final model's performance, where good result is taken to be a sufficient justification for the design of the proposed model. There is therefore currently no known process of systematically arriving at a deep learning model that provides improved predictive performance as compared to contemporary state-of-the-art models in a given domain [113].

Given that the architecture of a deep learning model directly determines the predictive success of the model, there is need for a systematic process to arrive at an effective deep learning model. Accordingly, the main focus of this research is to propose, develop and evaluate a framework that can provide a systematic approach to design and enhance deep learning models, as compared to the state-of-the-art models, in a given domain. The key idea underlying the proposed framework is to apply a *systematic literature review* (SLR) process to first identify the best deep learning models proposed in the literature in a given domain and then systematically combine and refine on the best elements of these models to arrive at an enhanced model that provides improved performance.

A systematic framework of this kind should help to provide a design process that is repeatable, explainable and straightforward [107]. Repeatability refers to the extent to which a process and its corresponding results can be successfully repeated. Explainability is the opposite of trial and error, and refers to the ability to clearly describe how the results of a process were arrived at. Finally, straightforwardness refers to the simplicity to implement it in a given domain [94].

One of the main challenges in deep learning research is to search through large numbers of research papers, that are published on a regular basis, to uncover and identify studies of importance and disregard irrelevant studies. A formal process by which to carry out this task is lacking.

Another significant challenge in deep learning research is a lack of standardisation of methods of evaluating models. There are three main criteria to evaluate models [107], namely: efficacy, consistency and efficiency. Efficacy refers to how successful a model is at predicting the target variable of a test dataset. Consistency is a measure of the robustness of a model to deal with unseen data. Efficiency refers to the time taken to train a model in relation to its complexity.

In general, virtually all research studies evaluate model efficacy, and some—but not all—studies evaluate model consistency. Model consistency is usually evaluated by applying one or more models under consideration to an unseen dataset, which is very often an unseen portion of the same dataset from which the training set was extracted. Where more than one model is used, there is no metric that can quantify the consistency of models in order to be able to objectively compare them. Efficiency is a criterion that is often entirely overlooked, and when it is considered, it is usually done by means of timing analysis without considering model complexity. The proposed systematic framework additionally aims to provide standardized methods of evaluating models in a given domain.

A category of data which has proven to be particularly challenging to predict is discrete irregular sequential data. A discrete irregular sequential dataset is defined as a dataset which is characterised by volatile patterns. Examples of such datasets include foreign exchange rate data, financial fraud data, stock price data, epidemiological and disease propagation data and weather forecasting [98]. An illustration of a discrete irregular sequential dataset can be seen in Figure 1.2, which is a graph of the value of the *United States dollar* (USD) versus the *Pound sterling* (GBP) from 1900 to 2015. The sharp peaks and troughs are circled in yellow. Foreign exchange data has proven to be difficult to predict as a result of its highly volatile nature. Comparatively speaking, other irregular sequential datasets such as weather, electricity and water usage. largely some consistent, seasonal and/or repeatable patterns with lower overall volatility, all of which help provide a regularity to these sequences that makes forecasting more feasible. In contrast, currency exchange prices are highly volatile and irregular. The currency exchange price of a given currency pair is dependent on numerous complex factors including the political, social and economic climate of one or more countries at any given instant, all of which can change rapidly from moment to moment [103].

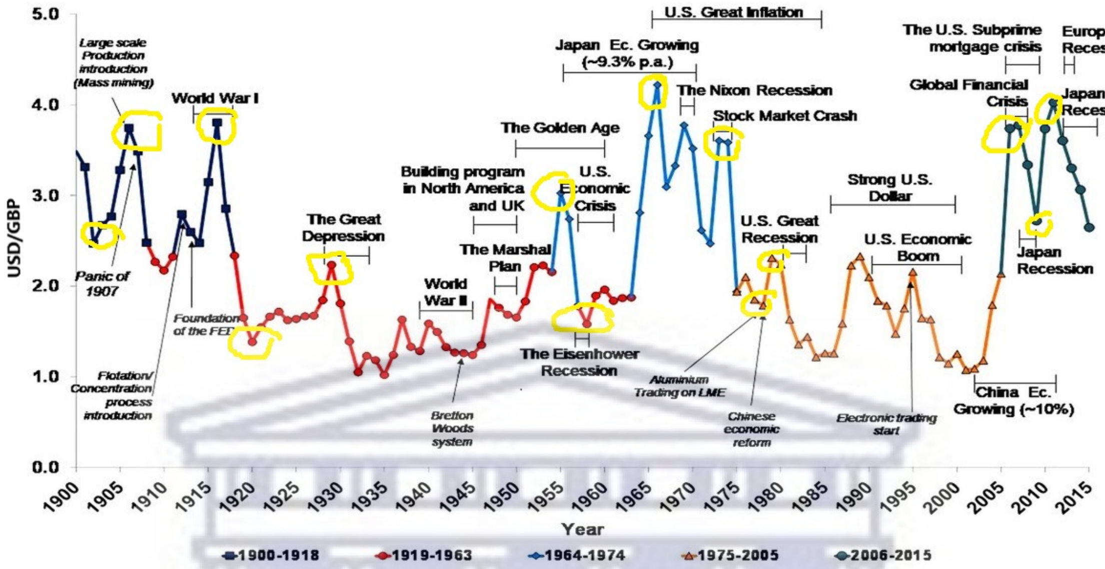


Figure 1.2: USD versus GBP price from 1910–2015 showing high volatility and discrete irregular sequential patterns (circled in yellow) [132].

The main outcome of this research is a systematic framework that can be used to produce an *enhanced deep learning model* based on current state-of-the-art deep learning studies. As newer studies emerge, the proposed framework will continue to remain relevant to produce enhanced models.

In summary, this thesis proposes a systematic framework that can be used to arrive at an enhanced deep learning model in a given domain, based on state-of-the-art deep learning models in that domain. The design of the framework is described in detail. In order to clearly demonstrate the potential of the framework, this thesis describes an application of the framework to the domain of forex price prediction to arrive at an enhanced forex price prediction deep learning model, as compared to state-of-the-art models which were found using a systematic literature review process.

Although it is customary in a thesis for the literature to be reviewed early in the thesis, in this thesis—since a systematic literature review process forms part of the proposed framework—-the literature is reviewed when the proposed framework is discussed in Chapter 4.

1.3 Aim of the research

The aim of the study was to develop a novel framework to enhance deep learning models for time-series prediction from existing state-of-the-art models and demonstrate that it works in a given context.

1.4 Research questions

The main question is: *How should deep learning models be developed in order to improve on existing state-of-the-art models when predicting discrete irregular sequential datasets?* This question can be expanded into the following sub-questions:

1. How should existing state-of-the-art deep learning approaches and relevant existing datasets in a given domain be effectively identified and selected?
2. How should the state-of-the-art deep learning approaches identified be systematically combined and improved to arrive at a deep learning model with enhanced performance?
3. How should the performance of deep learning models be evaluated when applied to such datasets?

1.5 Research objectives

To address the research questions:

1. A systematic literature review process was used to identify well-known deep learning models and datasets in a given domain and their respective performance and methodological challenges.

2. A framework to systematically improve on existing models in order to arrive at an enhanced deep learning model was developed.

3. The developed framework was applied to the field of forex price prediction in order to demonstrate the framework's potential to arrive at an enhanced deep learning model. To achieve this, experiments were carried out to compare the enhanced model with the identified state-of-the-art models.

1.6 Research methodology

It was found that *design science research* (DSR) methodology was an appropriate methodology to use for the research [140]. The methods included a SLR process combined with the *"preferred reporting items for systematic reviews and meta-analysis"* (PRISMA) [13, 104] and for the experiments—*grounded theory* by Glaser and Strauss [128] contributed towards the identification of deep learning models as well as discrete irregular sequential datasets. The utilisation of a hybrid methodological approach facilitated the design, deployement and evaluation of the SeLFISA framework. A hybrid methodological approach of this study allowed for the exploration of theoretical concepts as well as the development of practical artefacts which was compared with state-of-the-art models. The research methodology was presented in Section 3.2.3.

1.7 Contributions

1. The main contribution of this work is a novel framework which provides a systematic procedure which researchers can use to systematically im-

prove on the best deep learning models found in the literature in a given domain. As a concrete example, this work demonstrates the proposed framework in action in the challenging field of forex price prediction.

2. The second contribution of this work is a SLR methodology that can be used to systematically uncover, rank and select research studies of significance.

3. The third contribution is an enhanced deep learning model arrived at by applying the developed framework. The enhanced model outperformed state-of-the-art models in terms of performance, consistency and efficiency.

1.8 Research ethics

The research deals with data and information from academic, commercial, societal, environmental, state and personal resources. Appropriate digital moral behaviour and standards of cyber-ethics have been followed in this research study [65]. The research refrained from any cyber morality issues such as scientific fabrication, distortion or plagiarism, intellectual property infringement, hacking and posting incorrect/inaccurate information [65]. The University of the Western Cape Research Committee approved this research. The research does not qualify to go through the University of the Western Cape Faculty Board Research and Ethics Committees because it uses secondary data which does not involve direct connection with humans or animals.

1.9 Thesis outline

The thesis is organized as follows:

Chapter 1: Provided a brief background to the aims of the research, the research questions, objectives, methodology and contribution of the study.

Chapter 2: Provides background on the tools and techniques used when creating deep learning models. It discusses methods of preparing datasets, techniques of carrying out prediction, popular deep learning architectures, as well as delving into a number of deep learning optimization techniques.

Chapter 3: Details the research methodology applied to answer the research questions.

Chapter 4: Discusses in detail the structure of the proposed framework and its implementation. The results of the implementation of the framework include: a systematic literature review on forex price prediction; a series of state-of-the-art deep learning forex price prediction implementations; an enhanced forex price prediction deep learning model; and a comprehensive evaluation and comparison of the enhanced model with state-of-art deep learning models.

Chapter 5: Concludes the thesis by reflecting on the study, evaluating the findings, listing the contributions achieved, as well as providing directions for future work.

Chapter 2

Deep Learning Tools and Techniques

In this Chapter, core deep learning tools and techniques are presented. The development of theoretical and experimental foundational notions for the building of a deep learning framework for sequential time series prediction requires the use of these tools and techniques.

2.1 Terms and theories used

The field of artificial intelligence is characterised by often confusing and inconsistent terminology and theory. To provide a context for the rest of the thesis and avoid misunderstandings, this section will clarify the use of important terms and theories.

2.1.1 Terms

A list of important terms is provided below:

- **Architecture** : Refers to the arrangement, structure and configuration of the internal units of a neural network. Architectures such as the convolutional neural network and long short-term memory are influenced by networks such as feed-forward neural networks, and recurrent neural networks. An architecture influences algorithm design, while an algorithm influences model design [120].

- **Algorithm** : A machine learning algorithm is defined as the steps to execute, or recipe followed using data [120]. An algorithm is based on an identified architecture guided by training approaches such as supervised learning, unsupervised learning, or reinforcement learning [114]. Each training technique is supported by an internal training algorithm. Examples of unsupervised learning algorithms trained on unlabeled data include k-means clustering and association rules. Examples of reinforcement learning algorithms include Q-Learning algorithms [114] a model-free reinforcement learning algorithm which does not require a model of the environment and temporal difference algorithms [25, 120].

- **Model** : A machine or deep learning model is the result of applying a learning algorithm on a specific dataset. Thus, an algorithm becomes a model when it is trained or exposed to data, as shown in Equation 2.1. The goal of machine learning algorithms is to convert a dataset into a model. As a result, an algorithm can be defined as part of a model, but not the other way around. Generative models based on unsupervised machine learning algorithms and discriminative models based on supervised machine learning algorithms are examples of models [120].

$$Model = Training_t(Algorithm + Data), \tag{2.1}$$

where t is the training time.

– **Framework** : A machine or deep-learning framework is an interface, library, or tool that allows developers to easily build machine learning models without having to understand the underlying methods [120]. The *Caffe, Microsoft Cognitive Toolkit*, and *TensorFlow* are examples of general machine or deep-learning frameworks [120]. An implementation framework is the connection between efficient optimization algorithms, architectures, models, performance evaluation techniques and the problems being solved [66, 147].

– **Artefact** : Refers to a construct or any by-product of a development process. It is an umbrella term that could refer to an architecture, algorithm, model, or framework [53].

2.1.2 Theories

Grounded theory proposed by Glaser and Strauss is concerned with formulating hypotheses based on data that have been systematically collected and analysed. It provides a structured and analytical method for making discoveries by ensuring that the data gathered is adequate to justify the analysis of research results [128].

Grounded theory encourages researchers to be creative by applying critical thinking towards data analysis [64]. Coupling any recommended research methodology with grounded theory improves the research narrative's robustness. According to Debnath et al. [34], grounded theory is an ideal instrument for developing a theoretical foundation for deep-learning research.

2.2 Dealing with the prediction of sequential time-series

Artificial intelligence artefacts are currently being used for a wide range of real-world applications such as energy planning and smart grid management, sensor network monitoring, disease propagation analysis, natural language processing, image classification, and sentiment analysis [156]. The bulk of these artefacts rely on real-time sequential data exchange [93]. This results in large-scale complex data repositories comprising structured, semi-structured, and unstructured records. These are challenging to analyse using conventional tools and methodologies [127].

2.2.1 Datasets

Wherever data is acquired and indexed against time, time-series features appear in the dataset, this creates a series of time-stamped observations, with each observation x_t corresponding to a specific time t. Such sequential data can be categorised as time-series.

Discrete irregular patterns in any dataset reveal distinct signal patterns or characteristics, which might take the shape of regular, rhythmic growth or logarithmic decay. Irregular patterns can be found in many datasets with complex features such as high dimensionality, outliers, incompleteness and noise [102, 29].

For example when results are announced in political elections or *Coronavirus disease 2019* (COVID-19) protocols are announced, these can influence financial stock market prices [30]. Figure 2.1 shows such complex patterns for the adjusted closing prices of the Malaysia Stock Market from 2 January 2020–19 January 2021 [56]. The lock-down measures announced in Malaysia had a direct effect on the stock market as shown in Figure 2.1. The removal of the lock-down showed some recovery shocks thereby causing irregular patterns.

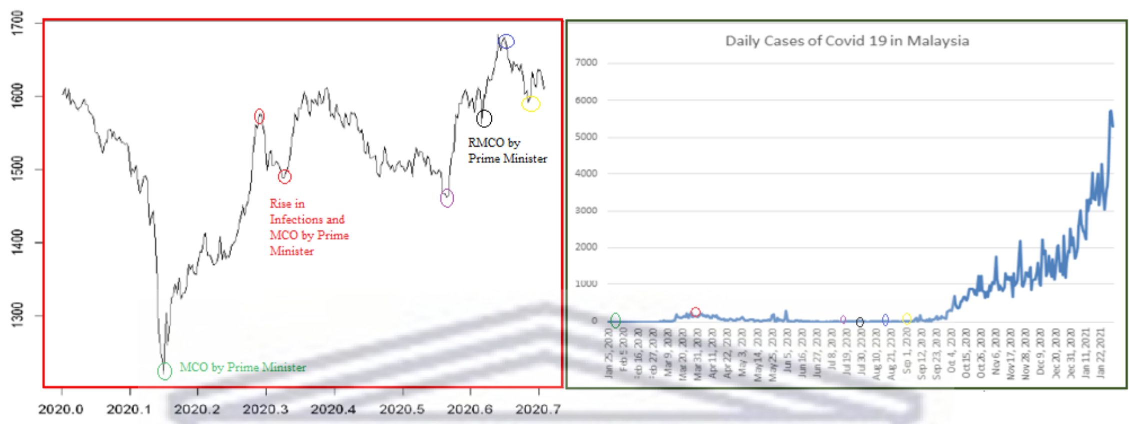


Figure 2.1: Irregular sequential patterns of the Malaysia stock market as a result of COVID-19 infections [56]

Discrete irregular sequential patterns are also prominent in health datasets such as heart beat rate. Procedures that provide accurate early detection for possible heart problems could assist health workers, potentially saving many human lives [96]. Systems that can detect and flag abnormalities allowing for better diagnosis are very important. This emphasizes the significance of a link between such datasets and prediction methods [115]. Several researchers such as the Bengio, Hinton, and LeCun developed data analysis tools with the capacity to provide future answers on such complex datasets [43, 75].

2.2.2 Prediction

Forecasting is about making future predictions, but this can only be done with data from the past, emphasizing the relevance of both the dataset and the prediction artefact. The demand for alternative explainable time series forecasting artefacts that can deliver accurate point forecasts is on the rise. Complex events, such as health, weather, and economic recessions or booms, as well as natural disasters, are one area of forecasting that have attracted more attention and contributed towards the development of such artefacts. Such events have a tremendous socioeconomic impact, but are difficult to predict [86].

Time series prediction has been used in a variety of domains with varied forecast horizons at different scales, and aggregation levels [89]. Time series forecasting utilise a model to predict future values of a target $y_{i,t}$ for a given entity i at time t. An entity can represent the temperature or the price of financial stock on the market [79]. The three most common approaches for time series forecasting are [134]:

1. **Naive approach:** This is the simplest time series forecasting method, which uses the most recent observation in a historical sequence of data to forecast the future value. It lacks the ability to forecast complicated sequential datasets efficiently and accurately.

2. **Statistical approach:** Such as *auto-regression* (AR) is a more complex method than naive ones [18]. AR forecasts future time series values using linear statistical methods based on historical sequential data.

3. **Machine or deep learning approaches:** are the most advanced time series forecasting methods. They go beyond linearity by being able to forecast several time points using regression and classification approaches based on machine learning. Machine learning has become a significant tool for time series forecasting and modelling due to the variety of time-series challenges and rising data availability and computing power [79, 89]. This method was discussed in greater depth in Section 3.2.3.

Feature and structure-based similarity measurement and visualisation techniques provide capabilities to distinguish between different sets of such sequential time-series datasets [116]. Examples of such techniques include VizTree, created to assist visual exploration techniques for examining time series patterns in datasets. It supports data analysis by discovery patterns in data with no requirement for prior knowledge. VizTree gives users the capacity to visually inspect data, it also offers creative interactive techniques for finding unexpected discrete patterns and anomalies in a dataset [80].

Another technique for visualizing time-series data is the use of spiral charts. It facilitates the detection of underlying data patterns for the analysis of challenging time series data and works well with massive data sets. Spiral charts generate graphs that reveal the underlying patterns in time series data, making them useful for making predictions about a variety of phenomena, including climatic changes and financial trends. Additionally, this offers a graphical comparison and summary reading method designed to help people's natural ability to find deep hidden patterns in datasets [87]. Calendar-based visualization is another technique for presenting meaningful time series graphically. It provides the ability to unpack temporal variables at different resolutions to reveal hidden patterns. Linear algebra is used to restructure the data, which is subsequently shown as calendar patterns, heatmaps and layouts for deciding the time series prediction approach [138].

Another technique for visualizing time series data that uses time boxes to represent visual queries is called TimeSearcher. To assist the human eye in data visualization, it offers anomaly detection. But in contrast, it is limited when processing substantially large sized data sets due to constrained graphical display space which makes it essentially unreadable [57].

Datasets characterised by a high level of irregularities pose prediction challenges and opportunities [125]. In order to address such challenges, a number of forecasting methods and solutions have been explored, proposed and applied in the context of discrete irregular time-series forecasting. Researchers have specific objectives when studying time-series datasets. Some are interested in understanding the behaviour of the data, others are aligned towards forecasting future values or optimizing systems [49, 155].

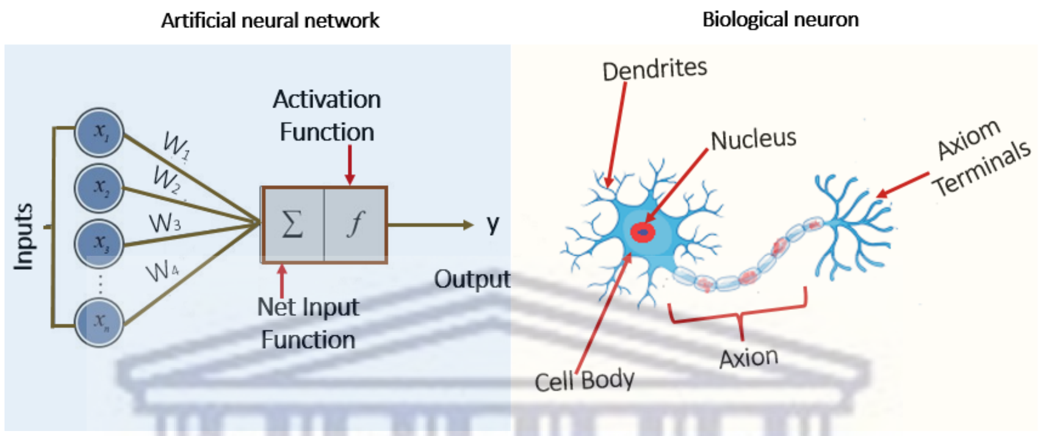


Figure 2.2: ANN—perceptron versus biological—neuron [45]

Traditional statistical sequential forecasting approaches can handle time-series. Technical statistical prediction methods that have been applied include *autoregressive integrated moving average* (ARIMA) or the exponential smoothing method by Box and Jenkins [18]. Predicting f_t from f_1, $f_2, \ldots, f_n$, is given by Equation 2.2:

$$f_t = \beta_0 + \sum_{t=1}^{n} (\beta_t f_{t-1} + \epsilon_t), \tag{2.2}$$

where t is the time and f_1, $f_2, \ldots, f_n$ is the stationary time-series, $\epsilon_1, \epsilon_2, \ldots, \epsilon_n$ is the residual and β_0, $\beta_1, \ldots, \beta_n$ are the model coefficients.

However, statistical linear auto-regressive models are known to be weak when considering non-linear or complex environments [18]. This makes them unattractive for predicting such data. The biology of the brain served as inspiration for various artificial intelligence techniques [43]. There are few effective methodologies that enhance time-series forecasting performance in terms of efficiency, accuracy, and reliability due to the complexity of time-series analysis and forecasting [82]. Figure 2.2 shows a biological neuron on the left modeled by an *artificial neural network* (ANN)—perceptron on the right by McCulloch and Pitts in 1943 [45].

An ANN can simulate the computational function of neurons within the human nervous system. ANNs allow non-linearity by means of models designed with non-linear activation functions [71]. Machine learning allows a machine to learn or extract patterns from data without human intervention [32]. Known machine learning driven models that have been applied for sequential time-series modelling include a multi-layer perceptron, decision trees, support vector machines, linear regression and Bayesian networks [31, 106]. Modern models have demonstrated significant accuracy improvement over traditional linear statistical models [114]. However, they face a wide range of challenges such as:

1. inefficiency when predicting highly nonlinear patterns [99],
2. they can not accurately analyse or predict increasingly sophisticated modern complex data [30],
3. having repeatability and reproducibility issues since their designs have been created through trial and error [153] and,
4. are not able to match evolving hardware infrastructure such as *graphical processing units* (GPUs), digital signal processors, and field-programmable gate arrays [81].

2.2.3 Architectures and algorithms

Standard machine learning algorithms are constructed with shallow architecture, which has performance limits [75] and thus can not keep up with the exponential increase in data complexity. Deep learning was developed as a solution for difficult problems based on hierarchical situations [4, 99, 148].

Models generated through deep learning networks have been more accurate and effective in learning features compared with machine learning models [43]. This is illustrated by the artificial intelligence maturity diagram in Figure 2.3. The diagram demonstrates three transitions namely (1) threshold logic units—early 1940s to the mid-1960s, (2) connectionism—early 1980s to mid-1990s and (3) modern deep learning—mid-2000s to the present. These developments come with a complexity cost as more variables and parameters are involved [70].

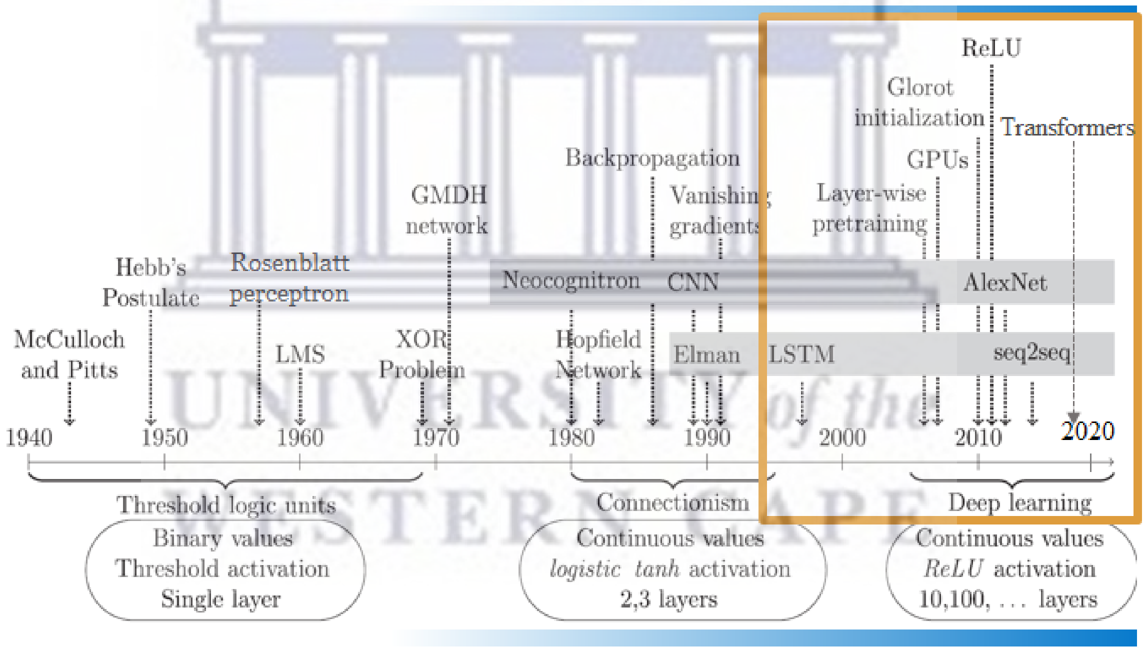


Figure 2.3: History and maturity of neural networks [70]

Deep learning networks have the capacity and potential to solve problems in areas where primitive machine learning is insufficient even if the field is still developing [13, 46, 98]. Factors that influenced the development of deep learning networks are:

1. big data [119],
2. supercomputers, GPUs, *tensor processing unit* (TPU), *neural processing unit* (NPU) and InfiniBand communication systems [148],
3. huge investment in human capital [122] and
4. the global race for artificial intelligence dominance [60].

Four categories of algorithms that have been applied to prominent deep learning networks models are:

1. **Recurrent Neural Networks (RNNs)** favoured by Schmidhuber and Hochreiter for sequential time-series processing [58], are based on a cyclic connection which distinguishes them from other feed-forward neural architectures. The cyclic connection facilitates updating of the current state based on the previous state and the current input during sequential learning [72].
2. **Convolutional neural networks (CNNs)** are a special form of neural network (ConvNet) designed by LeCun et al. [15]. CNN imitates human vision by using several building parts such as convolution layers, pooling layers and fully connected layers. Facial recognition, image processing and virtual, augmented and mixed reality are examples of computer vision applications that use them. A convolution is a mathematical approach for combining two functions to create a third function that shows how the shape of one is affected by the other. ConvNet uses a backpropagation learning algorithm to condense a given image into a form that is easier to analyse without sacrificing crucial internal qualities in order to produce an accurate forecast. [76]. A dilated architecture of CNNs is suited for forecasting long historical financial time series as an alternative to RNNs. Dilated CNNs are faster and easier to implement, and they outperform linear and recurrent models [17].

3. **Generative adversarial network (GAN)** models by Goodfellow et al. [44] are made up of a generator and a discriminator, both of which are trained using the adversarial learning concept. *Gated recurrent units* (GANs) are used to estimate the potential distribution of real-world data samples and create new samples from that distribution. The field of computer vision, image processing, cybersecurity, and voice and language processing have all benefited from GANs [143]. GANs trained on historical time-series datasets have lately shown success in tasks previously dominated by RNNs, such as medication sales forecasting and financial stock price prediction. This illustration shows how GANs increase the accuracy of modelling tasks that are dominated by anomaly detection [63].

4. **Transformer neural networks** provides a sequential design for efficiently resolving long-range dependencies [139]. They are supervised sequential learning model, which instead of employing an RNN mechanism—it intelligently exploits the attention mechanism in a deep feed-forward network as an alternative of traditional encoding and decoding approaches. Transformers, unlike RNNs, do not always process data in the same order, the attention mechanism offers context for any place in the input sequence. This allows them to process a sequence quickly since they focus on the most critical elements. Transformers have revolutionized the field of sequential modelling but rely on huge datasets where they can be trained to learn everything from scratch [68].

Transformers have been shown to be particularly successful for *natural language processing* [139]. *Bidirectional encoder representations from transformers* (BERT) and *generative pre-trained transformer* (GPT) have been trained using large datasets [35]. This makes them appealing in terms of forecasting long time-series sequences with high prediction capacity [157].

Predominant variants for sequential data influenced the development of RNNs to process dynamical systems with long-term dependencies [50]. When compared to traditional sequential data modelling techniques, the RNN architecture provides a stable performance. This makes them a good candidate for sequential processing [126].

RNNs were developed by Elman (1990) but they suffered from the problems of exploding and vanishing gradients which led to the development of LSTM by Hochreiter and Schmidhuber in 1997 [58]. The *rectified linear unit* (ReLU) activation function in LSTM addressed the vanishing gradient problem. Whilst the *backpropagation through time* (BPTT) learning algorithm tackled the exploding gradient problem through gradient clipping. The stochastic gradient descent with momentum by Sutskever [127] introduced dropout for reducing overfitting [91].

These developments failed to fully exhaust the optimum strength of RNNs when solving complex sequential problems [155]. A detailed analysis of some RNNs for sequential time-series prediction is shared below:

1. **Gated recurrent architectures** are an improvement of general RNNs described as a family of gated or cell algorithms. LSTMs or GRUs are flavours of gated RNNs used for sequential modelling [126] and can memorize useful historical sequential time-series data by forgetting irrelevant data. A combination of gated RNNs and attention techniques focus the predictive process onto the most relevant data [155].

   (a) **LSTM** [58] learns long-range dependencies better than conventional RNNs. The LSTM shown in Figure 2.4 has a gated RNN architecture equipped with accumulators and gating functions. The introduction of LSTM architecture was largely influenced by the deficiencies in the general RNNs such as (1) the exploding and vanishing gradient problem, (2) the sensitivity of the network decays over time as new inputs overwrite the activations of the hidden layer and the network, and (3) optimization hurdles that plague general RNNs [13].

LSTMs remember information to avoid the long-term dependency problem [4]. An LSTM architecture can be improved by manipulating the bias of the forget gate. The anatomy of a LSTM model can be enhanced to solve time series such as stock exchange data [48].

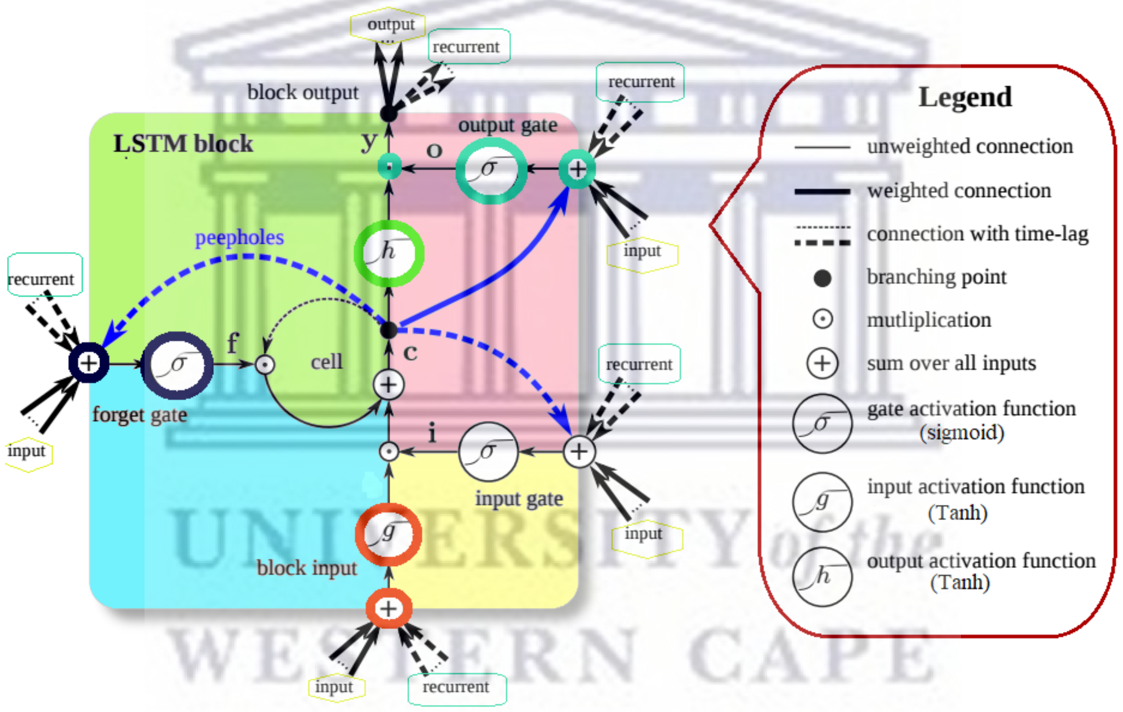


Figure 2.4: Illustration of LSTM architecture [48], where f is forget gate, i is input gate, o is the output gate, c is the cell state and y is the output.

LSTMs learn long time-series dependence in sequential forecasting using gates with memory for remembering past sequential patterns. This makes them an effective tool for predicting future patterns:

i. **Forget gate or front gate** is a block equipped with a sigmoid function that determines what information to throw away as illustrated in Equation 2.3 [56].

$$f_t = \sigma(W_f.[h_{t-1} + x_t] + b_f), \tag{2.3}$$

where f_t is a forget gate, W_f is the weight, h_{t-1} is the hidden cell at time $t-1$, x_t is input at the current time stamp t and b_f is the bias.

ii. **Input gate** consists of a $\tanh$ layer. It tells what new information is going to be stored in the cell state from current input as shown in Equation 2.4. It has two parts, namely, (1) the input gate which is another sigmoid layer which fires outputs between 0 and 1 and decides which values we will update and (2) the $\tanh$ layer creates a vector of new candidate values, C_t^h which was used to update in Equation 2.5 the cell state at time t. An aggregation of these two layers creates an update to the state.

$$i_t = \sigma(W_f.[h_{t-1} + x_t] + b_i), \tag{2.4}$$

$$C_t^h = \tanh(W_c.[h_{t-1} + x_t] + b_c, \tag{2.5}$$

where i_t is an input gate, W_x is weight for respective gate x neurons, hi_t is the output of the previous LSTM block (at timestamp t_{t-1}), x_t is input at current timestamps t and b_x is bias of the respective gates x

The next step is to update the old cell state, C_{t-1} into the new cell state C_t as shown in Equation 2.6,

$$C_t = f_t.C_{t-1} + i_t.C_t^h \tag{2.6}$$

iii. **Output gate** is the last gate to determine the next hidden cell as in Equation 2.7. There is a need to first run a sigmoid layer to determine what will be output. This is then attenuated between -1 and 1 using a $\tanh$-layer in Equation 2.8, multiplied by the output of the sigmoid gate.

$$o_t = \sigma(W_o.[h_{t-1} + x_t] + b_o), \tag{2.7}$$

$$h_t = o_t. \tanh(C_t). \tag{2.8}$$

These extra memory cells or gates in LSTM demand more memory which increases the computational complexity for sequential prediction. Developing different flavours of LSTM has led to better models. LSTM flavours such as: *stacked LSTM; bidirectional LSTM; multidimensional LSTM; grid LSTM; frequency-time LSTM almost similar to grid type; differential RNN and local-global* have been developed [59].

Jozefowicz et al. [68] have optimised GRU architectures which outperform LSTM architecture for sequential datasets. This is because the LSTM architecture is ad hoc and uses components whose purpose is not immediately apparent. GRU tends to outperform LSTM in time-series applications analysis not related to language manipulations [68].

(b) **GRU** architecture is an improved LSTM that combines forget and input gates into a single gate called the update gate and a reset gate [26]. Figure 2.5 shows a GRU structure. In sequential prediction, LSTMs are outperformed by GRUs [68]. GRU architecture is faster than LSTM because it uses less network parameters [68].

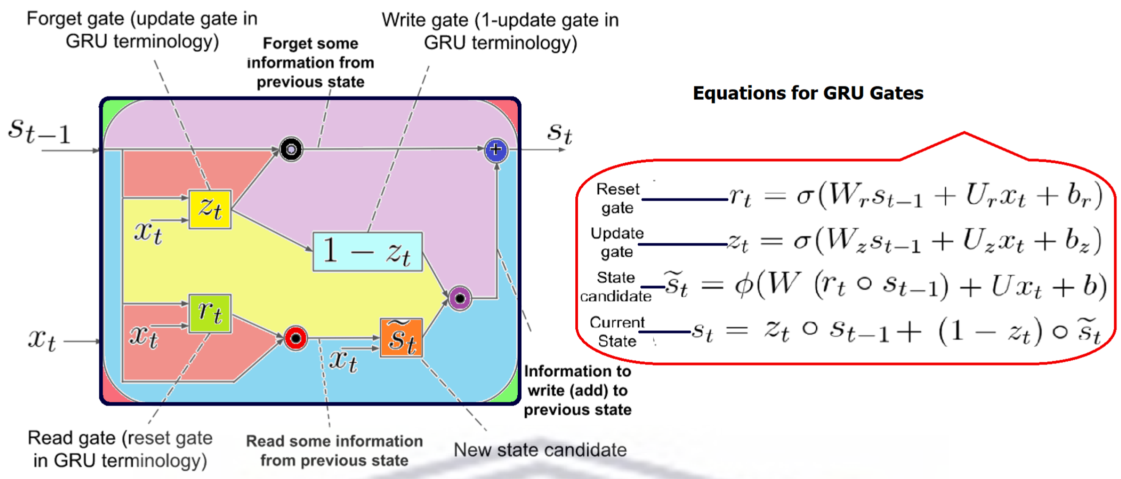

Forget gate (update gate in GRU terminology)
Forget some information from previous state
Write gate (1-update gate in GRU terminology)
Equations for GRU Gates
s_{t-1}
s_t
z_t
x_t
$1 - z_t$
r_t
$\tilde{s}_t$
Reset gate
$r_t = \sigma(W_r s_{t-1} + U_r x_t + b_r)$
Update gate
$z_t = \sigma(W_z s_{t-1} + U_z x_t + b_z)$
State candidate
$\tilde{s}_t = \phi(W\ (r_t \circ s_{t-1}) + U x_t + b)$
Current State
$s_t = z_t \circ s_{t-1} + (1 - z_t) \circ \tilde{s}_t$
Information to write (add) to previous state
Read gate (reset gate in GRU terminology)
Read some information from previous state
New state candidate

Figure 2.5: A GRU architecture [40]

RNNs exist in a variety of variants, some of which include:

(a) **Unitary RNNs** solve hard tasks involving very long-term dependencies which cannot be solved using ordinary RNNs. They are more computational and memory-efficient than gated RNN architectures [78].

(b) **Gated orthogonal RNNs** hybridise remembering the strength of unitary RNNs and the forget ability of gated RNNs to come up with an architecture at modelling long-term sequential dependencies [95].

(c) ***Clockwork RNNs*** have a unique approach to tackling long-term dependencies in sequential forecasting. Clockwork RNNs lower the number of RNN parameters in sequential forecasting and classification which enhance performance and accelerates network analysis [73].

(d) ***Independent RNNs*** by Li et al. [78] are good at solving gradient vanishing and exploding problems associated with general RNNs.

(e) ***Quasi-RNNs*** alternate between convolutional layers and recurrent pooling functions to improve low parallelism. Stacked quasi-RNNs outperform stacked LSTMs of the same hidden size in terms of prediction accuracy. They are up to 16 times faster at train and test times due to their higher parallelism [19]

(f) ***Skip RNNs*** are self-optimised models which shorten the effective size of the computational graph and conduct fewer updates for long-term sequences. Skip RNNs can learn to skip state updates making the optimization problem simpler [20].

(g) ***Multi-dimensional RNNs*** are an important tool for modelling high dimensional sequences which cannot be handled by simple RNNs [47].

(h) ***Fast–slow (FS) RNNs*** outperform other models when learning long-term dependencies [95]. This was illustrated when comparing FS-LSTM architecture with stacked-LSTM and sequential-LSTM architectures, under the same parametric conditions. FS-LSTM outperformed other models in learning long-term dependencies [47].

The ***bidirectional RNNs or LSTM or GRU*** [117] improve sequential optimisation and increase the amount of input information available to the network. Bidirectional RNNs split neurons of a regular RNN into two directions during their deployment—one for positive time direction focused on forwarding states and the other for negative time direction of backward states. The development of the Bidirectional RNN was meant to address analysis problems of outputs that may depend on next elements rather than only the previous sequential elements. The BiRNN normally contains RNNs which process input information based on the generic forward sequential order as well as reverse input order [155].

The bidirectional mechanism provides RNNs with the ability to detect irregular elements during training and forecasting. The bidirectional mechanism captures both the future and past context of the input sequence better than the LSTM. The forward pass for hidden layers in bidirectional RNNs is the same as for a unidirectional RNN, except that the input sequence is presented in opposite directions compared with the two hidden layers. The output layer of bidirectional RNNs is not updated until both hidden layers have processed the entire input sequence. Bidirectional LSTMs show very good results as they can deal with the context more efficiently than LSTMs [28]. They require the start and end points of a sequence to predict both the positive and negative time directions simultaneously. When applying the bidirectional LSTM architecture in traffic speed prediction it produces exciting results both in terms of accuracy and robustness [28]. Differential RNNs proved to be the best for sequential modelling especially in capturing spatio-temporal patterns and thus these RNNs are good candidate architectures for sequential prediction [92]

RNNs combined with an attention mechanism in deep learning improve model performance efficiency, especially for long sequences. The attention mechanism was proposed to solve long-range dependence problems in sequential modelling which cannot be solved by bidirectional RNNs of LSTMs [155].

The attention mechanism is good at capturing features that could influence any situation. It is specific in its approach and has been widely used in *natural language processing* (NLP). Sheny et al. [155] applied in several attention mechanisms with a combination of RNN models. The research observed that the bi-directional block self-attention network was more accurate and time-efficient than RNNs, self-attention networks, CNN, bidirectional LSTM and bidirectional GRU. Combining gated RNNs and the attention improves the predictive analysis process [155].

Sequential architectures have further influenced the design of some modern enhanced artefacts such as:

(i) *BERT* by Devlin et al. [35] is an extension of the transformer architecture. It is simple and cheap to design because it is focused on transfer learning. It demonstrates its robustness when deployed in natural language processing with large unlabeled datasets [35].

(ii) *Capsule networks* by Hinton et al. [54]—analyse images in a manner that approaches the way humans do it. Capsule networks are primarily designed to solve limitations associated with CNNs such as the Picasso problem when analysing spatial relationships between high-level components of an object. However, capsule networks have not been scaled to larger datasets.

(iii) *Temporal convolutional networks* by Bai [9] use a convolutional architecture for sequential modelling.

(iv) *Dilated recurrent neural networks* [23] are more suitable for modelling very long-term dependencies. It has been used in the design of hybrid sequential models.

2.3 Optimisation

Having identified a diverse set of architectures, algorithms and models, it is notable that these technologies present with some common characteristics that have an impact on their performance. Optimal deep neural network design is difficult to achieve due to the large number of variables involved [98].

As a result, tuning a deep neural network model to outperform existing performance standards is a process that must take into account a wide range of criteria such as the selection of input data sets, data preparation processes, network designs and algorithms [98].

Other factors that influence performance include the type of activation function [45], loss functions or cost function; optimization methods, a layer and parameter counts controlling technique; network memory requirements estimation procedure, weight initialization strategies, understanding learning rates, application of learning optimisers; computational configuration; epochs controlling technique during training, nature of regularization method and weight initialization strategy [45]. It is important to implement a hyper-parameter tuning method or any configuration setting that might affect performance such as layer size, magnitude, regularization techniques and normalization schemes for input data [55].

2.3.1 Regularization

Mathew et al. [88] found that designing any deep neural networks is focused on optimisation of the network structure, avoiding the problem of both over and under-fitting and by allowing the network to converge using a correct loss function and selecting proper hyper-parameter, parameters and optimisers.

A good regularization method avoids over-fitting by using a penalisation mechanism of weights. A child can learn to recognize a new kind of object or animal using only a few examples and is then able to generalize that to other circumstances. To mimic this learning ability in complex environments, designing an artefact that can produce consistent and understandable results in different prediction scenarios would be a game-changer for the previously highlighted 4IR challenges [88].

Regularization techniques address [55] over-fitting and under-fitting issues by adjusting the gradient update scheme and managing the capacity of the deep neural network [88], where overfitting and underfitting are avoided by stopping the training at the best network performance point. In the sequential prediction process, the creation of a separate validation data set will help in addressing this problem but this is not enough on its own, there is a need to couple it with other techniques. Major regularization methods help in reducing the margin between the training loss and the validation loss of a hyperparameter tuning process and include dropout, drop-connect, Lasso regression penalty and ridge regression penalty (weight decay) [111].

2.3.2 Activation functions

Activation or suppression functions are important parameters for the performance of any neural network. They allow neural networks to learn non-linear relations between inputs and the desired outputs [45]. They are critical in the performance of deep neural network models since they capacitate networks to learn complex relationships. Task performance and training dynamics of a DNN model depends on the choice and deployment mechanism of activation functions [45].

Every activation function produces a unique performance result when applied to the same architectures. Applying different activation functions on to the same architecture or within different hidden layers and comparing their performance provides a better approach towards the design of optimal prediction artefacts. However, relying on a single activation parameter when designing optimal models may not provide state-of-the art performance. It is important to explore the possibility of applying a multidimensional approach that considers a variety of parameters but this comes at a cost of complexity with variable explosion characteristics [111].

Some examples of these activation functions are listed by Nwankpa et al. [100] include linear activation for linear neural networks, binary step activation for binary classification tasks, sigmoid for binary classification and attention models, hyperbolic $\tanh$, *rectified linear unit* (ReLU) family with many flavours such as ReLU, leaky ReLU and *parametric ReLU* (PReLU), *exponential linear unit*

(ELU), *scaled exponential linear unit* (SELU) for CNNs and RNNs, softplus and maxout. Neurons get stuck in the upper and lower areas of the sigmoid and hyperbolic tangent as described by Nwankpa et al. [100]. The introduction of ReLU activations changed deep neural networks since it was almost impossible to train the networks using other activation functions.

Salehinejad et al. [111] demonstrated how the problem together with the respective input data signal guide the choice of activation function. An activation function such as tanh and sigmoid, saturate the network neurons faster, which can result in the gradient vanishing. Furthermore, the tanh activation function also causes dynamic instability during the gradient update process for different weights. ReLU has flexibility in large gradient flows, thus when the weight matrix grows the neuron may remain inactive during training [100]. The technique of setting the threshold of an activation value at zero when using the ReLU activation function, makes it computationally cheap. Experiments by Agarap [2] cited the dying neurons problems associated with deep neural network models driven by the ReLU activation function during the learning process compared to the softmax activation function. They also show that ReLU also experienced convergence and accuracy challenges during image prediction processes [28].

2.3.3 Training strategy

The model optimisation process is also affected by the choice of a training method. Training model is part of solving an optimization problem in deep learning and involves the selection of proper training algorithms, scheduling step sizes as well as tuning other hyperparameters in a sensitive manner. A proper training approach balances the reduction of training complexity—reduction of training error with an acceleration of algorithm convergence by picking the correct global minima [111]. Correct learning in deep neural networks produces good generalization behaviour.

2.3.4 Architecture design

To design compact deep neural network models for sequential predictive performance on unstable data with the best accuracy and least computational demands—it is important to consider algorithms within the internal architectural design. The choice of architecture and network is a major bottleneck when considering both the performance and energy efficiency of a deep neural network algorithm. Design inefficiencies are normally caused by existing ad-hoc designing procedures. A well-designed architecture enables smooth internal data flow between each layer similar to the data flow neurons in the human brain. An efficient model has a simple topology with few hidden layers and parameters but gives a similar or better performance than its deeper and more complex counterparts [46].

The deeper the network in terms of neurons and hidden layers with internal weight adjustments, the more the model can reach a convergence point, which then improves its generalisation. Some researchers adopted an ad-hoc method of manipulating design properties of each layer of neurons to achieve optimum performance of a particular model. It is not advisable since it is not an understandable procedure or approach [46].

2.3.5 Choice of optimiser function

Selection of a correct learning rate stabilises the training process of any DNN model by reducing of error of the neural network's guesses [46] . If the learning rate is low, the network may diverge instead of converging and can waste resources since the network could converge more slowly. Momentum updaters such as Nesterov's momentum, RMSProp, Adam and AdaDelta are critical in determining the right choice of the learning rate for an optimisation problem [44].

The *minimal gated unit* (MGU) model for sequential learning is a derivative of the GRU. In the MGU the reset gate and input gate are integrated into one gate. To improve minimal gated units for sequential data modelling, Takamura and Yamane [130] introduced an optimised MGU for processing sequential data in a simpler, faster and more accurate way than LSTM, GRU, and ordinary MGU. The design principle of this optimised MGU model is based on applying a Chrono initialiser as the initialization method of MGU [130].

2.4 Summary

Chapter 2 presented a range of deep learning tools and techniques that deal with the prediction of sequential time series. These tools and techniques are the primary elements that are considered, utilised and configured in the framework proposed in this thesis to arrive at an enhanced deep learning model in a given domain. Therefore, the background provided in this chapter serves as an underpinning for the description of the framework in the next chapter, and more importantly, for the implementation of the framework to the domain of currency exchange forecasting in Chapter 4.

Chapter 3

Research Methodology

In Chapter 2, an overview of a range of deep learning methods and techniques towards sequential time series prediction was given. In this chapter the theoretical perspective, methodology and methods that drive the research are discussed.

3.1 Research approach

To improve the performance of any deep learning artefact is a process of iterated refinements [30]. Many methodologies and methods for sequential time-series prediction have been proposed and implemented [131]. The philosophical stance and theoretical perspective of the researcher inform the methodology and provide a context for the research process [27].

The research approach illustrated in Figure 3.1 broadly outlines a work plan to address the following seven questions:

1. What are our research questions?

2. What are our research objectives?

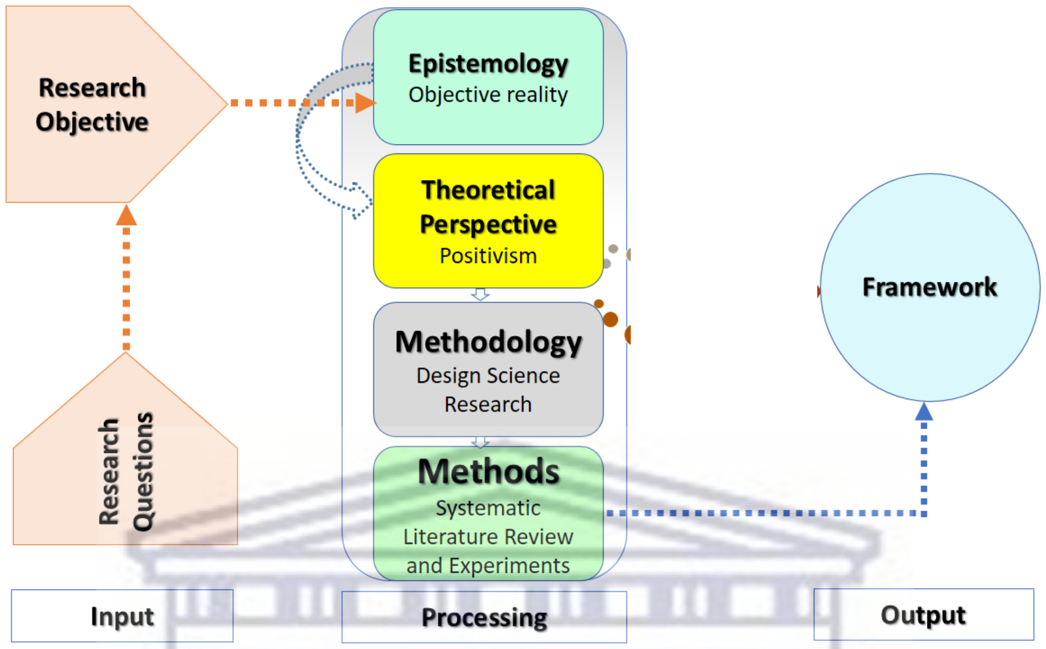


Figure 3.1: Research approach [27]

3. What epistemology informs the suggested theoretical perspective?
4. What theoretical perspective underpins the preferred methodology?
5. What methodology guides the choice and use of the proposed methods?
6. What are the appropriate methods and how do we plan to use them?
7. What is the end product?

Observing the bottom row of Figure 3.1, it can be seen that the overall research work plan is divided into three phases, namely, *Input*, *Processing* and *Output*. The following sub-sections describe these phases.

3.1.1 Input

The start node of the research approach in Figure 3.1 deals with:

1. **Research questions**—are the foundation of this investigation. The problem is identified and the specific research questions are developed to ad-

dress it.

2. **Research objectives**—are inextricably linked to the research questions. Properly designed research questions lead to *specific, measurable, achievable, relevant and time-bound* (SMART) objectives [133].

3.1.2 Processing

This is the most important section of the conceptual research approach with the following components:

1. **Epistemology**, according to Crotty [27], epistemology is concerned with the theory of the nature of knowing. Metaphysics combines epistemology with ontology. It gives a conceptual foundation for the decision about the types of knowledge that are feasible, as well as ensuring that they are both adequate and legitimate. Epistemological beliefs can range from objectivism to subjectivism. Objectivism asserts that a meaningful reality exists outside of the mind and can be described by quantifiable attributes that are unaffected by the observer or subject [27]. Subjectivism on the other hand, maintains that meaning is imposed by the subject's mind without the help of the outside world, that there is no meaning outside of the mind and that the observer is not independent of the subject. The objectivist viewpoint provides a theoretical foundation for quantitative research and researchers are more likely to do quantitative rather than qualitative study as a result [62]. Some studies feel objectivism and subjectivism are completely contradictory, while others believe they are complementary [38]. This research takes on an objectivist stance and thus does quantitative research [29].

2. **The theoretical perspective** is directly influenced by the choice of the epistemology. The theoretical perspective distinguishes between two major research philosophies (1) positivism—often known as scientific and (2)

interpretivism or anti-positivism.

Positivists use objective analysis to separate the object from the topic of the research [135]. Positivists concentrate on the facts or what is already known and disregard anything else, claiming that events should be isolated from one another and that observations should be repeatable. According to Gall [39], positivist research is fundamentally the same as quantitative research.

Interpretivism, on the other hand, or anti-positivism, denies objective reality and maintains that any reality can only be examined through creations such as awareness, language and shared meanings from the perspective of the individuals directly engaged. Interpretive research aims at comprehending phenomena by the subjective interpretations that observers attribute to them and it argues that phenomena should be investigated in their natural setting. As a result, interpretivism concentrates on the study and scope of meaning [109].

This research study is positivist. This paradigm provides a variety of methodological options for researchers who use quantitative methodologies.

3. **Methodology**—the theoretical approach influences the research methodology. A methodology provides a set of guidelines for conducting research. The methodology best-suited for designing a deep learning framework is DSR, which focuses on the production of an artefact [140]. This methodology was discussed in more depth in Section 3.2.

4. **Methods**—Every research instrument is intrinsically linked to commitments to specific representations of the world and to understand them.

This means that a method's efficiency is largely determined by the epistemology's justification [61]. The term research methods refers to a set of tools that can be used to collect data. Methods are led by the study goal, thus they are inextricably linked to the research methodology. Experiments, observations, systematic literature reviews, surveys, interviews and other research methods are examples of how data can be collected. Experiments, surveys and some observations are quantitative in character, whereas interviews are qualitative in nature. When a method involves dealing with original data, it is referred to as a primary method and when data was collected by someone else for other reasons, it is called secondary method.

This study used experiments and a hybrid systematic literature review to collect data.

3.1.3 Output

The final output of the conceptual methodological pipeline shown in Figure 3.2, is a deep-learning framework capable of producing enhanced deep-learning models. Research articles, journal and conference paper publications are considered as outputs. In conclusion, the output, selected methods, methodology, theoretical perspective and epistemology are all intertwined.

3.2 Design science research

The DSR methodology illustrated in Figure 3.2 is a research strategy that comes with a set of standard guidelines to describe and justify the methodologies and tools chosen for this research [140]. The chosen methodology has a robust loop-back function to accommodate changes done at various stages during the execution cycle. Knowledge creation and contribution is the overall

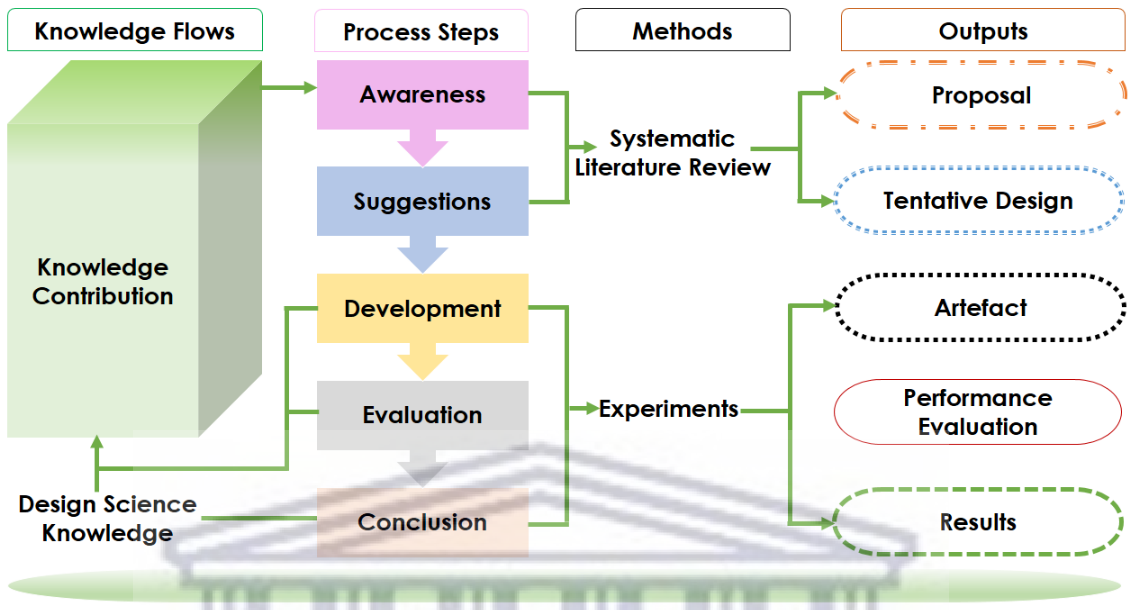


Figure 3.2: Outline of DSR methodology [140]

outcome of this methodology illustrated in Figure 3.2. The four phases of DSR are:

3.2.1 Phase 1—Knowledge flows

In the context of developing a novel framework to enhance deep learning models for time-series prediction, the study presents a unique contribution to knowledge. The foundation for knowledge generation is provided by the first phase of DSR shown in Figure 3.3. Design, analysis, reflection and abstraction are used to fill in the gaps.

3.2.2 Phase 2—Process steps

1. **Awareness of the problem**—New developments in industry or research groups, individual experiences, experiments, relevant sources, interdisciplinary discussions, workshops, conferences, seminars and interviews, can all help raise awareness of the problem. In this scenario, the method

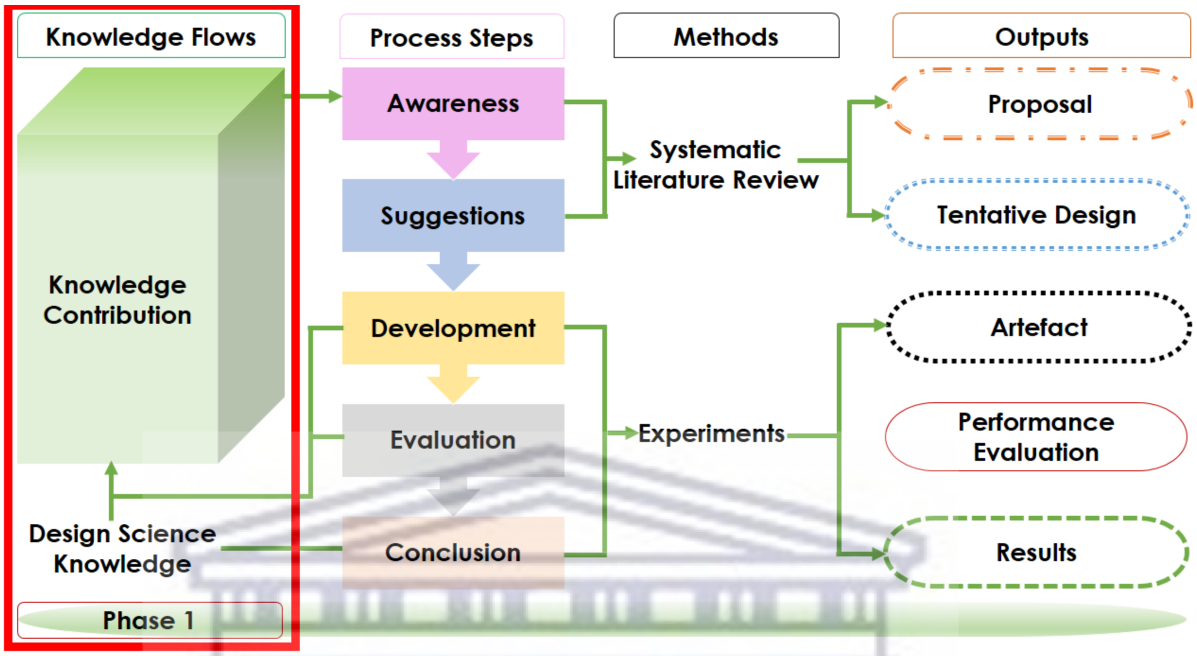


Figure 3.3: Phase 1 of knowledge flows of the DSR [140]

used, such as the systematic literature review to be discussed as part of selected methods in Section 3.2.3, really does have an impact on the awareness phase. This is illustrated in Figure 3.4.

2. **Suggestions** result in a tentative design. The systematic literature review guides the development of a tentative design. It's also possible to incorporate the performance of a prototype based on the proposed design at the suggestion phase. The research is thrown aside and deemed unsuitable if the tentative design does not present a solution. This stage is simply a brainstorming session for new functionality based on a unique arrangement of current or new parts.

3. **Development** step may begin when the tentative design of the suggestion phase has been accepted. The tentative design is further examined and executed during the development step. Depending on the artefacts, the experimental methods employed and the development process fol-

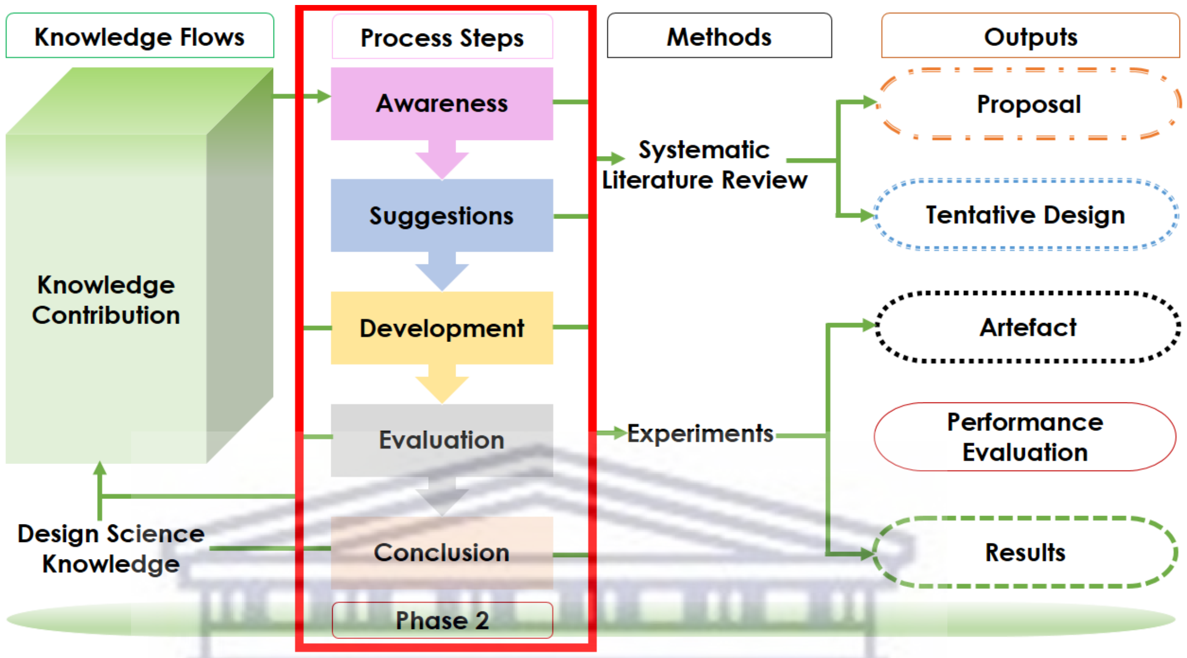


Figure 3.4: Illustration of DSR showing Phase 2 of process steps [140]

lowed may differ. Beyond the application of models, the novelty is essentially in the design of a framework as an artefact. Unexpected problems may arise during the development phase, resulting in various constraints, at which time the tentative design is re-evaluated and the process is restarted from phase one. It is worth noting that, despite these potential issues, useful content is being added to the knowledge base.

4. **Evaluation** follows the development of a solution. During the evaluation step, the artefact is tested against certain multidimensional criteria that are always implicit and made explicit. Quantitative and qualitative deviations from the predicted findings are documented, analysed and explained. These findings either confirm or contradict current literature, expectations and domain norms in line with the study. If the findings are not satisfactory, the process is resumed. In general, rather than being fully abandoned, adjustments are made in light of any new observations as shown in Figure 3.4 on Page 43.

5. **Conclusion** step is the end of the research cycle. The evaluation phase's findings are summed together and added to the body of knowledge for deep learning. These findings are compiled and the knowledge gained is classified either as a fact that has been learned or as aberrant behavior that defies explanation and could be the focus of future research. If the findings were not satisfactory, the method was evaluated and improved in order to achieve a greater level of accuracy.

3.2.3 Phase 3—Selected methods

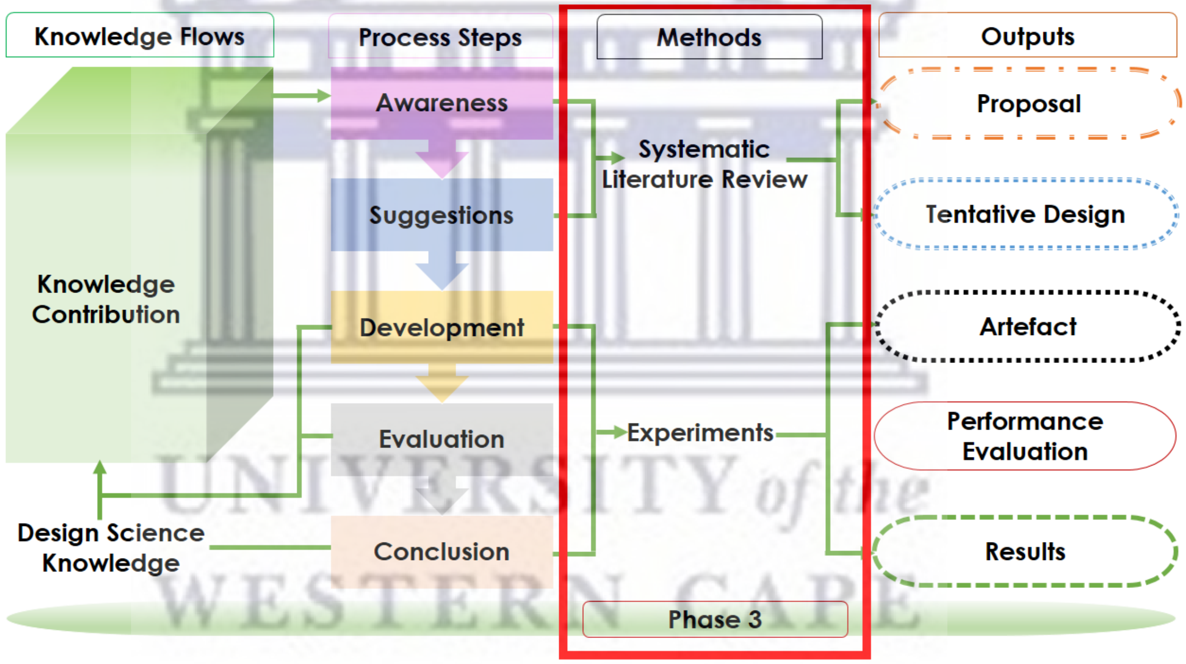


Figure 3.5: Illustration of DSR showing Phase 3 of selected methods [140]

In the context of Figure 3.2 in Section 3.1.2, there are a variety of methods that can be used in research towards knowledge generation. However, the approach used must be acceptable for the theoretical perspective at hand. The following are the two main methods employed in this study shown in Figure 3.5 are:

(a) Method 1—Systematic literature review

In this research, awareness of the problem and suggestion phases of the process steps in Section 3.2.2 are guided by the SLR. It is a review of clearly formulated DSR questions that use systematic and explicit methods to identify, select and critically appraise relevant research, and to collect and analyse data from the studies that are included in the review. The SLR process is a robust tool that follows the PRISMA methodology of Abhela et al. [1]. The SLR is deemed to be the most appropriate method to identify existing datasets, prediction models and evaluation mechanisms as it guides the development of a tentative design. This stage focuses on trying to find initial answers to the following questions:

1. *Which datasets are sequential in nature with irregular characteristics?*
2. *Which artefacts in the form of models have been applied elsewhere to analyse such datasets?*
3. *How were those artefacts evaluated in terms of determining their performance?*

The chosen SLR methodology for the identification step of the proposed framework demonstrates the breadth and depth of the existing body of knowledge of deep-learning frameworks, as well as identifying inconsistencies and gaps in this body of information, as illustrated in Figure 3.6 [150].

This SLR, according to Peter et al. [101], provides a comprehensive, targeted, dependable, reproducible, and extensive literature overview [6, 104, 137]. The SLR procedure employed in this study had the following five stages:

Stage 1—**Identification** of specific keywords to search for irregular-patterned sequential time-series prediction, as well as identifying appropriate online research database platforms, for example Google Scholar. The identified keywords are used to search for published accredited peer reviewed journal arti-

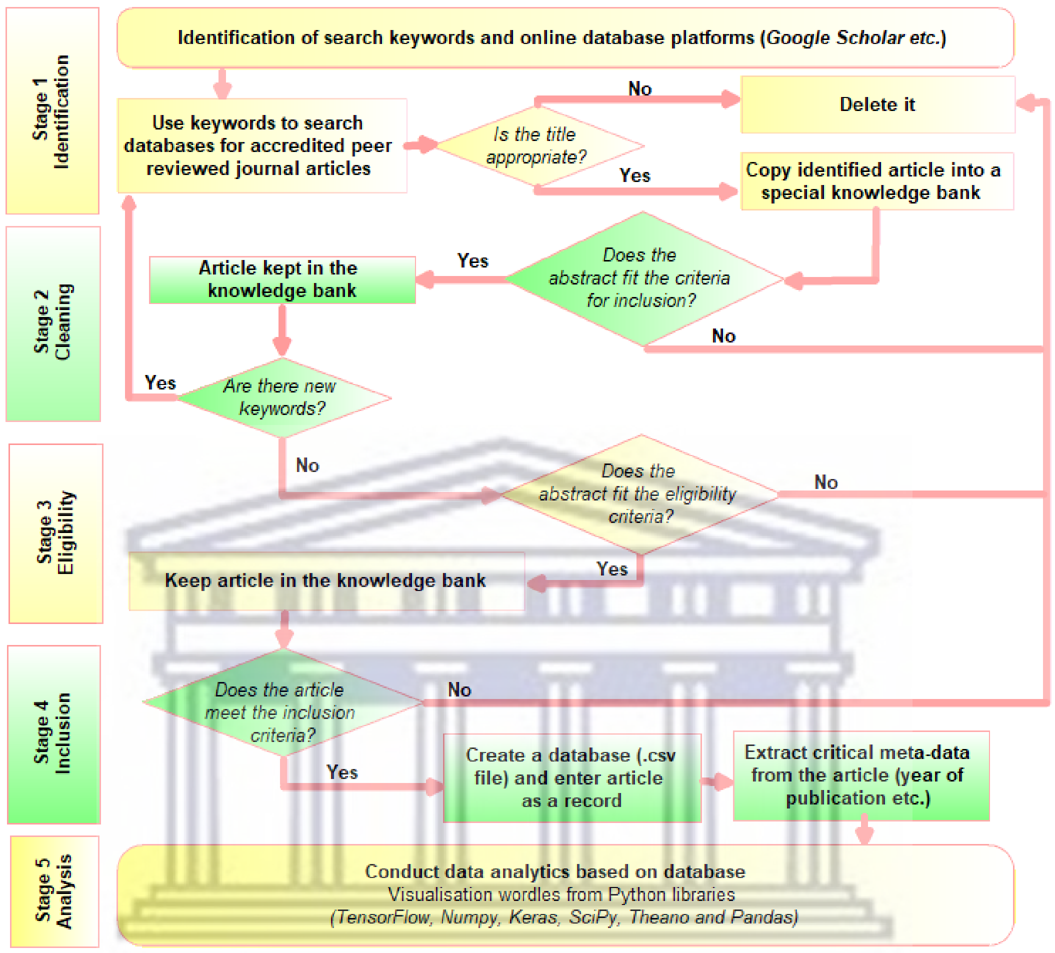

Stage 1 Identification
Identification of search keywords and online database platforms (Google Scholar etc.)
Use keywords to search databases for accredited peer reviewed journal articles
Is the title appropriate?
No
Delete it
Yes
Copy identified article into a special knowledge bank
Stage 2 Cleaning
Article kept in the knowledge bank
Yes
Does the abstract fit the criteria for inclusion?
No
Yes
Are there new keywords?
No
Stage 3 Eligibility
Does the abstract fit the eligibility criteria?
No
Yes
Keep article in the knowledge bank
Stage 4 Inclusion
Does the article meet the inclusion criteria?
No
Yes
Create a database (.csv file) and enter article as a record
Extract critical meta-data from the article (year of publication etc.)
Stage 5 Analysis
Conduct data analytics based on database
Visualisation wordles from Python libraries
(TensorFlow, Numpy, Keras, SciPy, Theano and Pandas)

Figure 3.6: Systematic literature review methodology

cles on online research database platforms [104]. The journal article titles are then used to determine which papers to consider and these are then saved for later evaluation.

Stage 2—**Cleaning** of the list of identified journal articles gathered in Stage 1 of the SLR. This stage requires a qualitative assessment of each article's suitability based on its abstract. The articles that did not meet the inclusion criteria were then removed. If additional keywords discovered in the abstracts, they were added to the original list of keywords and Stage 1 was repeated, in a *grounded theory fashion* [128].

Stage 3—**Eligibility** of the articles identified in Stage 2. Stage 2's output is subjected to a thorough qualitative screening process based on predefined eligibility criteria [124]. This stage results in a long list of articles, with those that did not meet the qualifying criteria being discarded.

Stage 4—**Inclusion** of the state-of-the-art articles selected based on identified inclusion criteria. The articles found during this step are then entered into a database. The entire article is considered at this point. Critical metadata, such as the year of publication, is qualitatively recognized and recorded in fields, each with an appropriate column heading. The articles that are not included in the database should now be deleted. The output for this stage is a comma-separated values file with extension '`.csv`'.

Stage 5— An **analytical** quantitative method was used to examine the database. Within a Jupyter Notebook environment, data analytic processes were performed utilizing Python libraries such as TensorFlow, Numpy, Keras, SciPy, Theano and Pandas. The relationship between essential aspects of the records was mapped across selected fields. During stage 5 the relationship of records was visualised as a *word cloud* for better graphical representations. Finally, the used codes, tools and the database of identified literature was uploaded on an online platform as open data to allow free access to other researchers.[1]

(b) Method 2—Experiments

The outputs of the SLR procedure informs the tentative experimental design. To assist researchers in the future, a novel framework was developed that describes a process that starts with a review of existing state-of-the-art deep learning models and datasets for time-series prediction and uses knowledge of these models to systematically derive an enhanced deep-learning model that

[1]All the experimental code is given in the Jupyter Notebook files on the GitHub website at: `https://github.com/Dandajena/SDA/`.

can predict discrete irregular sequential patterns more effectively. The research utilized DSR through experimental methods to design the framework.

Deep-learning workloads require an intensive computing environment. Complex deep learning modelling cannot be done by researchers using simple computational resources such as laptop or desktop computers. This is because time is needed for iterations, specifically, time to train with multiple specifications in the form of hyper-parameters and general parameters. Hence, to accommodate complex permutations, experiments utilized scalable hybrid high-end computational processing environments provided by the *South African Centre for High-Performance Computing* (CHPC). The CHPC resources provided NVIDIA GeForce MX130 *graphical processing units* (GPUs) and random access memory ranging between 20–210 gigabytes of random access memory, 10 terabytes hard disk drive memory; a CUDA toolkit for GPU deployment; Anaconda distribution software with Python and Jupyter Notebook, and the Keras, TensorFlow, Pandas and other libraries. Detailed experimental settings have been described [30].

3.2.4 Phase 4—Expected outputs

The DSRM's final phase in Figure 3.7 is made up of research outputs, which include the following:

1. **Proposal** implementation of SLR method to inform research proposal. A precise plan or blueprint for the proposed research is generated as an output to create a plan for the research. The proposal is the most important step in the research process and it involves conducting keyword searches in order to develop a scientifically appropriate study proposal.

2. **Tentative design** is guided by insights from the SLR, a tentative design is developed for the proposed artefact.

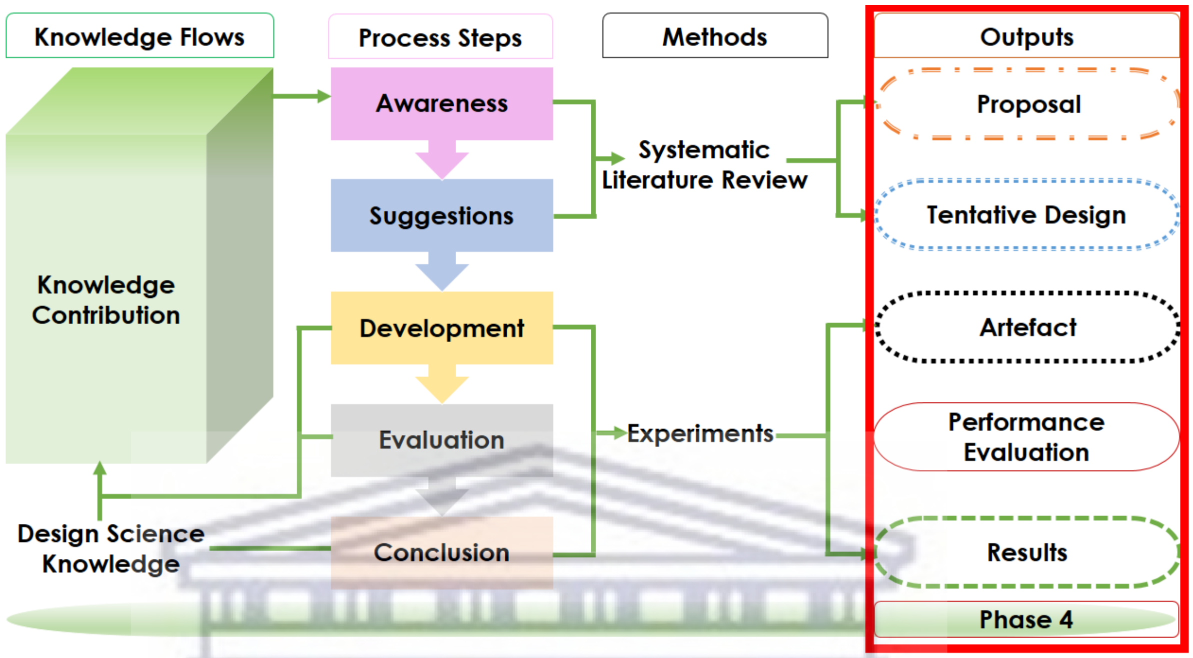


Figure 3.7: Illustration of DSR showing Phase 4 of produced outputs [140]

3. **Artefact**—experiments highlighted in the previous experimental section, guide the development of a novel framework to enhance models for predicting discrete irregular sequential patterns as the end product of the research.

4. **Performance measures** specific metrics were applied to evaluate the implemented artefacts. These include *mean absolute error* (MAE), *mean squared error* (MSE) and the coefficient of determination—R-squared R^2, the ratio of the total number of parameters and modelling duration. The stability, explainability and repeatability are some of qualitative criteria evaluated using visualisation, analysis and the expertise of human experts.

5. **Results** are recorded, published and shared using online platforms such as Github, conference papers and journal publications.

3.3 Summary

This section explained the DSR methodology to design the framework for enhancing and upgrading the leading deep learning models. The implementation of the designed framework and the recording of findings are discussed in Chapter 4.

Chapter 4

Framework Design, Implementation and Results

The purpose of Chapter 4 is to create and execute a novel framework based on the research methodology and methods presented in Chapter 3. This chapter will also demonstrate the effectiveness of the framework through the results. There is a literature evaluation and examination section that considers specific arguments as well as the gaps in seminal readings identified by the systematic literature review.

4.1 SeLFISA framework design

Currently no known procedure could be found that can be systematically followed to improve existing research in a given domain [83]. In most cases a trial-and-error approach is being used to develop models with superior performance over current *state-of-the-art* models [154]. There is no comprehensive approach that encompasses the varying stages of design, implementation and evaluation. To address this shortcoming a six-step method for improving the best deep learning artifacts for such difficult forecasting was developed.

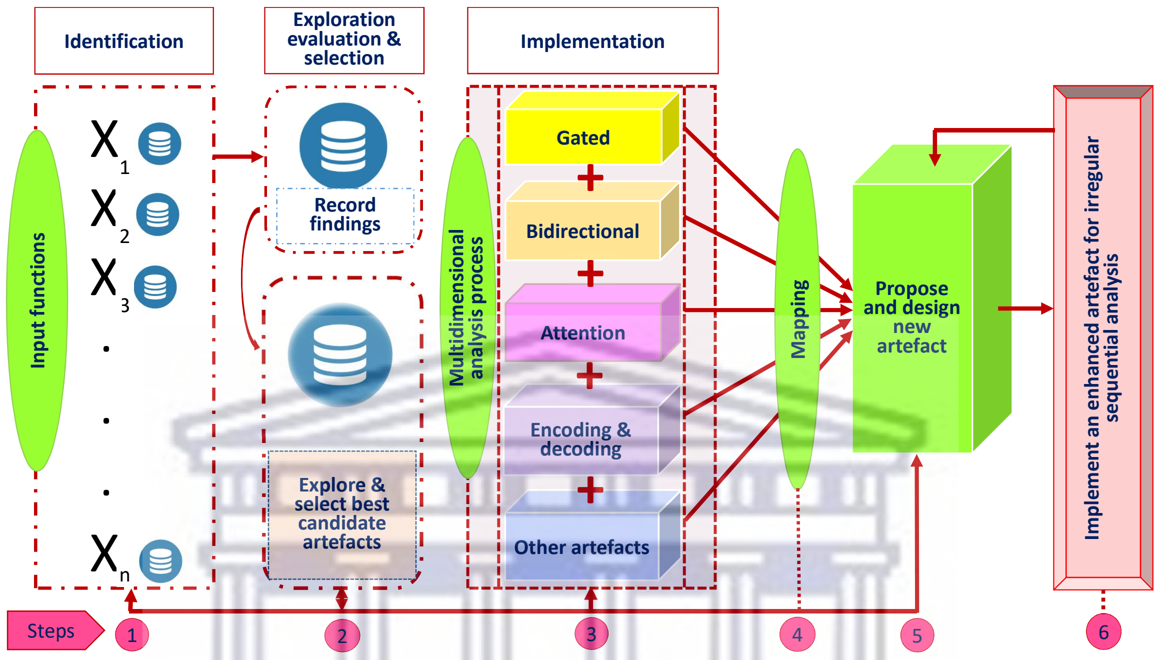

Identification
Exploration evaluation & selection
Implementation
Input functions
X_1
X_2
X_3
X_n
Record findings
Explore & select best candidate artefacts
Multidimensional analysis process
Gated
Bidirectional
Attention
Encoding & decoding
Other artefacts
Mapping
Propose and design new artefact
Implement an enhanced artefact for irregular sequential analysis
Steps
1
2
3
4
5
6

Figure 4.1: SeLFISA Framework

The five interrelated process steps of the selected DSR of Chapter 3, provided a platform for creating and contributing to scientific body of knowledge. The suggested DSR methodology resulted in the ***Systematic enhanced deep Learning Framework for Irregular Sequential Analysis*** (SeLFISA) shown in Figure 4.1. SeLFISA was used to create an enhanced deep-learning model for currency exchange prediction. It begins by identifying and implementing the best deep learning models available in the literature for a given domain, then combining and refining the best components of these models to create a more effective model [30].

4.2 Steps for SeLFISA framework

Six key steps make up the structure of the novel framework. These are:

4.2.1 Step 1—Identification

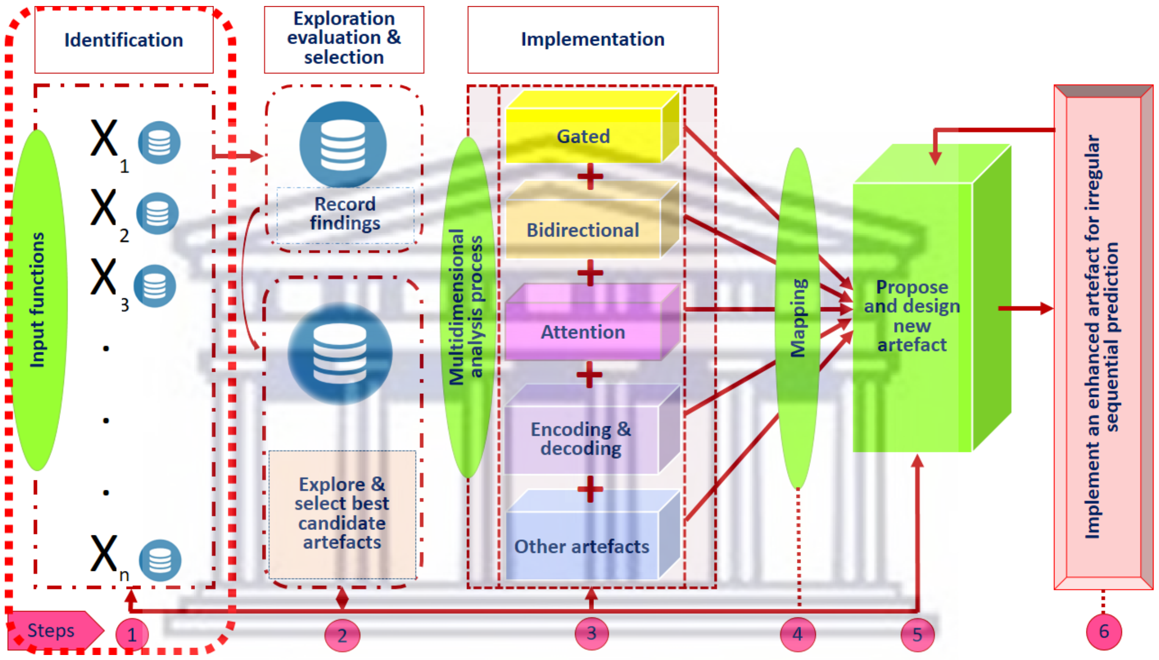

Identification
Exploration evaluation & selection
Implementation
Input functions
X_1
X_2
X_3
X_n
Record findings
Explore & select best candidate artefacts
Multidimensional analysis process
Gated
Bidirectional
Attention
Encoding & decoding
Other artefacts
Mapping
Propose and design new artefact
Implement an enhanced artefact for irregular sequential prediction
Steps
1
2
3
4
5
6

Figure 4.2: Step 1—Identification of the SeLFISA Framework

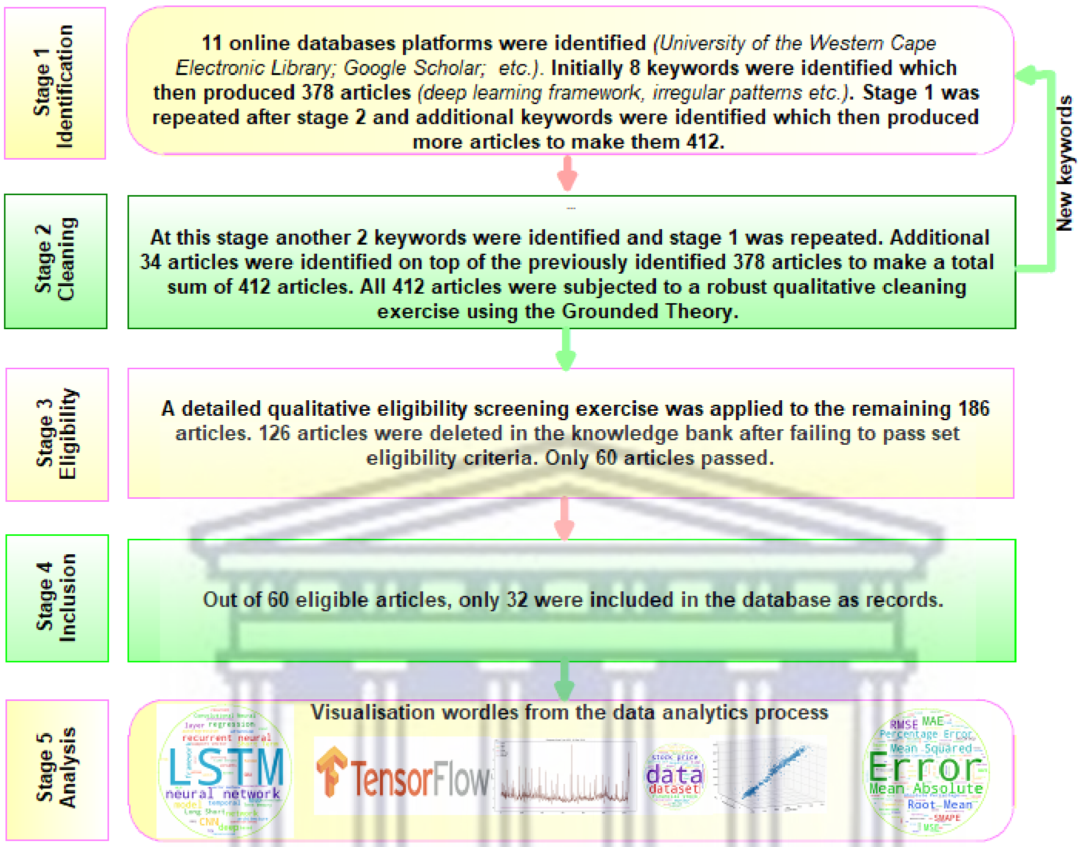
This first step in Figure 4.2 involves carrying out a SLR method of DSR to identify key existing datasets, prediction models, and evaluation mechanisms. This step also creates a precondition for implementing the SeLFISA framework by ensuring that it produces repeatable and reliable output. Failure to satisfy the initial condition will not only affect the entire development process of creating better models to analyse such datasets but will negatively affect the substantiation of the results. It considers the outcomes of the SLR process. This process results in a core set of relevant papers [29, 30].

4.2.2 Applying Step 1 and results

The following is a step-by-step account of the implementation and results of the first step of the SeLFISA Framework, guided by the design principles outlined in Chapter 3. The SLR was used to find relevant research publications that could provide details on current datasets, prediction models and evaluation processes [29]. The systematic process defined by Step 1 of Identification addresses questions concerning (1) the nature of time-series sets of data with discrete irregular sequential characteristics, (2) artefacts in the form of models that have been applied elsewhere to predict using such datasets and (3) evaluation mechanisms in order to arrive at the first research objective—portraying past and present approaches using deep learning models. Figure 4.3 on Page 55 summarises the phases of the systematic literature review to provide the results of the implemented SeLFISA framework and Figure 4.4 on Page 56 shows a word cloud of widely used architectures. This step's execution plan is as follows:

(a) **Stage 1**—at initially, a total of eight keywords or search phrases were used to search relevant literature. As shown in Figure 4.3, some of the subjects presented included a *deep-learning framework, sequential models optimization, irregular patterns, time-series forecasting, parameter, volatile financial prediction* and *unusual weather forecasting*. Two more new keywords were uncovered after the Stage 2 cleaning process. As a result, articles were identified using a total of ten keywords.

The following 11 online research database systems were used in the search University of the Western Cape Electronic Library, Google Scholar, CiteSeerx, GetCITED, Microsoft Academic Research, Bioline International Directory of Open Access Journals, PLOS ONE, Papers with Code, BioOne, Science and Technology of Advanced Materials, New Journal of Physics,



11 online databases platforms were identified *(University of the Western Cape Electronic Library; Google Scholar; etc.)*. Initially 8 keywords were identified which then produced 378 articles *(deep learning framework, irregular patterns etc.)*. Stage 1 was repeated after stage 2 and additional keywords were identified which then produced more articles to make them 412.

At this stage another 2 keywords were identified and stage 1 was repeated. Additional 34 articles were identified on top of the previously identified 378 articles to make a total sum of 412 articles. All 412 articles were subjected to a robust qualitative cleaning exercise using the Grounded Theory.

A detailed qualitative eligibility screening exercise was applied to the remaining 186 articles. 126 articles were deleted in the knowledge bank after failing to pass set eligibility criteria. Only 60 articles passed.

Out of 60 eligible articles, only 32 were included in the database as records.

Figure 4.3: Processing the results of the SLR

ScienceDirect and NIPS. During this stage, 412 search results were compiled into a complete collection of peer-reviewed literature. The inclusion of an article or document was based on the findings of each online database platform's first ten pages. These items were kept for subsequent processing.

(b) **Stage 2**—the *grounded theory approach* [128] was used to identify new keywords whilst considering the abstracts of 378 articles discovered in Stage 1. This revealed two more keywords namely *sequential learning* and *financial signal processing* were used as input for Stage 1 to identify more possible articles. This resulted in the identification of 34 more articles. However, 226 of the 412 articles were eventually deemed to be unsuitable for the study and were deleted, 186 articles were kept.

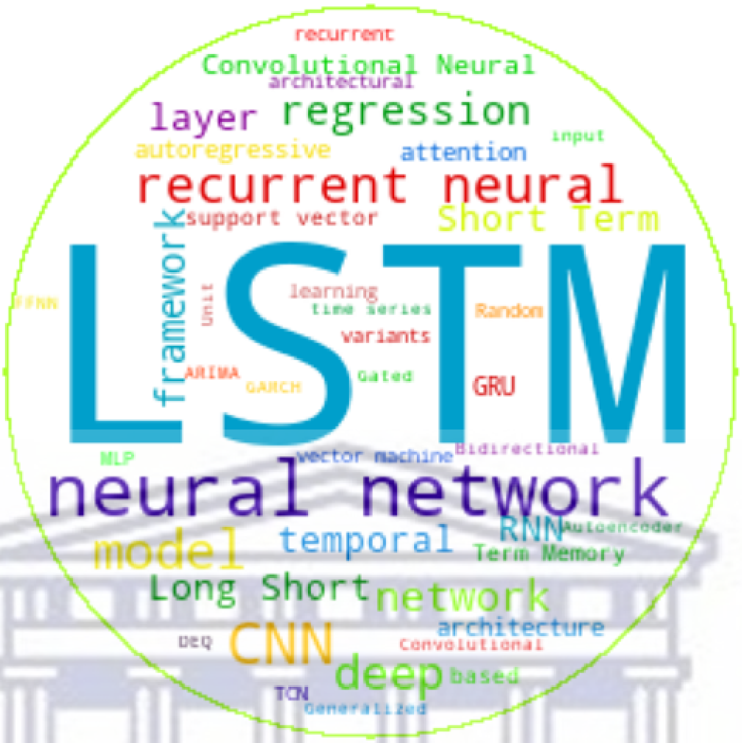


Figure 4.4: Word cloud illustrating frequently used sequential architectures

(c) **Stage 3**—the remaining 186 publications were assessed using the following criteria: identified keywords, publication dates from 2016 to the present, model relevance in terms of complicated datasets, accreditation and journal quality (i.e., citation index) [124]. Each article's abstract was thoroughly read, if it failed to meet eligibility criteria, it was deleted leaving 60 articles.

(d) **Stage 4**—all 60 articles in the folder were now thoroughly examined and read. An excel **`.xlsx`** database was created to capture a record of each article that was deemed to be eligible.[1] The article's critical meta-data was recorded in designated fields such as: journal source web link, journal name, journal title, authors, pages, timelines (day, month and year), editor, volume, issue number, city, country, continent, standard number, day accessed, month accessed, year accessed, data set, data set type, set of

[1]Accessible at: `https://github.com/Dandajena/SDA/`

data sources, dataset summary, research problem, research objective, implementation framework, architectural style properties, baseline models, best models, methodological approach, evaluation mechanism or criteria, evaluation metric, result comments, future recommendations and gaps comment. The tabled data fields were important for the empirical data analytic operations for visualisation, which would be introduced in Stage 5. In Stage 4 of the procedure, a total of 28 ineligible articles were removed because they did not directly relate to the goal of the literature review study. The database now had only 32 relevant articles, with the rest being removed. The `.xlsx` database file was further cleaned and converted to a `.csv` file.[1]

(e) **Stage 5**—a program was implemented to analyse the database—the cleaned `.csv` result of Stage 4. The analysis was visualized in the form of graphs, word cloud and graphical representations. Ultimately, the code, tools, and literature search database were published to GitHub, allowing other investigators unrestricted access to all experimental study materials and code. The GitHub website provides free access to the literature study database. All outputs, schematic diagrams were documented online.[2]

Finally, sequential datasets, pre-existing deep learning models and evaluation criteria were selected from 32 scholarly articles to be employed in the next steps.

4.2.3 Step 2—Exploration, evaluation and selection

This stage, shown in Figure 4.5, begins with a thorough examination of the research articles obtained in Step 1 in order to make a final selection of core rele-

[1]Accessible at: `https://bit.ly/3e9mgHy`

[2]All the experimental code is given in the Jupyter Notebook files on the GitHub website : `https://github.com/Dandajena/SDA/`.

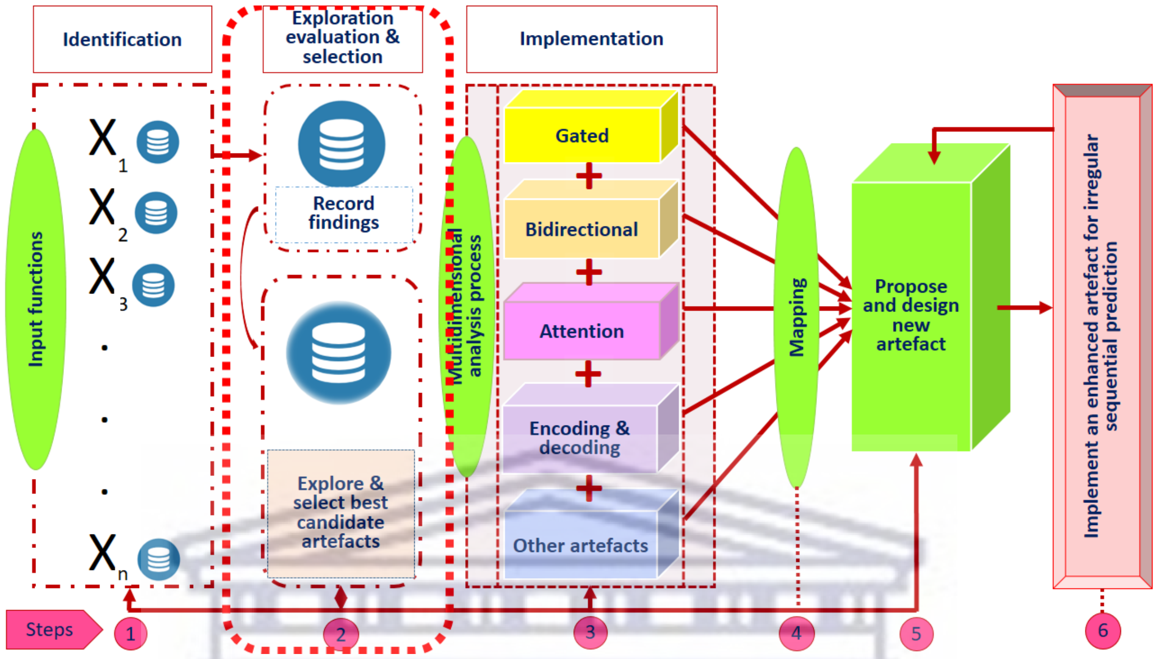

Identification
Exploration evaluation & selection
Implementation
Input functions
X1
X2
X3
Xn
Record findings
Explore & select best candidate artefacts
Multidimensional analysis process
Gated
Bidirectional
Attention
Encoding & decoding
Other artefacts
Mapping
Propose and design new artefact
Implement an enhanced artefact for irregular sequential prediction
Steps
1
2
3
4
5
6

Figure 4.5: Step 2—Exploration, evaluation and selection of the SeLFISA Framework

vant publications from which to extract applicable data, deep-learning models and assessment procedures. In these significant linked research studies, key topics highlighted in these publications were examined and appraised. This was done in order to create an ecosystem that included high-performance computer resources as well as software libraries and tools. Parts where inconsistency and ambiguity may affect future implementation processes and procedures were removed. A selection procedure was initiated, based on the following criteria:

1. **Datasets**—This step entails a thorough quantitative and qualitative analysis of the information gathered in Step 1. Combining descriptive numerical and graphical tools makes it easier to choose the right datasets and models. To determine the amount of discrete irregular sequential patterns in datasets, the following is considered while assessing the levels of irregularity in selected datasets:

(a) *Box and whisker plot*—using the *inter-quartile range* (IQR) outlier calculation, statistical descriptive analysis of abnormal patterns on all datasets is performed [97]. This method displays the dataset's minimum and maximum values. The box plot depicts the data's first, second and third quartiles, namely $Q1$, $Q2$, and $Q3$. The gap between the minimum and maximum values determines the range of the dataset. Finally, Equation 4.1 offers an IQR based on the difference between $Q3$ and $Q1$.

$$IQR = Q3 - Q1 \tag{4.1}$$

Outliers in irregular patterns can easily be detected as those data points that are either one and a half times IQR below $Q1$—in Equation 4.2 or above $Q3$—in Equation 4.3, i.e.,

$$\text{below} = Q1 - (1.5 \times IQR), \tag{4.2}$$

$$\text{above} = Q3 + (1.5 \times IQR). \tag{4.3}$$

(b) *Billauer's algorithm*—is utilized in [149] to validate the outcomes of the box and whisker plot results, as well as to detect local maxima and minima. It calculates the degree of irregularity in the original data environment using a graphical visualization analysis of peaks. The method is then tweaked to give *irregular pattern peak period detection* (IPPD) capabilities, which looks for values that are surrounded by smaller or bigger values for maxima and minima throughout the y-axis and matching x-axis to find discrete peak values. To provide the maximum number of discrete peaks, a look-ahead value for determining the look-ahead distance for a potential peak must be set as a specific value.

   (c) *Further exploratory data analytics*—is then conducted to gain insights into how variables in the chosen sequential dataset are connected to one another. This includes data description, data pre-processing, data crunching, data cleaning and exploratory data analysis.

2. **Artefacts or models**—after that, candidates are chosen based on their inherent design, application, context and authenticity. At this point, the focus must be on current issues related with common, contentious and contradictory issues raised by various authors. When there is a larger search space, factors like variable explosion must be carefully considered throughout the selection process of candidate models. The use of customisable clipping mechanisms is a potential solution.

3. **Evaluation**—deep learning models are chosen from a pool of resilient candidate models based on their performance reports. Models are chosen based on three key criteria (1) efficacy, (2) consistency and (3) efficiency. The *efficacy* of a model relates to how well it predicts the target variable on a test dataset. In general, many research articles assess the model's efficacy but only a few assess model consistency. *Consistency* is a measure of a model's resilience in terms of its ability to generalize data that was not seen during training. There is currently no metric that can quantify the consistency of models in order to compare them objectively as described in Equation 4.4.

   The time it takes to train a model in relation to its complexity is referred to as efficiency, with a shorter training time and lower complexity being preferable. *Efficiency* is a frequently disregarded criterion. When it is taken into account, it is usually done through timing assessments that ignore model complexity.

The SeLFISA framework includes a comprehensive examination of the models developed as a fundamental element. The first step is to choose an adequate set of criteria to assess accuracy. Two new measures have been created to quantify criteria for consistency and efficiency. Analyzing core research papers to determine a set of domain-specific metrics that can be used to evaluate model efficacy and consistency, is an important aspect of this step.

MSE, MAE and R^2 are the most often used metrics for regression efficacy. For consistency and efficiency, custom-defined metrics are used. For consistency it is based on MAE metrics and for efficiency it is based on model parametric magnitude and overall modelling time. When assessing model consistency, it is recommended that two separate testing sets be used. The first should be an unseen portion of the same dataset used for training, i.e. the *primary dataset* and the second of which is a different dataset with the same input and output as the primary dataset, i.e. the *validation* dataset. For the metric value of the primary testing set m_1 and the validation testing set m_2, each model created using the SeLFISA framework will be assessed with both testing sets to get a coherent efficacy metric value—assuming that there is reasonable variability across datasets. The *SeLFISA model consistency* C_s is defined by Equation 4.4 and is a measure to compare the two *different* metric values m_1 and m_2.

$$C_s = \frac{m_1 + m_2}{\alpha|m_1 - m_2|} \tag{4.4}$$

where $\alpha = 2$ averages the metric values m_1 and m_2. C_s provides the reciprocal of the normalized difference of the metrics m_1 and m_2 across the primary and validation datasets. The better the *consistency* C_s of the model across the two sets, and the *higher* the corresponding value of C_s gets, the closer the two metrics are. In general, greater C_s values indicate that the model is more effective in predicting values in a series.

The *SeLFISA model efficiency* E_s metric was created to quantify and compare the efficiency of the models under consideration. The goal was to benchmark models in terms of training time. A model with higher complexity might be regarded more efficient than a model with lower complexity when both models have the same time for training. To account for this, E_s in Equation 4.5 calculates the ratio of a model's total number of trainable parameters n to the time it takes to train the model t in seconds; the greater the metric value, the faster a model can train with a particular amount of learnable parameters.

$$E_s = \frac{n}{t}, \tag{4.5}$$

Given two models with the same number of trainable parameters, the efficiency E_s is directly dependent on the inverse of the training time. If two models have the same training time and one of the models has twice as many training parameters as the other, the E_s value of the first model is double that of the second, suggesting that it can train twice as many parameters in the same amount of time. In general, a higher E_s number suggests improved training efficiency. In other words E_s is directly proportional to n.

4.2.4 Applying Step 2 and results

The following is a step-by-step account of the implementation and results of the second step of the SeLFISA Framework, guided by the design principles outlined in Chapter 3.

1. **Datasets**—following on from the previous phase, 32 papers were found to be directly applicable to the current implementation in terms of the mod-

els proposed and the results achieved in these studies, out of a total of 60 publications reviewed [30]. These 32 articles were picked for execution and, maybe, selective inclusion into the improved model in a later step. At this point, all of the datasets discovered were gathered into a dataset bank [1]. A hybrid high-end computational processing environment provided by the CHPC was used as the experimental processing enviroment.

For this application, a total of 16 currency exchange rate datasets extracted from datasets of the 32 articles were analyzed and compared. Table A.1 in Appendix A on Page 134 contains a list of these datasets along with methods and evaluation metrics. The box and whisker plot and Billauer's technique, which emphasizes outliers in a dataset through IQR calculation as shown in Figure 4.6 on Page 63, guided the process of estimating the number of irregular patterns in a dataset. The number of irregular patterns in a specific sequence is used to calculate the overall number of outlier patterns.

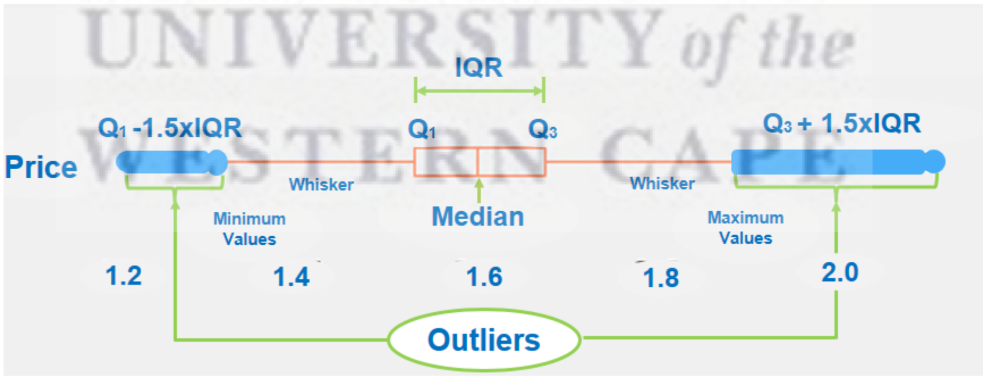


Figure 4.6: Box plot showing outlier distribution of daily exchange data between the GB Pound and the US Dollar from 1990 to 2016

Only three datasets contained the required number of irregular sequences.

[1]Accessible at: `https://github.com/Dandajena/SDA/`

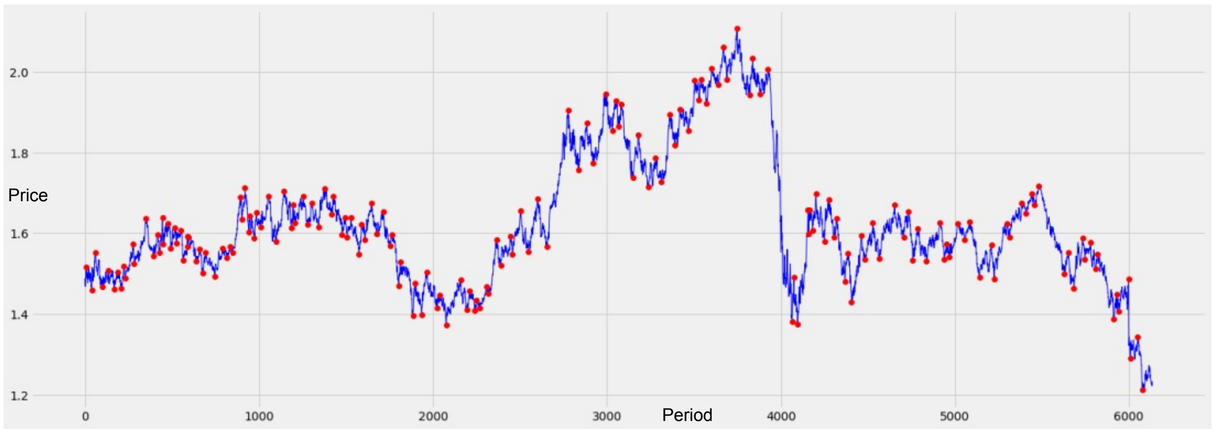


Figure 4.7: IPPD visualization analysis of daily exchange data between the GB Pound and the US Dollar from 1990 to 2016 [23]

The GBP/USD dataset had the most irregular sequences, 639 out of the 6135 daily records. The JPY/USD dataset was then examined, and 24 unusual sequences were discovered out of 5000 daily records. The GBP/USD dataset illustrated in Figure 4.7 was chosen as the primary dataset and JPY/USD as the validation dataset, respectively. These datasets were gathered and loaded into a database to be processed further.

2. **Models**—from the 32 publications reviewed, 12 deep learning models were discovered. These were used as candidate models—Table 4.1 on Page 97 contains the specific architecture of each of these models, as well as the scholars who proposed the architecture [112, 59].

   The models employ various combinations shown in the table of: GRU, LSTM unit, *one-dimensional convolutional layer* (Conv1D), *one-dimensional transpose of a convolutional layer* (Conv1DTranspose), *fully connected neural network layer* (Dense), Bi, *global attention mechanism* (Attention), *self-attention mechanisms* (SeqSelfAttention), *drop out layer* (Dropout), as well as the Keras TimeDistributed (TimeDist) layer which applies a given convolutional operation separately during each time step of a feature vector, [8, 41, 22] and the Keras RepeatVector layer which transforms a feature vector to a given desired output shape. Note that the following notation applies in the table [10, 106, 144]:

(a) Layer(q) specifies that q nodes were used in a given Layer.

(b) Layer1(q_1) + Layer2(q_2) + $\ldots$ + LayerN(q_N) implies a model with sequentially connected layers starting with Layer1, followed by Layer2 and so on, and terminating with LayerN, noting from the previous point that the number of nodes in these layers is $q_1, q_2, \ldots, q_N$.

3. **Evaluation**—three generally used model efficacy metrics, MAE, MSE, and the R^2 (coefficient of determination), were used together with the SeLFISA model consistency and efficiency criteria defined in Equation 4.5.

Description of most relevant SLR papers

This section reviews the most relevant SLR papers of the 32 seminal papers identified in Step 2 of the SeLFISA framework. These papers were selected because of their specific findings, methodologies, procedures and tools obtained by different authors in predicting sequential time series datasets. These papers all gave state-of-the-art results with regard to accuracy and efficiency but all did not conform to explainability, reproducibility and simplicity criteria.

Determining the impact of window length on time series forecasting using deep learning

Azlan et al. [8] investigated the effects of window length or time lags or forecasting horizon on financial stock market price prediction using a simple LSTM model. The research used the *Standard and Poor's 500* (S&P 500) daily closing price stock market dataset from Yahoo Finance with 4171 observations. By executing three experiments with distinct windows—25, 50, and 100 days—on the same univariate dataset, the studies were conducted in a desktop computing environment.

Accuracy of the model was used as a criterion for measuring model performance, and findings showed that a 25-day window length provided 81.25% performance accuracy, a 50-day window length had 62.5%, and a 100-day window length produced 100%. It is significant to note that this work introduced a window period as a hyper-parameter for sequential prediction model improvement. This study by Azlan et al. [8] showed that LSTM-based models offer good performance accuracy in financial forecasting tasks if the proper window length is established. To get better results, similar time series models can be employed and ranked using a wide range of evaluation criteria. This add value towards establishment of an approach for improving such models by examining their performance in a multidimensional manner.

Motorway Traffic Flow Prediction using Advanced Deep Learning

In a sequential time series prediction study, Mihaita et al. [90] applied LSTM based deep learning and other variants to predict the sequential flow of traffic on Sydney's motorways. The research suggested a framework for creating a deep learning model for traffic prediction in four steps: network identification, data profiling, feature generation, and trial-and-error deep learning model building. For the experiments, a sequential traffic dataset with 36.34 million data points or observations was used, which required a high speed computer environment (24 Intel cores). CNN, LSTM, and *backpropagation neural networks* (BPNN) were used individually or in combination in this investigation. The models needed between 10 and 15 epochs to converge, and the hyper parameter tuning procedure was restricted to batch sizes in the range (20; 30; . . . 100). RMSE, MAE, and *symmetric mean absolute percentage error* (SMAPE) metrics were used to evaluate models using prediction accuracy as a criterion.

The LSTM-based model performed better than other sequential models, although it struggled to predict on smaller datasets accurately. The proposed methodology by Mihaita et al. (2019) [90] offered a four-step process to direct the creation of sophisticated deep learning prediction models, as well as the selection of hyperparameters and assessment measures.

Improved Forecasting of Cryptocurrency Price using Social Signals

Glenski et al. [41] used sequential LSTMs and ARIMA models to look into ways to improve currency stock price prediction using social signals. The study used historical cryptocurrency price data (daily high, low, and price at market open and close) plus social media data from GitHub and Reddit to anticipate prices for three cryptocurrencies: Bitcoin, Ethereum, and Monero. RMSE, MSPE, MAPE, *max absolute percentage error* (MaxAPE), and RMSPE metrics were used to assess accuracy performance of these models. The historical price alone, the historical price and each social signal, and the historical price and combinations of each of the social signals were used as a method to train and evaluate the models.

The LSTM design that performed better than other models was made up of Dense (1), LSTM Layer (800), and LSTM Layer (400). Hyper-parameter tuning considered batch sizes (16, 32, and 64); learning rates [0.1, 0.01, 0.001, 0.0001, and 0.00001]; stopping call-backs with a maximum limit of 20 epochs; and combinations of LSTM layers ranging from 10 to 400-dimensional layers followed by 20 to 800-dimensional layers [41]. The best-performing LSTM model by Glenski et al. (2019) [41] had a batch size of 16, a learning rate [0.001, 400 and 800 units] for LSTM layers, respectively. The LSTM architecture reduced computation speed of the model. This work's methodology focuses on ways to assess the effectiveness of combining signals from social media into models that predict the future price of a cryptocurrency. According to this study, the complexity of the domain makes it difficult for present models to accurately predict financial currency price movements. Gated recurrent neural network models are regarded as good models for sequential tasks, despite the fact that this study did not adequately explain how the adopted models were chosen. Incorporating a more finely grained approach and taking into account such models in extreme scenarios are some of the recommendations of this paper—both of which could have positive effects if put into practice with a systematic implementation framework [41].

High-Performance Stock Index Trading: Making Effective Use of a Deep LSTM Network

The sequential time series prediction study by Chalvatzisa et al. [22] was aimed at creating a deep LSTM model with high sequential prediction and good trading performance (buy and hold strategy). In order to predict the price for the upcoming time period, the deep LSTM model took into account a limited amount of features and past time steps varying from 11 to 22. The performance of the proposed model was tested on four major US stock indices, namely the S&P 500, the DJIA, the NASDAQ and the *Russell 2000* (R2000) over the period of 8 year. Hyper-parameter tuning was performed using a grid search algorithm on a desktop computer, 1600 epochs, a rolling window training procedure, hidden units (32; 64; 128), input sequence length (11; 22; 44), dropout (0; 50; 70), input sequences of 11–22 time-steps, and 3–128 neurons.

Model accuracy was assessed based on different datasets using accuracy criteria based on *mean directional accuracy* (MDA), MSE, MAE, MAPE and R^2 metrics. The experiments obtained the best MDA = 50.02% on R2000, MAPE = 0.6% on DJIA, MAE = 9.01 on R2000, MSE = 142 on R2000 and R2 = 99.95% on NASDAQ. The suggested method for effectively using a deep LSTM network by adjusting various hyperparameters is a technical strategy for model optimization. However, the study was restricted to the gated architecture, attention mechanisms, and bidirectional mechanisms which was not fully accounted for by the methodology framework. The study also offered a wide variety of performance evaluation indicators to evaluate the models' accuracy. To offer design clarity and an explainable framework, it is necessary to broaden performance evaluation at various stages of implementation [22].

Multimodal deep learning for short-term stock volatility prediction

According to research by Sardelicha and Manandhara [112], a deep multimodal learning framework that generates a multimodal hierarchical network model was suggested to predict volatile short-term or daily stock prices of different industries. The hierarchical model demonstrated the impact of NLP news data for enhancing the process of forecasting stock movements.

This model was designed using LSTM, bidirectional and attention mechanism architectures which require more data during training. The attention layer of the design improved the results by paying attention to important volatile patterns during training. Its hyper-parameters were tuned using grid search which included mini-batch SGD, Adam optimizer and eight early stopping epochs to effectively predict volatility. Prediction performance was measured using accuracy with MSE, MAE and R^2 metrics. Experimental results demonstrated that the multimodal approach equipped with a global training mechanism outperformed a baseline econometric *generalized autoregressive conditional heteroscedasticity* (GARCH) model. The best results of the multimodal hierarchical network model were: R^2 = 0.455, MSE = 1.90E-05 and MAE = 2.82E-03 [112]. The choice of the adopted models namely a gated recurrent neural network, bidirectional mechanism and attention mechanism are good examples of sequential modeling architectures. However, this approach did not adequately address uncertainty in model parameters.

DSANet: Dual Self-Attention Network for Multivariate Time Series Forecasting

A study by Huang [59] developed a novel deep learning framework called the *dual self-attention network* (DSANet) for volatile multivariate time series prediction on a real-world sequential dataset—a 4 year daily revenue data of five gas stations. To predict sequential time series patterns, the hybrid DSANet model architecture combined local temporal convolution, a self-attention mechanism, and autoregressive components in a parallel technique. The model implemented in HPC outperformed the baselines models—with Intel i7-8700 CPU, GTX1060 GPU, 6 cores and 32 GB RAM.

The framework evaluated the performance of the model using accuracy criterion based on *relative squared error* (RRSE), MAE and *empirical correlation coefficient* (CORR). For RRSE and MAE, a smaller value is preferable, whereas a larger value is preferable for CORR. The GRU, *recurrent-skip layer* (LSTNet-S), *temporal attention layer* (LSTNet-A), *temporal pattern attention mechanism* (TPA) and DSANet produced better performance than other nine baseline models. This performance was demonstrated on an experiment of predicting complex time series datasets at different window lengths and forecasting horizon ranging from 32, 64, 128 and 3, 6, 12, 24 respectively. DSANet produced the best results in all cases: at window-horizon pair of 32-3—DSANet had RSSE = 0.781 and MAE = 0.407; at window-horizon pair of 32-6—DSANet had RSSE = 0.771 and MAE = 0.410; at window-horizon pair of 32-12—DSANet had RSSE = 0.829 and MAE = 0.436; and at window-horizon pair of 32-24—DSANet had RSSE = 0.927 and MAE = 0.442 [59].

Forecasting of Jump Arrivals in Stock Prices: New Attention-based Network Architecture using Limit Order Book Data

For predicting jump arrivals or direction in stock prices across five different NASDAQ stocks, Makinen et al. [85] proposed a state-of-the-art *CNN-LSTM with attention* (CNN-LSTM-A) model. This model architectural design provided capabilities to adapt and focus its forecasting attention on the most important features. To validate the performance of the CNN-LSTM-A model, additional baseline models with the following network architecture were also deployed on the same time series datasets. Architectural design of all implemented models is as below:

```
    (1) CNN-LSTM-A = [Input+Attention(SeqSelfAtt(32))+Conv1D(32)+Conv1D(MaxP)+
LSTM+(40)+Dense(40)+Dense(1)]
    (2) MLP = [input+Dense(40)+Dense(40)+Dense(1)]
    (3) CNN = [input+Conv2D(16)+Conv2D(16)+Conv2D(MaxP)+Conv1D(32)+Conv1D(32 filters)+
Conv(MaxP)+Dense(32)+Dense(1)]
    (4) deep LSTM = [Input+LSTM(40)+Dense(40)+Dense(1)]
    (5) Random classifier = [Input+Classifier]
```

Performance of these models was evaluated using precision, recall, F1 and *Cohen's Kappa* (CK) metrics. The CNN-LSTM-A produced highest average F1, recall and CK scores (F1 = 0.72, Recall = 0.8, CK = 0.62 and Precision = 0.66), the second best was a deep LSTM (F1 = 0.69, Recall = 0.66, CK = 0.60 and Precision = 0.73), followed by the CNN (F1 = 0.66, Recall = 0.66, CK = 0.55, and Precision = 0.66), and MLP (F1 = 0.72, Recall = 0.50, CK = 0.00 and Precision = 0.24). This study demonstrated how difficult it is to predict the stock price. By considering experimental procedure of this work it was not clear how they arrived at the results [85].

Leveraging Financial News for Stock Trend Prediction with Attention-Based Recurrent Neural Network

Using sequential financial news titles, Liu [81] developed a *deep Attention-based LSTM* (At-LSTM) model to predict the directional movements of the S&P 500 index and individual company stock prices. The model's self-attention mechanism was used to distribute attention on the most relevant words, news, and days. Gated RNN was used to encode the news text and capture context information. Reuters news data for 473 S&P 500 listed companies was collected over a five-year period, and the historical stock price data for each individual share of the S&P 500 was gathered from Yahoo Finance from 2006 to 2018. Experimental baseline models included SVM, CNN-LSTM, *embedding input and CNN* (EB-CNN), *knowledge graph and CNN* (KGEB-CNN), *At-LSTM without sentence encoder* (Bag-At-LSTM), *At-LSTM without character level composition* (WEB-At-LSTM), *At-LSTM with news abstract* (Ab-At-LSTM), *At-LSTM with news document* (Doc-At-LSTM) and *At-LSTM with technical indicator* (Tech-A (KGEB-CNN)).

When compared to the baseline state-of-the-art models, the At-LSTM model demonstrated promising and competitive performance. On each individual prediction of the price of an S&P 500 stock, it had an accuracy of 70.36% on average and a maximum accuracy of 72.06%. In terms of multi-dimensional performance evaluation of the model to validate its performance, the implementation framework's clarity was not fully demonstrated during the experiments [81].

Autoregressive Convolutional Recurrent Neural Network for Univariate and Multivariate Time Series Prediction

In 2019, Maggiolo and Spanakis [84] proposed an auto-regressive convolutional recurrent neural network model for univariate and multivariate time series prediction on 4 different sequential time series datasets, including two univariate datasets (daily values for Melbourne's Minimum temperature and Zurich sunspot) and two multivariate datasets (Energy production for 10 different photovoltaic power plants and SML2010). The three components of the model architecture design were: (a) a multi-scale, convolutional part to extract features from the input TS; (b) a recurrent part with three GRU units to encode the sequence; and (c) an auto-regressive part.

Three metrics, including the MSE, MAE, and *dynamic time warping* (DTW), were used to evaluate model performance based on prediction accuracy. The suggested model appears to perform 50% better than generally accepted baselines. These baseline models were namely the ARIMA models, *support vector machines* (SVM), simple LSTM, Deep GRU and LSTNet. This demonstrated the strength of models with a hybrid architecture in complex sequential time series prediction. The best results of the proposed model on multivariate energy were MSE = 10.1 and MAE = 1.824 then MSE = 8.5 and MAE = 1.061 on a multivariate SML 2010 dataset. It is also significant to highlight that when applying this model to predict temperature and sunspot datasets, the proposed model was outperformed by ARIMA and LSTNet. This leads to disagreement and inconsistency in how the results are interpreted [84].

Comparison of Deep Learning Models on Time Series Forecasting: A Case Study of Dissolved Oxygen Prediction

Utilizing a 4-year real-time oxygen time series data in dissolved water from China's Yangtze River Basin, Qin (2019) [106] compared sequential deep learning models for time series prediction. A training set of 90% of the data, including 64411 observations, was used. RMSE, MAE, and R^2 metrics were used to evaluate the performance of the model. In this work, better performance was demonstrated by values that are nearer to 0 for RMSE and MAE and nearer to 1 for R^2. The GRU-based model performed significantly better than competing baseline models including CNN, *temporal convolutional network* (TCN), LSTM, and *bidirectional recurrent neural network* (BiRNN).

Models were created utilizing the Adam optimiser. Units, batch size, and validation split were set to 50, 128 and 0.1 for LSTM, GRU, BiLSTM, and BiGRU models. In each time step, GRU produced the highest average R^2. The LSTM model performed the worst since it had the highest RMSE, MAE, but the lowest average R^2 values. In order to avoid increased error rates during training, this study advised researchers to pay close attention to time during training [106].

Deep Equilibrium Models

A study by Bai [10] presented a novel the *deep equilibrium model* (DEQ-Transformer) for modelling sequential data. The model was applied on the large-scale high-dimensional sequential WikiText-103 benchmark, which contains over 100M words and a vocabulary size of over 260K, and a smaller *Penn Treebank* (PTB) corpus, with 888K words at training and a vocabulary size of 10K,. The work demonstrates how the DEQ-Transformer can be applied to two state-of-the-art deep sequence models: self-attention transformers and trellis networks. The DEQ-Transformer equipped with a good temporal memory retention mechanism produced improved performance accuracy over baseline state-of-the-art models such as the *trellisnet* (TCN), LSTM and GRU. Due to an analytical backward step, the deep equilibrium model uses substantially less memory than traditional deep nets. It vastly reduced memory consumption by 88% memory reduction which is often one of the efficiency bottlenecks for training large sequential models [10].

Specific issues that came to the fore during SLR

Based on the outcomes of these 32 seminal papers of Step 2 of the SeLFISA framework, it is clear that our daily lives are largely dependent on deep learning tools and systems in various fields such as health, education, business, and socio-economic services [108]. The exponential growth in data volume and dimension has created new problems that call for innovative and effective approaches, particularly in the prediction of complicated sequential time series datasets [51]. It is difficult to predict the stock price with enhanced model performance, as this is a difficult area [29].

The M1 to M4 Competitions by Makridakis and Hibon, the Sante Fe Competitions by the Santa Fe Institute, the *Knowledge Discovery and Data Mining* (KDD) cup competitions by the Association for Computing Machinery's Special Interest Group on KDD, the Kaggle time series competitions by Goldbloom, the Global Energy Forecasting Competitions by Tao Hong, and the International Journal of Forecasting include some of the most influential competitions that have influenced sequential time series prediction [142]. For time series prediction, a number of cutting-edge deep learning models, including stable LSTMs and their hybrid variants, have been largely proposed [51, 103]. It is worth noting that the underlying data-driven technologies [8, 10, 22, 41]—algorithms and models—were created by different designers with different and conflicting objectives [59, 81, 84, 90, 106, 112].

The quantity of scientific and media publications about artificial intelligence is increasing rapidly. Existing research on deep learning frameworks for sequential modelling has resulted in a substantial body of publications that cover a wide range of subjects, including methods, experimental design, optimization techniques, input signal issues in the form of datasets, application domains and theories. However, these studies have numerous problems, constraints, and contradictions as well as strengths and opportunities. When developing and examining deep-learning frameworks, researchers undoubtedly have a variety of goals in mind. In most cases, the interest of the designer does not necessarily align with those of the users or readers [121, 145]. This triggers the need for a systematic implementation approach that eradicates elements of confusion.

The following list some critical issues:

1. **Literature inadequacy, contradiction, and inconsistency with one another**—no systematic strategy or process with refined steps, that can be used to improve the design of state-of-the-art models ethically, could be found in the current scientific literature [21, 69, 74]. Numerous sequential prediction models are being pitched as ”robust” or ”state-of-the-art” by various authors under various conditions, however, they lack adequate evidence in terms of implementation details and procedure. There is inadequate demonstration on how decisions were made at various levels to arrive at an enhanced deep learning model, understandable to researchers, general users, and other stakeholders. Predictive performance challenges associated with existing models are conflicting since they lack traceable, well-established and explainable literature for sequential prediction on the same implementation platform [141]. This causes uncertainty within the body of deep learning knowledge. Thus there is a need for systematic guidelines to shape deep learning models through an implementable framework.

2. **Design and implementation complexity**—A wide range of designs and arrangements of deep learning models have been suggested to date. New performance capabilities in predicting financial time series are provided by hybrid architectures that combine the strengths of multiple deep learning models. Most models with higher performance or enriched efficacy—above current state-of-the-art models—are typically developed through random or ad-hoc design strategies with no clarity on design and implementation mechanism [5, 94, 113, 142]. These trial-and-error methods

have led to high levels of model design bias, poor transparency [23] and non-interpretability [74]. This is also driven by high levels of uncertainty which exist from model parameter selection that best explains the selection of deep learning models to the interpretation of results. This compromise levels of understandability of a deep learning models. Understandability is often inversely proportional to its prediction accuracy, i.e. the better the prediction accuracy is the lower the model understandability [152]. The bulk of existing deep learning approaches are classified as black boxes since they do not provide enough explanation of how these models function or are derived [3]. This raises issues of bias, suitability and transparency that reduce the chances of attaining human trust in AI systems [129].

3. **Lack of multidimensional performance analysis**—while the majority of earlier research has been done to improve the performance of AI models based on general accuracy and efficiency [59, 81, 84, 90, 106, 112], but understandability is rarely considered as part of the multidimensional performance evaluation criteria [23, 103, 72]. The extent to which the result of a measurement, calculation, or specification agrees with some benchmark, or standard is not consistent in most literature [8, 10, 22, 41].

4. **Inadequate transparency, repeatability and understandability**—regarding the procedures used to create models, produce data sets, and explain the results has remained an issue within the field of deep learning. There is a trade-off between achieving algorithm explainability and maintaining performance robustness [152]. Existing seminal readings have limited ease of reproducing the same result, they do not provide adequate distinct implementation that any researcher in any domain can follow and use to improve state-of-the-art prediction models. The majority of these follow an ordinary machine learning pipeline framework that

struggles to provide step by step implementation and evaluation of process and outcome. Hence, this work argues that a state-of-the-art framework cannot be said to be optimal without a comprehensive explanation. The path to explainable models begins with a simplified description with systematic evaluation mechanisms embedded at various levels. Such an approach ensures trustworthiness when such systems are deployed [7]. This also compromised straightforwardness, or simplicity of the implementation process, or the ease of understanding as opposed to a research by Dignum (2018) [36] which indicated that the understandability of AI systems should be a priority of any deep learning development [42].

5. **Prediction inefficiencies**—where most state-of-the-art deep-learning frameworks [33] experience efficiency performance problems when exposed to different sequential datasets [136, 142]. Because of stochasticity of features especially in financial stock price datasets characterised with irregular patterns [37, 105]. Sequential stock price datasets require extensive analysis resources [151]. There is a wide deficiency of traditional deep learning models for capturing nonlinear or dynamic dependencies between time steps and between multiple time series [59]. Lack of accurate, reliable, and interpretable modern deep-learning models for uncertainty estimation over continuous variables [74]. This further demands ever-growing requirement of computing power, time and resources particularly when predicting unstable extreme sequential weather patterns [24].

In summary, to the best of our knowledge this step of SeLFISA framework has revealed that—recent research has primarily demonstrated that deep learning model selection and design are largely trial-and-error procedures, with researchers adopting tools and methodologies based only on randomization and then applying them to specific circumstances. The selection of a particular dataset, computing environment evaluation criteria and their corresponding metrics, network architecture, hyperparameters, evaluation criteria and their respective metrics, and many other decisions are not supported by a sound scientific theory or methodology. The majority of sequential deep learning results lack of transparency, interpretability and clarity on the methods adopted for designing models, generation and explanation of data sets and results. There are a number of challenges associated with deep learning, no single ideal framework addresses the behaviour associated with such time series, and top-ranked baseline algorithms and models have limited methods for evaluating performance at different levels of implementation. The gap extends to no comprehensive approach that encompasses the varying stages of design, implementation and evaluation. There is room for exploring better ways to design sequential frameworks with potential to systematically produce models with better performance in understandable way. The `.csv` file.[1] file on GitHub contains metadata along with a description of each of the 32 selected articles. This file also provides a detailed examination the above-mentioned scientific gap towards the prediction of sequential time series patterns.

4.2.5 Step 3—Implementation

In this step, the training component of the primary dataset is used to implement and train selected candidate deep learning models found in the core research papers. Algorithm 4.1 summarizes the steps involved in completing this task. The procedure divides the primary dataset into training and testing sets using an appropriate ratio and then applies the model to the validation dataset to compute the SeLFISA consistency criterion, as explained in the previous step.

Lines 4–11 of the Algorithm 4.1 are used to train the model of the training set, generate testing results on the testing portion of the training dataset and gen-

[1]Accessible at: `https://bit.ly/3e9mgHy`

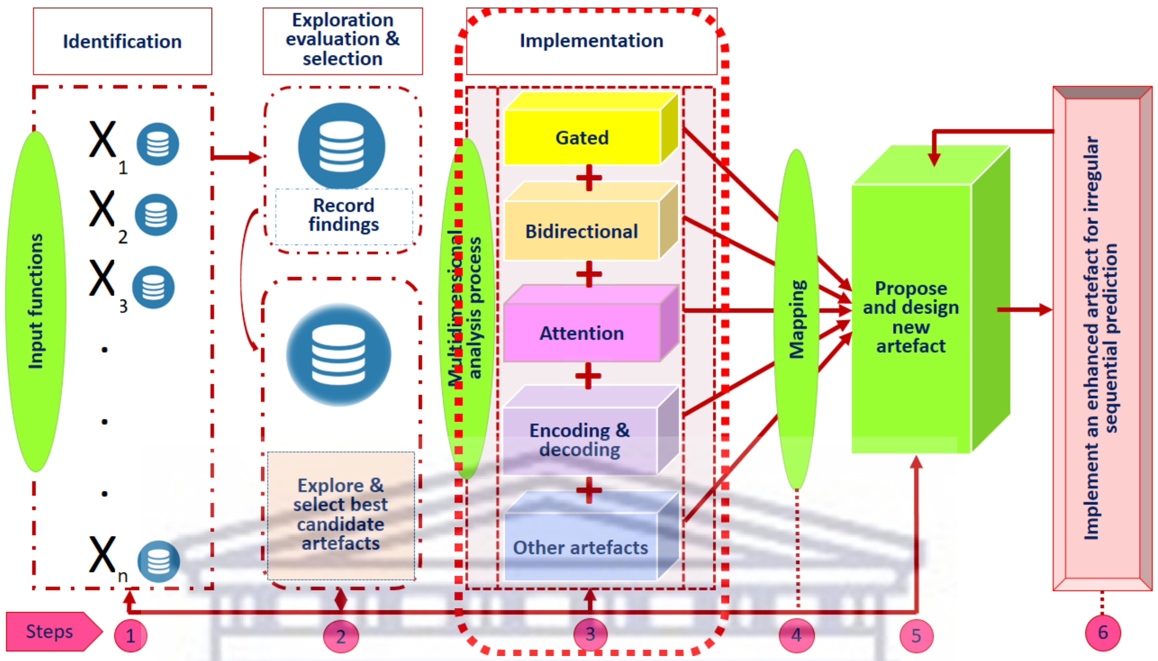

Identification
Exploration evaluation & selection
Implementation
Input functions
X1
X2
X3
Xn
Record findings
Explore & select best candidate artefacts
Multidimensional analysis process
Gated
Bidirectional
Attention
Encoding & decoding
Other artefacts
Mapping
Propose and design new artefact
Implement an enhanced artefact for irregular sequential prediction
Steps
1
2
3
4
5
6

Figure 4.8: Step 3—Implementation of the SeLFISA Framework

erate testing results of the validation set for each of the models found in Step 2 of the SeLFISA framework. To minimise generalization mistakes, a hyper-parameter regularisation strategy is used, which involves randomly dropping out or changing internal neurons and their connections during training [127]. These findings are then collated and compared in order to identify and the pick N top-performing models that will serve as the foundation for the next phases in creating an improved model. In the following steps, the top-performing N models was referred to as the *baseline models*.

Algorithm 4.1 Implementation of models from core research articles and selection of baseline models in line with Figure 4.8

```
Inputs––primary dataset split into training, and testing subsets; Validation dataset

For each model resulting from Step 2 of the SeLFISA framework:
    Get summary of model (design structure; total number of parameters)
    Set monitoring and regularization function
        to avoid overfitting (monitoring accuracy metric values)
    Fit model to training set using parameters recommended in articles
    Plot training performance loss to monitor training / validation loss
    Use trained model to generate predictions on the training set
    Evaluate training performance accuracy metrics

    Generate testing results on the testing subset:
        Use trained model to generate predictions on the testing set
        Record prediction accuracy metric results
        Visualise / Plot prediction results along with ground truth

    Generate testing results on the unseen validation dataset:
        Use trained model to generate predictions on the validation set
        Record prediction accuracy metric results
        Visualise / Plot prediction results along with ground truth

    Save the model as .tf file
    Save output results as .csv file

Tabulate all results observations
Analyse the results
Select top N performing models as baseline models based on accuracy
        metric values on the testing portion of the primary dataset.
Compute efficiency and consistency of baseline models
```

4.2.6 Applying Step 3 and results

The following is a step-by-step account of the execution and results of the third step of the SeLFISA Framework, guided by the design principles outlined in Section 3.2.3.

Algorithm 4.2 Implementation of models from core research articles and selection of baseline models

```
Inputs: GBP/USD for training 80% and testing 20%;
        JPY/USD for validation,

For each model resulting from Step 2 of the SeLFISA framework:
    Get summary of model (design structure; number of parameters)
    Set monitoring and regularization function
         to avoid overfitting (monitoring MSE rates)
    Train the model on the training set
            (epochs = 50; batch size = 50; validation split = 0.2; learning rate = 0.1)
    Monitor training performance using plots (training loss; validation loss)
    Use trained model to generate predictions on the training set
    Evaluate training performance
                using accuracy metrics (MAE; MSE; Adj. $R^2$ Score)

    Generate testing results on the testing dataset:
        Use trained model to generate predictions on the testing set
        Record accuracy prediction metric results (MAE;MSE; Adj R2 Score)
        Visualise / Plot prediction results along with ground truth

    Generate testing results on the validation dataset:
        Use trained model to generate predictions on the validation set
        Record accuracy prediction metric results (MAE; MSE; Adj. $R^2$)
        Visualise / Plot prediction results along with ground truth

    Save the model in a robust serialisable format
    Save output results as .csv file

Tabulate all results observations
Analyse the results
Select top 4 performing models as baseline models using accuracy
         metrics (MAE; MSE; Adj. $R^2$) on testing set
Compute efficiency and consistency of baseline models
```

The selected candidate models were implemented and trained on the chosen training dataset in this step. The implementation code can be found online[1].

The GBP/USD and JPY/USD datasets were employed as the primary and vali-

[1]Accessible at: `https://github.com/Dandajena/SeLFISA_Paper_2021_F`

dation datasets, respectively, as seen in Algorithm 4.2. The GBP/USD dataset was divided into two subsets: training and testing, with the training subset accounting for 80% of the total. For validation, the JPY/USD dataset was used. The Open, High and Low prices of all time steps inside a shifting window of a particular duration were utilized as input towards predicting the Close price of the 100th time step using a feature vector created from the datasets for training/testing. The shifting window was set to 100 pixels wide, as recommended by Azlan et al. [8]. This feature vector format was utilized to train all models in this implementation, including both the candidate models in this stage and the upgraded candidate models in the next step.

The training set was used to train the 12 models that resulted from the previous stage, and the testing set was used to evaluate them. With a batch size of 50, a validation split of 0.2, and a learning rate of 0.1, training was completed over 50 epochs. The training environment was a hybrid high-end computational processing environment offered by the CHPC.

The efficacy measures, such as MAE, MSE, and R^2, as well as the consistency and efficiency metrics provided in the preceding phase, were computed and tabulated for each model once it was trained. Models 2, 3, 4, and 7 were chosen as baseline models for the following stage since they had the best performance among the twelve models. Table 4.3 summarizes the outcomes of these baseline models.

1. The *Model* column refers to models in Tables 4.3.
2. The next six columns are divided into two groups, each of which summarizes the model accuracy findings for each dataset in terms of the three separate accuracy metrics used, namely the MAE, MSE, and adjusted $R2$. The results of the JPY/USD dataset are particularly informative of the generalization potential of each model, given that it was completely

unobserved throughout training and optimisation.

3. The next two columns group the *Training Efficiency* of the models and provide training time of each model, and the *SeLFISA efficiency* metric E_s based on Equation 4.5.

4. This is followed by the *SeLFISA consistency* column which computes the consistency Cs of each model according to Equation 4.4.

5. Two values are highlighted in each relevant column for comparison: the value corresponding to the best-performing baseline model and the value relating to the SeLFISA model.

Model 3 had a noticeably lower MAE and R^2 than the other models, while Model 7 had a marginally lower MSE than other models, as shown in Table 4.3. An examination of the error on the training set, i.e. GBP/USD training set, uncovered that the four baseline models had mostly comparable results, although Model 3 had a substantially lower MAE and R^2 than the other models, and Model 7 had a marginally lower MSE than the other models. In comparison with the MAE, the MSE is more susceptible to big outlier errors. This means that, while Model 7 has fewer outlier deviations, Model 3 is more accurate overall. Model 3's R^2 is higher than that of the other baseline models.

As previously stated, using a new dataset for validation purposes, such as the JPY/USD dataset, helps to provide a far more realistic reflection of the resilience and wide applicability of a particular model. While it is essential for a model to successfully predict using the datasets that it was trained on such as currency prediction, it is also valuable for the model to expand to be extendable. Model 4 emerged as the best of the four baseline models, with its MAE and MSE smaller than that of any of the other models [146], and R^2 is greater, see Table 4.3 on Page 102. Models 4's consistency value in the table is at least three times greater than that of the other models. Furthermore, Models 2 and

7 did not perform well with the new dataset, resulting in negative adjusted R^2 values. Other datasets were not used with these models.

Models 2, 3, and 4 were similar in terms of efficiency, with Model 2 being the most efficient of the three, while Model 7 was the least efficient of the three. A comprehensive examination of the results can be found in Section 4.2.11 where the results are discussed, it should be noted that the majority of the models used at this stage gave predictions that did not coincide with the ground truth values.

Figures 4.9 and 4.10 show a visual example of a high vertical and horizontal displacement between the ground truth and the prediction of a given arbitrary model. A horizontal shift indicates the presence of a prediction delay, whereas a vertical shift has a quantitative influence on each prediction. Horizontal shifts generate mistakes in the forecast direction at data points with large price inflections. An ideal model for solving such a problem would be one that can preserve the same vertical magnitude while shifting the horizontal dimension backwards to provide insight into future patterns before they occur.

A second casual observation is that gated LSTM-based models outperformed GRU models because the LSTM architecture contains more internal gates, such as input, output, and forget gates, whereas GRU models by [26] have reset and update gates. To accurately anticipate future trends, the LSTM models use linked features stored in the massive memory architecture at their disposal. As a result, the performance of the GRU models, which have fewer gates, is superior in terms of performance efficiency [72].

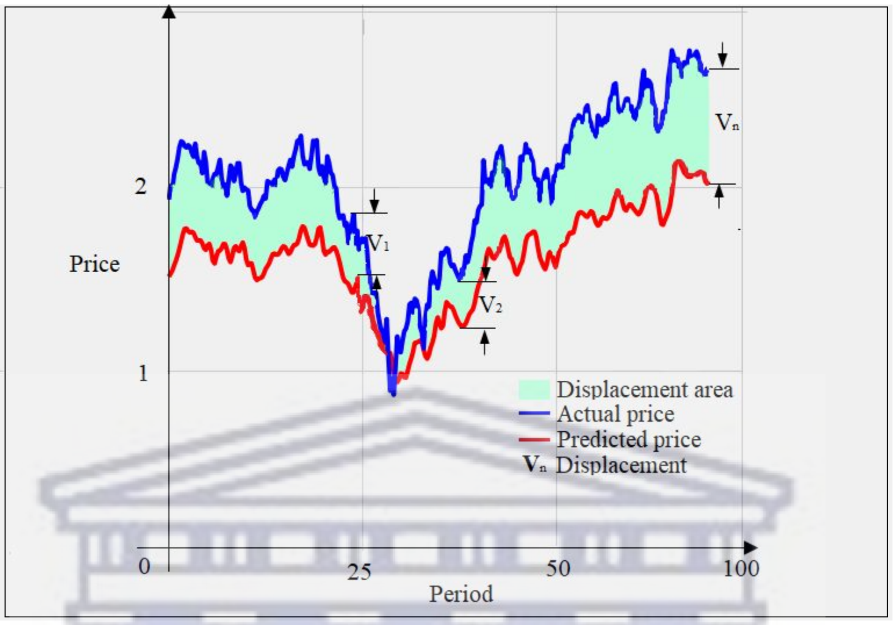


Figure 4.9: An illustration of the vertical displacement V_n from the ground truth prediction analysis of irregular patterns

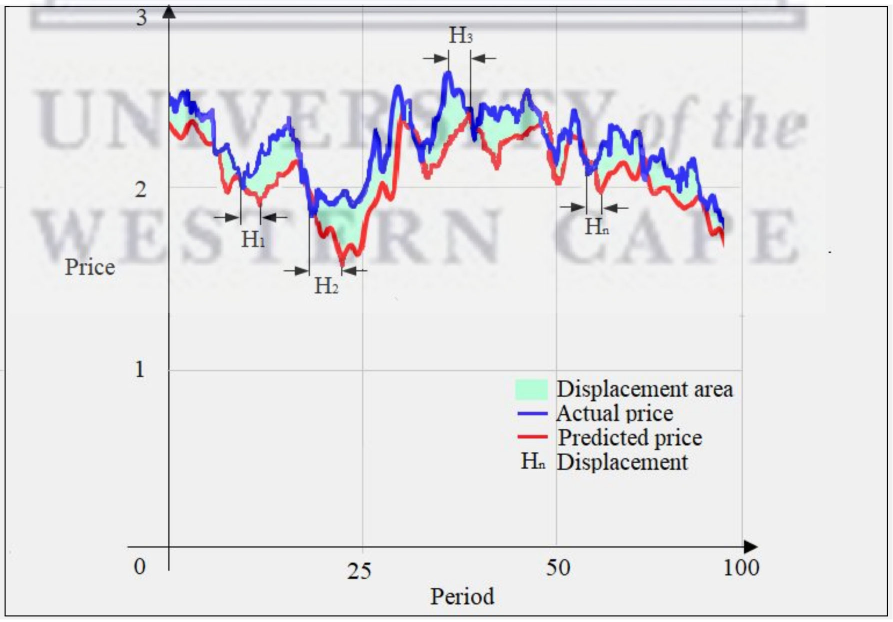


Figure 4.10: An illustration of the horizontal shift H_n from the ground truth prediction analysis of irregular patterns

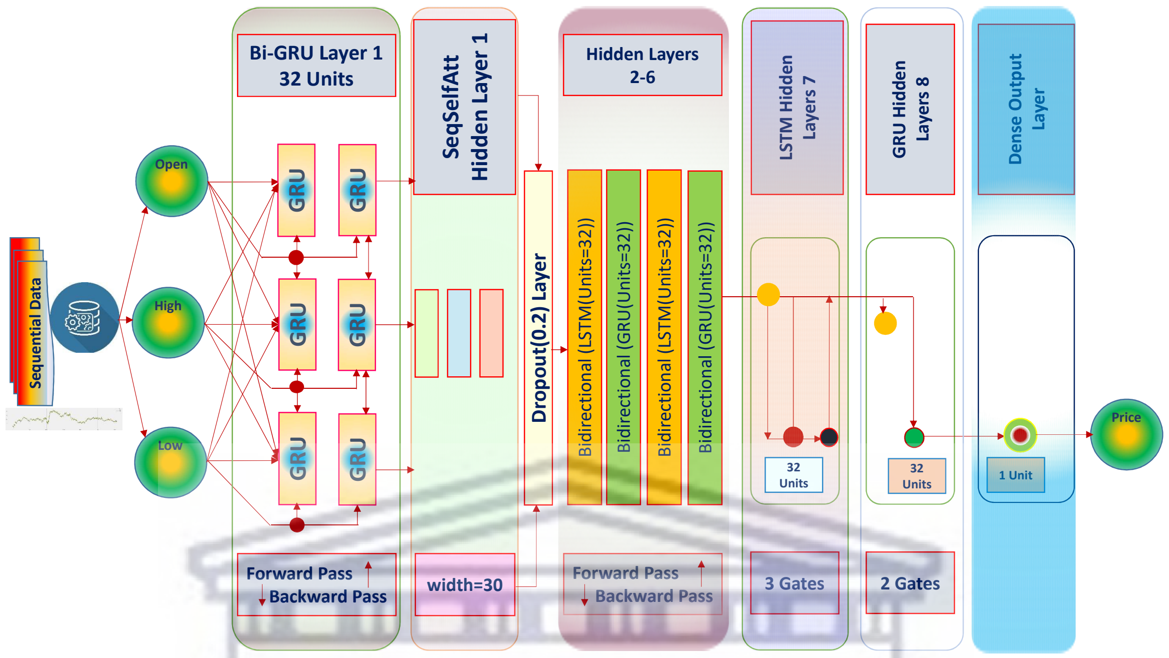


Figure 4.11: SeLFISA Deep learning model:
BiDirectional(GRU(32 Units)) + Attention(SeqSelfAtt(30)) + Dropout(0.2) + Bi(LSTM(32)) + BiDi(GRU(32)) + Bi(LSTM(32)) + Bi(GRU(32)) + LSTM(32) + GRU(32) + Dense(1)

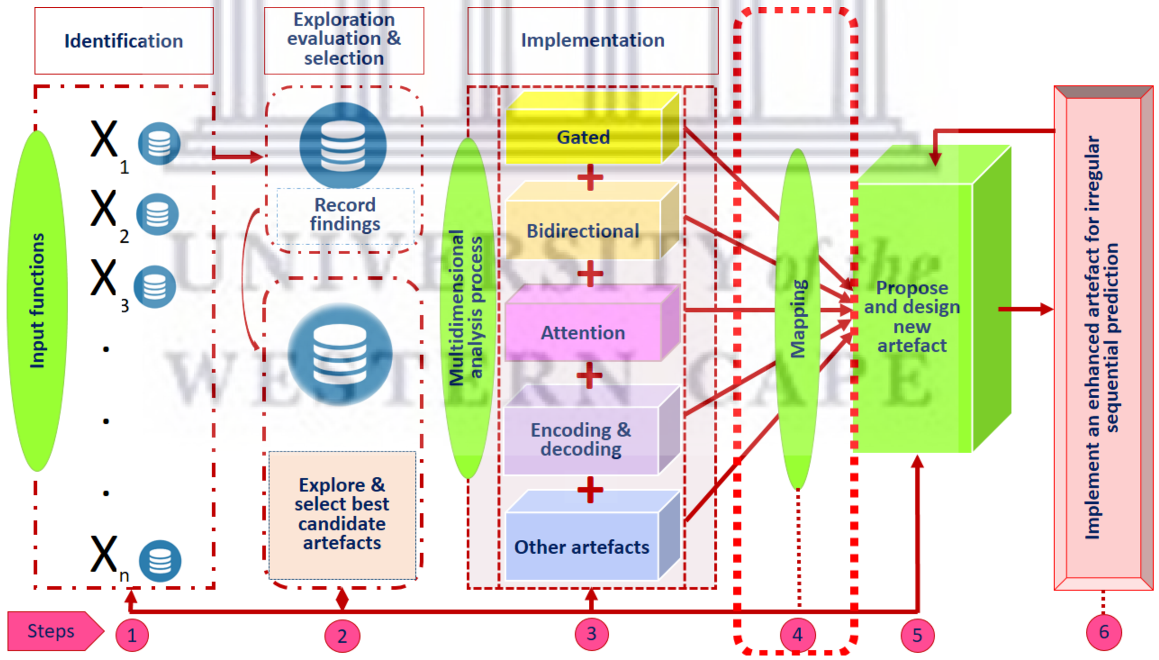

Identification
Exploration evaluation & selection
Implementation
Input functions
X 1
X 2
X 3
X n
Record findings
Explore & select best candidate artefacts
Multidimensional analysis process
Gated
Bidirectional
Attention
Encoding & decoding
Other artefacts
Mapping
Propose and design new artefact
Implement an enhanced artefact for irregular sequential prediction
Steps
1
2
3
4
5
6

Figure 4.12: Step 4—Mapping of the SeLFISA Framework

4.2.7 Step 4—Mapping

This stage depicted in Figure 4.12 entails compiling and tabulating all variables and intrinsic elements connected with the different ways in which the baseline models' architectures might be employed to build an improved model, as a culmination of the previous three steps.

A total of 29 variables in Table C.1 of Appendix C summarizes the variables identified—that can be considered, selected and manipulated in order to obtain an enhanced predictive model using the existing models were identified in Step 3. Table C.1 of Appendix C summarizes the count and listing of features that each variable may have in the *Feature Count* and *Feature List* columns. The features are aspects of each variable—options to select from and activities to carry out. For example, for the 6th variable in Table C.1 *Implementation Languages*, there are five possible options to use namely Python, Java, Lisp, Prolog and R. The 29 variables provided in the table are generally applicable to any deep learning domain of research. The feature list for some variables are applicable to a variety of domains, e.g. Variables 13 and 14 which are general data manipulation and splitting procedure applicable to any domain.

On the other hand, the feature list of some variables in the table may vary according to the domain of application, e.g. the feature list for Variable 4 in the table—*Existing research sources*—, will likely vary depending on the domain of research and the available papers at a given time. Compiling such a table is a principal part of this step as it serves as a guide towards the design and optimisation of the enhanced model. It provides a way of managing and pruning the number of variables to consider and optimize based on strategic importance and feasibility, since these variables may be considered to be hyperparameters of the model.

Table C.1 also includes a classification of each variable into one of two categories *combinatorial* or *permutative* [110]. Permutative features follow a fixed order, whereas combinatorial variables are unaffected by the order, placement, or organization of the process. The final column of Table C.1 contains a brief comment on each variable's implementation.

After the table has been created, a manual approach is used to identify features that may be important in achieving an upgraded model. To generate and improve a new design, key parameters and hyper parameters of relevance are selected, with specific attention to one performance metric at a time. The following phase involves using permutations and/or combinations of these factors that have the potential to increase performance.

4.2.8 Applying Step 4 and results

The following is a step-by-step account of the implementation and results of the fourth step of the SeLFISA Framework, guided by the design principles outlined in Section 3.2.3. Step 3 gave rise to ideas for a better model for discrete irregular sequential prediction, see Figure 4.8 on Page 79. This stage produced a thorough list of classified variables and hyper-parameters, which was used as a knowledge foundation for the next step (see Table B in Appendix:B).

4.2.9 Steps 5 and 6—Propose, design and implement a new model

Step 5 of the SeLFISA framework is for conceiving a new model, whereas Step 6 is for implementing it. It's worth noting that Step 6 loops back into Step 5, emphasizing that the two are inextricably linked. In reality, the process entails proposing and designing a new model based on insights acquired in prior steps and/or earlier iterations of Steps 5 and 6 shown in Figure 4.13; and executing and testing the new design to see if it outperforms the baseline models.

The goal is to iteratively reach a new neural network architecture with im-

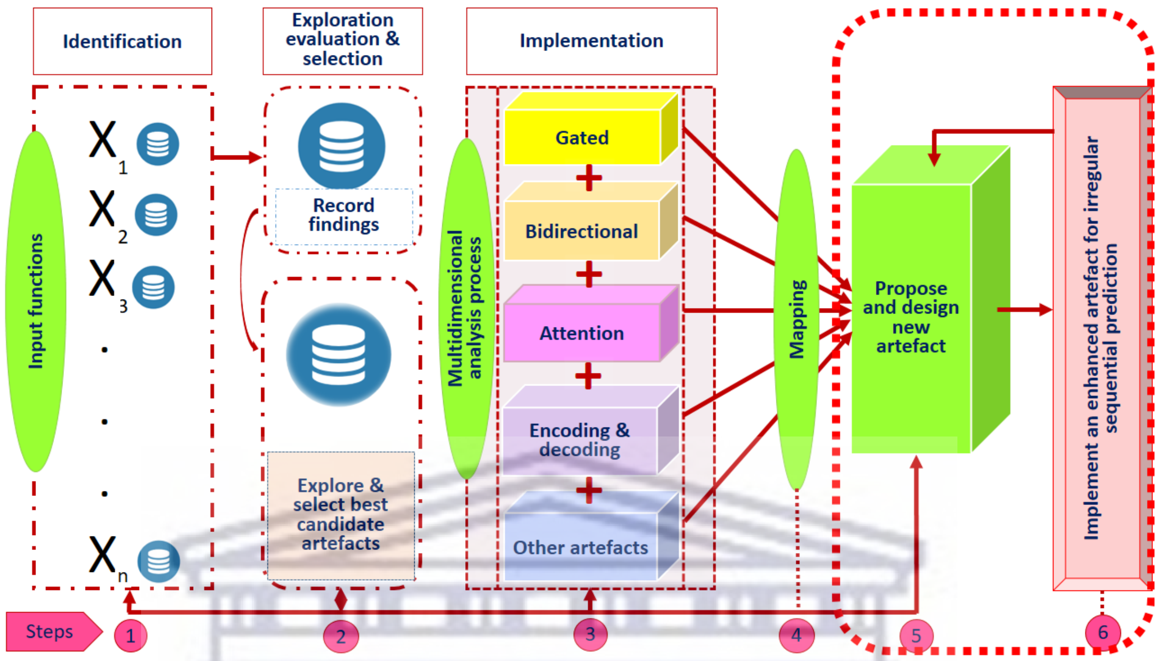

Identification
Exploration evaluation & selection
Implementation
Input functions
X_1
X_2
X_3
X_n
Record findings
Explore & select best candidate artefacts
Multidimensional analysis process
Gated
Bidirectional
Attention
Encoding & decoding
Other artefacts
Mapping
Propose and design new artefact
Implement an enhanced artefact for irregular sequential prediction
Steps
1
2
3
4
5
6

Figure 4.13: Steps 5 and 6—Propose, design and implement a new model using the SeLFISA Framework

proved performance depending on the outcome of the prior steps and the insights acquired from them. The new architecture takes advantage of the baseline models' best features. Algorithm 2 gives a rundown of the steps involved in creating a better model.

There are two parts to the algorithm. The baseline models' important combinations are identified in Part 1. The architectures of the models in these combinations serve as a design reference for future models. In each case, the new model is trained and then tested on the testing set to produce prediction values that are displayed against the ground truth, as well as accuracy measures.

Part 2 of the approach entails assessing whether any of the new models created in Part 1 surpass the baseline models, and if so, selecting the best new model as the final improved model. If not, the operation is repeated, this time using the newly learned insights and knowledge about the architectures of all baseline and new models. This is repeated until a better model is obtained or a certain number of iterations has been done.

Algorithm 4.3 Algorithm to arrive at an enhanced deep learning model

```
Inputs: Baseline models resulting from Step 3;
        primary dataset split into training set and testing set;
        validation dataset.
Part 1—Construct initial models based on the baseline models
Identify key combinations $K$ of baseline models in Step 3 of the SeLFISA
        framework to enhance performance
For each combination $M$ in $K$:
    Use the design of the models $M$ as a basis for a new design $EM$
        placed in set $E = \{EM : M \in K\}$ of enhanced models
    Train the model $EM$
    Use $EM$ to generate predictions on the testing set
    Record accuracy prediction metric results for $EM$
    Visualise / Plot prediction results along with ground truth

Part 2—Construct derivative models using
         baseline models and new models from Part 1
While performance of any model in $E$ is not significantly better than baseline
      models and iterations $< N$:
    Execute Lines 5–11 of Algorithm 1, with $M$ set to baseline models and
           models in $E$ to create a new derivative model $EM$
    Append $EM$ to $E$

    Select the highest–performing $EM$ in $E$ as the final enhanced model
    Apply optimisation to $EM$ on additional hyperparameters, i.e. layers,
         activation functions, ordering, etc.
    Apply grid search optimisation to $EM$ on additional hyperparameters,
         i.e. learning rate, dropout rate and batch_size.
```

This process gives an organized method of working towards developing an initial upgraded model given enough training data and a reasonable number of repetitions. Once an upgraded model has been identified, it is subjected to a number of optimization methods in order to further refine it and improve its performance. The initial optimisation method in Line 24 of Algorithm 4.3 on

Page 91 focuses on examining and selecting the optimal layer configurations and activation functions progressively. The second optimisation procedure in Line 26 of Algorithm 4.3 on Page 91 uses grid search to find several training hyperparameters such learning rate, dropout rate, and batch size.

This approach of improving a model has the advantage of being an organized procedure that combines knowledge from (1) the literature and (2) iterative experimentation. A second benefit is that the hyper-parameter optimisation procedure is designed to avoid an overabundance of hyper-values over a large number of models; individual hyper-parameters are only optimized when a promising upgraded model has been found using default parameters. Because the number of hyperparameters is related to the degree of overfitting, it aids in limiting overfitting [55].

4.2.10 Applying Steps 5 and 6 and results

The following is a step-by-step account of the implementation of the last two steps of the SeLFISA Framework, guided by the design principles outlined in Section 4.2.

Algorithm 4.4 Algorithm to arrive at an enhanced deep learning model

```
Inputs: Models (2,3,4,7) resulting from Step 3 of the SeLFISA framework;
        GBP/USD for training 80% and testing 20%;
        JPY/USD for validation
Part 1—Construct initial new models based on the baseline models
Identify combinations of models in Step 3 of the SeLFISA framework
        that enhance performance: $K = \{[2,3],[3,4],[3,4,7],[2,3,4,7]\}$
For each combination $M \in K$:
    Use the design of the models $M$ as a basis for a new design $EM$
        placed in set $E = \{EM : M \in K\}$ of enhanced models
    Train the model $EM$
        (epochs = 50; batch size = 50; validation split = 0.2; learning rate = 0.1)
    Use $EM$ to generate predictions on the testing set
    Record accuracy prediction metrics for $EM$ (MAE; MSE; RMSE, Adj. $R^2$)
    Visualise/Plot prediction along with ground truth

Part 2—Construct derivative models using baseline models
        and new models from Part 1 above
    While performance of any model in $E$ is not significantly better
        than baseline models and iterations $< N$:
        Execute Lines 8–14 of Part 1, for models $M \in [2,3,4,7]$ and models in $E$
            to create a new derivative model $EM$
        Append $EM$ to $E$

    Select the highest–performing $EM$ in $E$ as the final enhanced model
    Apply optimisation to $EM$ on additional hyperparameters
        (layers, activation functions, order)
    Apply grid search optimisation to $EM$ on additional hyperparameters
            (learn_rates = (0.001; 0.02; 0.2),
             dropout_rates = (0.0; 0.2; 0.4),
             batch_sizes = (10; 20; 30))
```

Algorithm 4.4, which is a more conceptual version of Algorithm 2, but provides detail, describes the process of proposing, designing and implementing an improved model utilized in this implementation[1]. In Part 1, we found critical combinations of the baseline models, such as $[2,3],[3,4],[3,4,7]$ and $[2,3,4,7]$. The architectures of the models in these combinations were used to create new models that corresponded to each combination. With a batch size of 50, a val-

[1]Accessible at: `https://bit.ly/3AQ2U1Y`

idation split of 0.2, and a learning rate of 0.1, each new model in this section was trained across 50 epochs. The testing set was then used to run each model to gather prediction values and record accuracy measures. Plotting the outputs against the ground truth allows them to be visualized.

As a result, four additional models were created. Because the new models created in Part 1 did not outperform the baseline models in Part 2, a number of iterations were conducted in which the baseline models and all new models created in each iteration were utilized to create a series of new models. This continued until the 11th iteration, when an improved model was discovered.

This improved model was then fine-tuned. On the new model, a range of layer configurations and activation functions were first investigated, and the best option was chosen. The learning rate, drop out rate, and batch size were then optimized using a grid search, with the values studied being $[0.001, 0.02, 0.2]$, $[0, 0.2, 0.4]$, and $[10, 20, 30]$, respectively. A total of 27 possible hyper-parameter combinations were investigated, with three alternatives for each hyper-parameter. The best combination was chosen and used to create the upgraded model in the end.

As a result of implementing the SeLFISA framework, an improved deep learning model architecture known as the *SeLFISA model*[1] and plotted in Figure 4.18 was created. The ideal model depicted in Figure 4.11 on Page 87 can be written as follows:

```
Bi(GRU(32)) + SeqSelfAtt(30) + Dropout(0.2) + Bi(LSTM(32)) + Bi(GRU(32)) +

Bi(LSTM(32)) + Bi(GRU(32)) + LSTM(32) + GRU(32) + Dense(1)
```

In contrast to ordinary GRUs or LSTMs, which have a unidirectional flow of input data or information flow, i.e. always backwards or always forwards, the

[1]Not to be confused with the SeLFISA *framework*, which is the overall process described in Section 4.2, pictured in Figure 4.13 and implemented in this section.

bidirectional mechanism within the GRU was positioned at the beginning to enhance performance of the overall neural network architecture by increasing concurrent multidimensional sequential flow of data in both directions, i.e. future to past and past to future, as shown in Figure 4.11. Because there is no previous information at this step of training, a GRU rather than an LSTM should be used as the first node. Following that, a self-attention technique was used as the first hidden layer to track and trace selected relevant discrete irregular sequential patterns within the dataset with three features, namely open price, highest price and lowest price in every time step in the input consisting of 100 steps to predict the price in the next time step [14].

By balancing the accuracy, efficiency and consistency of the models, alternating LSTMs and GRUs in other hidden layers improved overall performance. The learnt pattern was then supplied to a deeply linked dense neural network layer in the eighth hidden layer of an efficient plain GRU, which yielded the projected price.

The network was built by gating the model design with GRU as the head and tail of the model. It performed better than other combinations or baseline models. This is due to the fact that the financial dataset utilized, with 6135 data points, was a manageable size with a variable distribution of discrete irregular sequential points [29]. According to empirical's study by Jozefowicz et al. [68] on recurrent network architectures, any exponential increase in the size of a dataset would necessitate replacing GRU nodes in a model with an LSTM, which can store long sequences in larger datasets, but this could compromise model efficiency due to increased parametric size.

Self-attention, which is a component of transformer networks [68], proved to be a useful tool for uncovering hidden relationships among irregular patterns in sequential financial data items. This resulted in empirical exploration knowledge, which informed the decision to use an equal number of GRUs and LSTMs in the SeLFISA model's architectural design.

Table 4.1 shows a sampling of the sequential models generated by the SeLFISA framework. The SeLFISA framework's Step 2 found these prospective deep learning models. They combine GRU, LSTM unit, Conv1D, Conv1DTranspose, Dense layer, *Bi-directional mechanism* (Bi), *global attention mechanism* (Attention), *self-attention mechanisms* (SeqSelfAttention), *drop out layer* (Dropout), *Keras TimeDistributed* (TimeDist) and *RepeatVector layer*. The candidates are less successful than the Table 4.2 enhanced optimal model, whose findings are shown in Table 4.3.

4.2.11 Performance comparison of SeLFISA model

This chapter demonstrated how the framework can be used to develop better models. The unavailability of a systematic procedure for improving and upgrading the best deep learning models, with a focus on predicting irregular time-series sequences has been identified as a major challenge. The implementation process of the SeLFISA framework revealed that there are many underlying challenges associated with most modern deep-learning models for predicting sequential series. Complexity in modelling and capturing extremely long-term sequential patterns using traditional deep learning models such as RNN [59], lack of transparency and explainability in deep learning model implementation [23], and lack of a comprehensive comparison analysis of existing deep learning models for sequential learning are just a few of the challenges [29]

Table 4.3 on Page 102 shows the findings of the baseline models as well as the findings of the *SeLFISA model*, with the metric values highlighted to make

Table 4.1: Candidate deep learning models identified in Step 2 of the SeLFISA framework

Model	Architecture	Remarks
1	LSTM(32) + Dropout (0.2) + Dense (1)	Derived from Azlan et al. (2019) [8] and Mihaita et al. (2019) [90]
2	LSTM(32) + LSTM(64) + Dropout(0.2) + LSTM(128) + Dropout(0.5) + Dense(1)	Influenced by Glenski et al. (2019) [41] and Chalvatzisa et al. (2019) [22]
3	Bi(LSTM(50)) + Dense(10) + Dense(10) + Dense(1)	A gated LSTM suggested by Sardelicha and Manandhara (2018) [112]
4	Bi(GRU(50)) + Dense(10) + Dense(10) + Dense(1)	A gated GRU mentioned by Sardelicha and Manandhara (2018) [112]
5	LSTM(100) + Dropout(100) + Attention(SeqSelfAttention(32)) + LSTM(16) + Dense(10) + Dense(10) + Dense(1)	Derived from experiments by Huang (2019) [59]
6	LSTM(32) + Conv1D(32) + Dropout(0.2) + Conv1D(16) + Conv1DTr(16) + Dropout(16) + Conv1D(32) + Conv1D (16) + Attention(SeqSelfAttention(1)) + LSTM(16) + Dropout(0.2) + Dense (1)	As indicated by Makinen et al. (2018) [85] and Huang (2019) [59]
7	LSTM(32) + Dropout(100) + Attention(SeqSelfAttention(32)) + LSTM(16) + Dense(10) + Dense(10) + Dense(1)	As implemented by Liu (2018) [81]
8	LSTM(32) + Dropout(0.2) + Attention(SeqSelfAttention(32)) + Bi(LSTM(32)) + Bi(LSTM(32)) + Dense(10) + Dense(1)	Demonstrated by Sardelicha and Manandhara (2018) [112]
9	LSTM(32) + Conv1D(32) + Dropout(0.2) + Conv1D(16) + Conv1DTranspose(16) + Dropout(0.2) + Conv1DTranspose(32) + Conv1DTranspose(1) + GRU(32) + Dropout(0.5) + Dense(1)	Suggested by Maggiolo and Spanakis (2019) [84]
10	GRU(32) + GRU(64) + Dropout(0.2) + GRU(128) + Dense(1)	Designed by GRU by Qin (2019) [106]
11	LSTM(32) + LSTM(64) + RepeatVector(64) + LSTM(64) + TimeDist(1) + LSTM(128) + Dropout(128) + Dense(1)	Suggested by Qin (2019) [106]
12	LSTM(50) + Dropout + LSTM(100) + Dropout(0.5) + GRU(100) + LSTM(100) + Dropout(0.5) + LSTM(100) + Dropout (0.5) + Dense(100) + Dense(10) + Dense(10) + Dense(1)	Implemented by Bai (2019) [10]

Table 4.2: The optimal model produced by SeLFISA

Model	Architecture	Remarks
SeLFISA	BiD(GRU(32)) + SeqSelfAtt(att_width=30) + Dropout(0.2) + BiD(LSTM(32)) + BiD(GRU(32)) + BiD(LSTM(32)) + BiD (GRU(32)) + LSTM (32) + GRU(32) + Dense(1)	Proposed optimized model

comparisons with the best baseline models easier. When compared to the top performing baseline models, the total improvement P_i of the SeLFISA model, determined using Equation 4.6, is presented. P_i is the percentage difference between two metric values based on these data, adjusted by dividing the difference by the mean of the two values. In each metric category, using P_i to measure the improved performance of the SeLFISA model over the best baseline model for purposes of comparison, where a positive value of P_i signifies an improvement of the SeLFISA model over the baseline and a negative value of P_i indicates a drop in performance of the SeLFISA model over the baseline model.

$$P_i = \frac{m_S - m_b}{(m_S + m_b)/\lambda} \times 100, \tag{4.6}$$

where m_b is a given metric value of the best-performing baseline model, m_S is the corresponding metric value of the SeLFISA model and λ is an averaging constant which is usually set to 2.

When looking at the data in Table 4.3 on Page 102, the fact that the SeLFISA model outperforms the highest performing baseline models in every metric category is quite encouraging. This is quantified and proved by positive P_i values in every case. On the testing set, i.e. the GBP/USD testing set, the SeLFISA model outperforms the best baseline models by 47% in MAE and 156% in MSE, while providing a small increase in adjusted R^2. The gain in MSE is noteworthy; as previously stated, the MSE is sensitive to outliers and an increase in MSE shows that the SeLFISA model is significantly more robust to outliers, i.e. irregular patterns. As a result, it is quite successful for what it was designed.

The SeLFISA model outperforms the baseline models on the validation set which is different from the one used for training: 115% improvement in MAE, 51% improvement in MSE, and 15% improvement in R^2. This is supported by a consistency value of 2.74, which is about 1.5 times greater than the most consistent baseline model, Model 4, with a consistency of 1.88. This is a promising outcome, indicating that the SeLFISA framework has created a model that can be adaptable to an unknown dataset in the same format as the training set

.

Given the foregoing, it was critical to assess how the upgraded model's significant gains in accuracy and consistency would affect its efficiency. However, it is seen that the enhanced model delivers a significant improvement in this aspect as well, with an efficiency value of nearly 29 compared to approximately 20 for Model 2, the most efficient baseline model. The enhanced model is not only the most accurate and consistent of the models evaluated, but it is also the most efficient.

Visual findings have been provided in Figures 4.14–4.18, which are plots of the predictions of each baseline model or the SeLFISA model vs. the ground truth, as a confirmation of the objective results and analysis above. Each plot shows the actual vs. expected price over time. When the baseline models' plots are compared to the SeLFISA model's, it can be seen that the SeLFISA model's forecasts are closer to the ground reality than any baseline model's. This visually verifies the numerical results.

SeLFISA's robust performance is further validated in the graphs in Figure 4.14-18 that compares it to the top 4 existing state-of-the-art baseline models. This is after being trained on a separate portion of the GBP/USD training set. When their performance were evaluated on a separate GBP/USD testing set, and a completely unseen JPY/USD validation dataset. Comparing Model 2 in Figure 4.14, Model 3 in Figure 4.15, Model 4 in Figure 4.16 and Model 7 in Figure 4.17 to the SeLFISA model in Figure 4.18, the SeLFISA model produced best results in every performance category. The five graphs shows that SeLFISA model's prediction significantly decreased both the graphical horizontal and vertical displacement between the existing financial currency price and the predicted price. Hence, the SeLFISA model generated the best results in every category since it significantly decreased the graphical horizontal and vertical displacement.

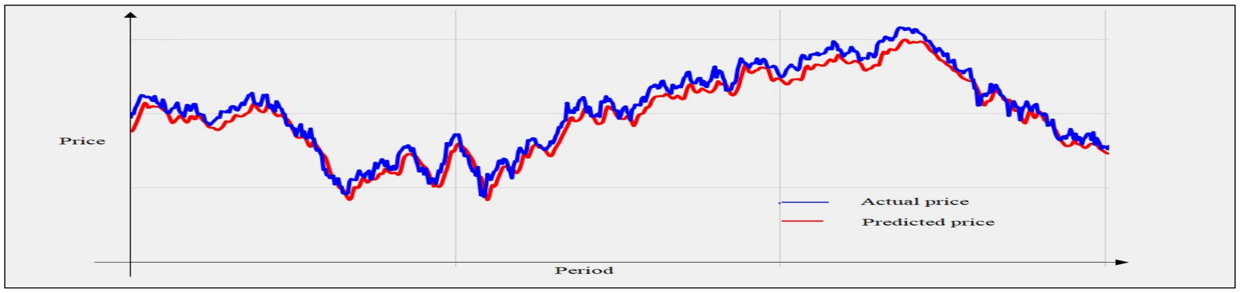


Figure 4.14: Model 2
LSTM(32) + LSTM(64) + Dropout(0.2) + LSTM(128) + Dropout(0.5) + Dense(1) implemented by Glenski and Hristu-Varsakelis (2019) [41]

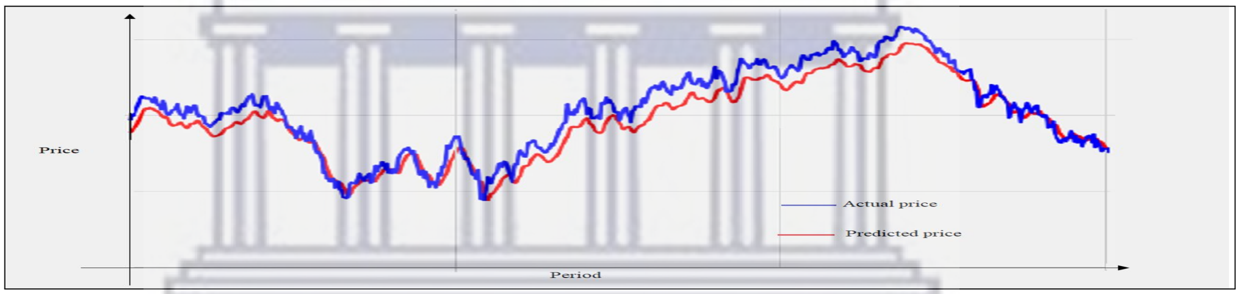


Figure 4.15: Model 3
Bi(LSTM(50)) + Dense(10) + Dense(10) + Dense(1) influenced by Sardelicha and Manandhara (2018) [112]

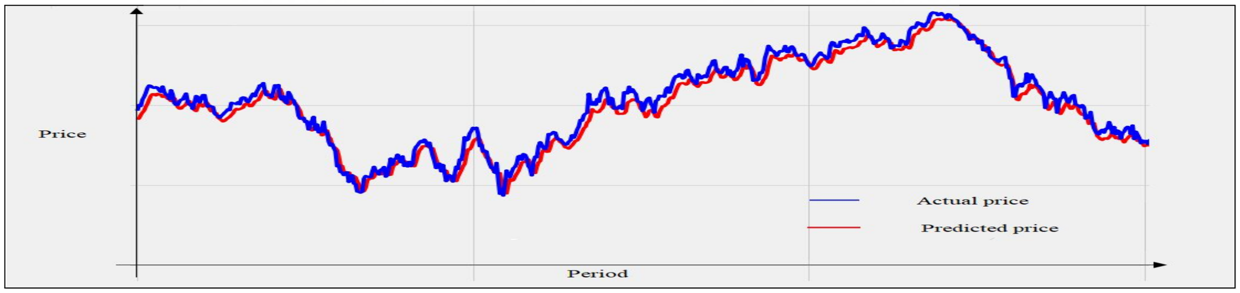


Figure 4.16: Model 4
Bi(GRU(50)) + Dense(10) + Dense(10) + Dense(1) by Sardelicha and Manandhara (2018) [112]

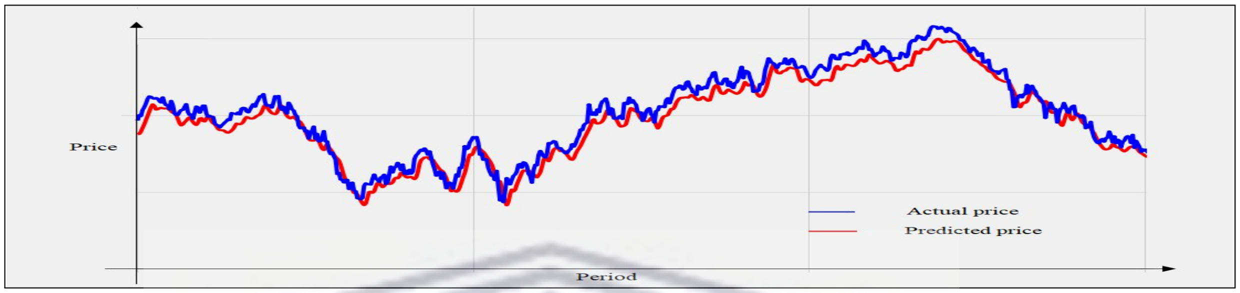


Figure 4.17: Model 7

LSTM(32) + Dropout(100) + Attention(SeqSelfAtt(32)) + LSTM (16) + Dense(10) + Dense(10) + Dense(1) by Liu (2018) [81]

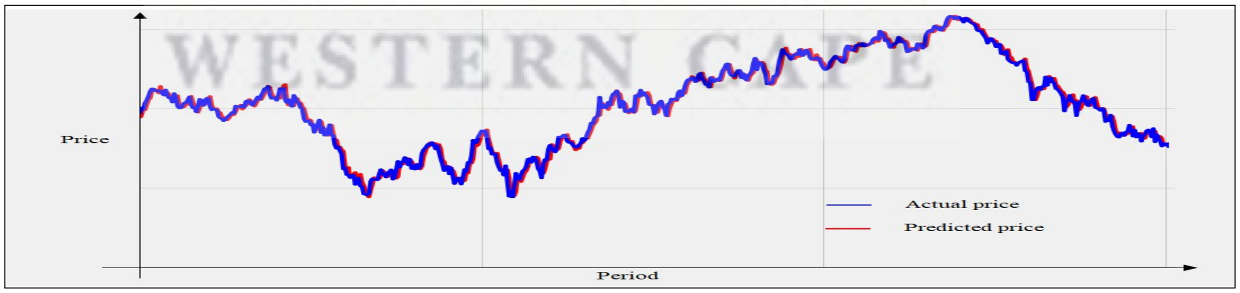


Figure 4.18: SeLFISA Model
Bi(GRU(32)) + SeqSelfAtt(30) + Dropout(0.2) + Bi(LSTM(32)) + Bi(GRU(32)) + Bi(LSTM(32)) + Bi(GRU(32)) + LSTM(32) + GRU(32) + Dense(1)

Table 4.3: Results of top-performing baseline models compared with SeLFISA model

Model	GBP/USD dataset			JPY/USD dataset			Training Efficiency			Consistency
	MAE	MSE	Adj. R^2	MAE	MSE	Adj. R^2	Number of Parameters	Time Seconds	Efficiency	
2	0.0487	0.00349	0.865	0.502	0.263	-5.15	128513	6430	**19.99**	0.61
3	**0.0167**	0.00311	**0.976**	0.172	0.0331	0.226	23131	1210	19.12	0.61
4	0.0321	0.00236	0.828	**0.0554**	**0.00561**	**0.362**	17931	963	18.62	**1.88**
7	0.0197	**0.00208**	0.885	0.345	0.127	-1.96	10276	3150	3.26	0.56
SeLFISA	**0.0103**	**0.000255**	**0.981**	**0.0149**	**0.00333**	**0.421**	117538	4080	**28.81**	**2.74**
P_i	47.41%	156.32%	0.51%	115.22%	51.01%	15.07%			36.15%	37.42%

4.3 Summary

Chapter 4 developed and implemented a novel SeLFISA framework to improve deep learning models for time-series prediction from existing state-of-the-art baseline models, based on DSR and specific methods identified in Chapter 3. Using a financial currency exchange dataset as an example, this Chapter demonstrated how the framework works. In Chapter 5, the outcomes of the implementation process are discussed.

Chapter 5

Discussion, Limitations, Future Recommendations and Conclusion

In this chapter the findings of the research are discussed in terms of the research questions posed. The contribution as well as limitations and future recommendations are highlighted.

5.1 Reflecting on the study

This research considered how deep learning models should be developed in order to improve on the existing state-of-the-art models when predicting discrete irregular sequential patterns. Deep learning models for discrete irregular sequential prediction encounter a variety of challenges as was found after doing a systematic literature review of recently published scientific articles. A six-step SeLFISA framework[1] was created in a high-performance computing

[1]See Figure 4.1 on Page 52.

experimental setting using a design science research methodology.

1. The **first sub-question** of the study was, *"How should existing state-of-the-art deep learning approaches and relevant existing datasets in a given domain be effectively identified and selected?"* To answer this question, a customized SLR research method using the *preferred reporting items for systematic reviews and meta-analyses*(PRISMA) methodology was blended with grounded theory to facilitate a detailed literature review [1]. This identified well-known deep learning models and datasets in a given domain and their respective performance and methodological challenges.

   Sequential time-series datasets from different domains were identified, captured and recorded in a sharable knowledge bank that can be utilised by researchers. This process systematically demonstrated that existing literature is inadequate, contradictory and inconsistent and it does not provide a comprehensive comparison. The systematic literature review resulted in a clear and thorough summary of research gaps, inconsistencies and conflicts, which served as a foundation for improving deep learning models for predicting such complex datasets [30].

2. The **second sub-question** of the research was, *"How should the state-of-the-art deep learning approaches identified, be systematically combined and improved to arrive at a deep learning model with enhanced performance?"* The sequence and design of layers in a deep learning model have a direct impact on the model's prediction efficacy. A further detailed review of current approaches indicated that there is a wide range of deep learning designs and arrangements, most of these used a trial-and-error strategy to construct more effective models. There is currently no established procedure for the systematic improvement of such models. Using

the SeLFISA framework, datasets characterised with the most discrete irregular sequential patterns were identified by applying Billauer's algorithm and interquartile range outlier calculations to select, explore and evaluate them [149]. The daily exchange rate data between the GB Pound and the US Dollar was identified as a dataset suitable for learning because of its high irregularity [23]. A daily exchange rate dataset between the Japanese Yen and the US Dollar was then applied as a validation dataset.

In addition, the systematic literature review identified 12 implementable or executable sequential forecasting models from various authors, which constituted candidate artefacts. A combination of architecture distinctiveness and referenced performance qualities were considered during the selection process. Gated RNNs, autoencoders, convolutional neural networks, bidirectional mechanisms, attention mechanisms, ensemble techniques, deep and vanilla architectures were used in some of these models. These 12 models were chosen for their architectural design qualities, which included the gated LSTM architecture [8, 41, 22]; the bidirectional mechanism combined with both LSTMs and GRUs [112]; the attention mechanism combined with gated neural networks [59]; deep CNN ensemble with LSTM and an attention mechanism [85]; a GRU [106] autoencoders combined with LSTM [144] and finally a deep gated recurrent neural network architecture made up of both GRU and LSTM [10].

A framework that includes the SLR process was developed in Section 4.2, which can systematically improve on existing models to arrive at an enhanced deep learning model. The SeLFISA framework produced an enhanced deep learning model with higher efficacy, explainability, performance consistency, and straightforwardness when compared with baseline models. To achieve this, Step 3—Implementation, Step 4—Proposing and designing a new artefact, Step 5—Mapping and Step 6—Implementation of the SeLFISA framework were used to demonstrate a way to enhance the model for predicting such datasets. This process took advantage of current deep learning artefact variants and the identified datasets with the most discrete irregular sequential patterns.

Based on the results and insights gained from the preceding steps, the SeLFISA framework iteratively arrived at a new neural network architecture design with improved performance. It used a two-part procedure[1] to harness the best attributes of the baseline models. The first part identified important best-performing combinations from the 12 baseline models, resulting in four new models, i.e $[2,3], [3,4], [3,4,7], [2,3,4,7]$ to drive the development of new models matching each combination. The new model was trained and then validated on the testing set to get prediction values that were displayed against the ground truth, as well as accuracy measures. With a batch size of 50, a validation split of 0.2, and a learning rate of 0.1, each new model in this section was trained across 50 epochs. The testing set was then used to run each model to gather prediction values and record accuracy measures. Plotting the outputs against the ground truth allowed them to be fully visualized.

Procedures of the SeLFISA framework provided a way to determine whether any of the new models created outperformed the baseline models [2]; if so, the best new model was chosen as the final enhanced model. If not, the operation was repeated, this time using the newly learned insights and knowledge about the architectures of all baseline and new models. This

[1]See Algorithms 4.1–4.4 on Pages 81–93.
[2]See Algorithm 4.4 on Page 93.

was repeated until a better model was obtained. Because the new models initially created by the SeLFISA framework did not outperform the baseline models in the case of the daily financial currency exchange rate dataset, a series of iterations were carried out in which the baseline models and all new models produced in each iteration were utilized to develop a series of new models. This continued until the 11th iteration when a better model was discovered. The improved model was then guided by the algorithm. A variety of layer configurations and activation functions were evaluated first on the new model, and the best option was chosen. The learning rate, dropout rate, and batch size were then optimized using grid search, with the values studied being $[0.001, 0.02, 0.2]$, $[0, 0.2, 0.4]$, and $[10, 20, 30]$, respectively. A total of 27 distinct hyper-parameter combinations were investigated, with three alternatives for each hyper-parameter. The best combination made up of:

```
Bi(GRU(32)) + SeqSelfAtt(30) + Dropout(0.2) + Bi(LSTM(32)) + Bi(GRU(32)) +

Bi(LSTM(32)) + Bi(GRU(32)) + LSTM(32) + GRU(32) + Dense(1)
```

was chosen as the final upgraded SeLFISA model.

The entire process of creating a new deep learning model with improved efficacy was done with high degrees of explainability, performance consistency, and straightforwardness characteristics of the SeLFISA framework.

3. The **third sub-question** asked, "*How should the performance of deep learning models be evaluated when applied to such datasets?*" The developed framework was applied to the field of forex price prediction in order to demonstrate the framework's potential to arrive at an enhanced deep learning model. Experiments were carried out to compare the enhanced

model with the identified state-of-the-art models. This inquiry was answered by evaluating the model's robustness based on multidimensional performance evaluation criteria. To evaluate the performance resilience of the specified deep learning model, new metrics such as SeLFISA efficiency and performance improvement were developed.

The performance of the SeLFISA model, measured qualitatively and quantitatively, exceeded baseline state-of-the-art models. This took into account the following performance evaluation criteria: (1) quantitative accuracy measured using MAE, MSE, and Adj. R^2, (2) quantitative efficiency measured using a derived SeLFISA efficiency metric in E_s Equation 4.5 on Page 62, (3) quantitative performance consistency measured using a derived SeLFISA consistency metric in C_s Equation 4.4 on Page 61—demonstrated by the process flow of the SeLFISA framework, (4) qualitative straightforwardness—demonstrated by the six distinct iterative steps of the SeLFISA framework which led to an enhanced output and finally, and (5) qualitative design consistencies—which is illustrated by the aggregation robustness performance characteristic of the framework over existing trial and error approaches.

The model produced by the SeLFISA framework of Section 4.2 was applied to predict forex price to demonstrate its potential. This work demonstrated SeLFISA framework provides a useful platform for deriving an enhanced deep learning model that outperformed the baseline deep learning models in terms of accuracy, efficiency and consistency performance.

5.2 Evaluation of the findings

The findings of this study revealed that most models with higher state-of-the-art performance were typically developed by trial-and-error [30]. The SeLF-ISA framework addresses model improvement in a systematic way. It has the advantage of being a structured process that pools the knowledge from the literature with iterative experimentation. A second advantage is that the hyper-parameter optimisation process is constructed in such a way that it avoids an explosion of hyper-parameters over a large number of models—specific hyper-parameters are only optimised once a promising enhanced model is arrived at. Models derived using this framework are consistent, efficient, accurate, explainable and straightforward [29, 115].

5.3 Contribution

Three important contributions to the field of enhancing deep learning models for the prediction of discrete irregular sequential datasets were achieved in this thesis. These unique contributions to knowledge came from various stages of the framework.

1. **Development of a novel framework**: The first and the main contribution of this work is a novel framework which provides a systematic procedure which researchers can apply in order to systematically improve on the best deep learning models found in the literature in a given domain. As a concrete example, the author demonstrates the proposed framework in action in the field of forex price prediction in Section 4.2.5.

2. **SLR**: The second contribution of this work is a *SLR* that can be used to systematically uncover, rank and select research studies of significance in a given domain. To demonstrate the proposed SLR process in action,

and as part of the proposed framework, this work practically applies the proposed SLR process to the domain of forex price prediction. The *SLR* created a knowledge bank of articles, variables and metadata to provide a comprehensive, targeted, dependable, reproducible and extensive literature overview [101, 6, 137, 104]. It answered the following questions:

(a) *Which datasets are sequential in nature with irregular characteristics?* Using Box and Whisker plot and Billauer's techniques, two datasets with most discrete irregular sequences were selected from a nucleus of 32 articles

(b) *Which artefacts in the form of models have been applied elsewhere to analyse such datasets?* From the 32 publications reviewed, 12 deep learning models based on specific algorithms and architectures were selected. These were recorded in a sharable open access knowledge bank for future use.

(c) *How were those models evaluated in terms of determining their performance*? Specific performance evaluation criteria and their respective metrics were identified. Most approaches for sequential time series prediction relied on limited performance evaluation criteria with accuracy as a widely used criterion calculated using metrics such as MAE, MSE, and adjusted R^2. This revealed a research gap and the need for the development of multidimensional evaluation procedures.

3. **A model for currency exchange rate prediction**: The third contribution is an enhanced deep learning model arrived at by applying the proposed framework to the domain of forex price prediction, where the enhanced model was shown to out-perform the state-of-the-art identified within the SLR in the framework in terms of performance, consistency and efficiency.

5.4 Limitations and future recommendations

Deep learning is a rapidly evolving domain. The scope of this study was restricted to the forecasting of financial time series data. The SeLFISA framework has proven to be reliable in financial currency exchange datasets, but it has to be tested on other datasets such as health, weather, and transportation to confirm its usefulness in different fields. Further research could look into other procedure for addressing the variable explosion issue for sequential time-series prediction leveraging SeLFISA framework's capabilities.

5.5 Conclusion

This study investigated the current problem associated with the design of deep learning models for analysis of discrete irregular patterned sequential datasets. The findings of this study indicated a lack of a systematic approach to guide the development of a deep learning model that outperforms existing state-of-the-art models, especially for discrete irregular sequential patterns such as currency exchange. The study presented a new framework with a six-step mechanism to design a better model for predicting discrete irregular sequential patterns.

An enhanced model was created, demonstrating competitive outcomes in relation to prior research in terms of enhanced performance efficiency and consistency. Specifically, it exhibited a 36.15% improvement in performance efficiency and a 37.42% improvement in consistency when compared to the most advanced models currently available for predicting financial currency exchange rates derived from 412 latest articles. The framework is a contribution to the field of artificial intelligence and machine learning since it is demonstrated how modern deep learning models can be systematically refined, improved and evaluated. This thesis's publications add to the body of scientific knowledge in a number of ways including model design and improvement, algorithmic development and enhancement, empirical multi-dimensional evaluations of models, interpretability and explainability considerations, irregular sequential datasets generation and curation and domain-specific applications.

Bibliography

[1] M. Abelha, S. Fernandes, D. Mesquita, F. Seabra, and A. T. Ferreira-Oliveira, "Graduate employability and competence development in higher education—A systematic literature review using PRISMA," *Sustainability*, vol. 12, pp. 33–46, 2020.

[2] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," *arXiv*, vol. [cs.NE], no. 1803.08375, pp. 1–7, 2018.

[3] U. Aivodji, H. Arai, O. Fortineau, S. Gambs, S. Hara, and A. Tapp, "Fairwashing: the risk of rationalization," in *Proceedings of the 36th International Conference Machine Learning Research*, vol. 97, pp. 161–170, 2019.

[4] M. Z. Alom, T. M. Taha, C. Yakopcic, and S. Westberg, "The history began from AlexNet: A comprehensive survey on deep learning approaches," *arXiv*, vol. [cs.CV], no. 1803.01164, pp. 1–39, 2018.

[5] L. Alzubaidi, J. Zhang, and A. J. Humaidi, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *Journal of Big Data*, vol. 8, no. 53, pp. 497–505, 2021.

[6] D. Andreini and C. Bettinelli, *Business Model Innovation: From Systematic Literature Review to Future Research Directions*. Cham, Switzerland: Springer, 2017.

[7] A. B. Arrieta, N. D. Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera, "Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI," *Information Fusion*, vol. 58, pp. 82–115, 2020.

[8] A. Azlan, Y. Yusof, and M. F. M. Mohsin, "Determining the impact of window length on time series forecasting using deep learning," *International Journal of Advanced Computer Research*, vol. 9, no. 44, pp. 260–267, 2019.

[9] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv:1803.01271*, vol. [cs.LG], pp. 1–14, 2018.

[10] S. Bai and J. Z. K. V. Koltun, "Deep equilibrium models," in *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, (Vancouver), pp. 1–16, 2019.

[11] T. M. Banday and I. Sultan, "Opportunities to address security challenges on the IoT node side," in *13th Jammu and Kashmir Science Congress*, (Srinagar, India), p. 707, 2019.

[12] W. Bao, J. Yue, and Y. Rao, "A deep learning framework for financial time series using stacked autoencoders and long-short term memory," *PLoS ONE*, vol. 12, no. 7, pp. 1–24, 2017.

[13] J. Batani, E. Mbunge, B. Muchemwa, G. Gaobotse, C. Gurajena, S. Fashoto, T. Kavu, and K. Dandajena, "A review of deep learning models for detecting cyberbullying on social media networks," in *11th Computer Science On-line Conference*, pp. 1–27, 2022.

[14] Y. Bathaee, “The artificial intelligence black box and the failure of intent and causation,” *Harvard Journal of Law & Technology*, vol. 31, no. 2, pp. 890–938, 2018.

[15] Y. Bengio and Y. LeCun, “Convolutional networks for images, speech, and time-series,” *The handbook of brain theory and neural networks*, pp. 255–258, 1997.

[16] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, *Feature Selection for High-Dimensional Data*. Switzerland: Springer, 2015.

[17] A. Borovykh, S. Bohte, and C. Oosterlee, “Dilated convolutional neural networks for time series forecasting,” *Journal of Computational Finance*, pp. 1–23, 2017.

[18] G. E. P. Box and G. M. Jenkins, “Time series analysis: Forecasting and control,” *Feedback Control Systems*, vol. 865, pp. 1–575, 1976.

[19] J. Bradbury, S. Merity, C. Xiong, and R. Socher, “The history began from AlexNet: A comprehensive survey on deep learning approaches,” *arXiv*, vol. [cs.NE], no. 1611.01576, pp. 1–11, 2016.

[20] V. Campos, B. Jou, X. Giró-i-Nieto, J. Torres, and S. Chang, “Skip RNN: learning to skip state updates in recurrent neural networks,” *arXiv:1708.06834*, vol. [cs.AI], pp. 1–12, 2018.

[21] V. Cerqueira, L. Torgo, and C. Soares, “Machine learning vs statistical methods for time series forecasting: Size matters,” *arXiv:1909.13316*, vol. [stat.ML], pp. 1–9, 2019.

[22] C. Chalvatzisa and D. Hristu-Varsakelis, “High-performance stock index trading: making effective use of a deep long short-term memory network,” *arXiv:1902.03125*, vol. [q-fin.ST], pp. 1–30, 2019.

[23] Y.-Y. Chang, F.-Y. Sun, Y.-H. Wu, and S.-D. Lin, “A memory-network based solution for multivariate time-series forecasting,” *arXiv:1809.02105*, vol. [cs.LG], pp. 1–8, 2018.

[24] A. Chattopadhyay, E. Nabizadeh, and P. Hassanzadeh, “Analog forecasting of extreme-causing weather patterns using deep learning,” *Journal of Advances in Modeling Earth Systems*, vol. 12, no. 2, pp. 1–14, 2020.

[25] T. Chen, J. Liu, Y. Xiang, W. Niu, E. Tong, and Z. Han, “Adversarial attack and defense in reinforcement learning-from AI security view,” *Cybersecurity*, vol. 2, 2019.

[26] K. Cho, B. van Merriërnboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” *arXiv:1409.1259*, vol. [cs.CL], pp. 1–9, 2014.

[27] M. Crotty, *The Foundations of Social Research: Meaning and Perspective in the Research Process*. Thousand Oaks, Ca: SAGE Publications, 1998.

[28] Z. Cui, R. Ke, Z. Pu, and Y. Wang, “Stacked bidirectional and unidirectional LSTM recurrent neural network for forecasting network-wide traffic state with missing values,” *Transportation Research Part C: Emerging Technologies*, vol. 118, pp. 102–674, 2020.

[29] K. Dandajena, I. M. Venter, M. Ghaziasgar, and R. Dodds, “Complex sequential data analysis: A systematic literature review of existing algorithms,” in *Conference of the South African Institute of Computer Scientists and Information Technologists 2020*, (Cape Town), pp. 44–50, 2020.

[30] K. Dandajena, I. M. Venter, M. Ghaziasgar, and R. Dodds, “Selecting datasets for evaluating an enhanced deep learning framework,” in *SATNAC 2021*, (Drakensberg, Natal), pp. 1–7, 2021.

[31] K. Das and R. N. Behera, "A survey on machine learning: Concept, algorithms and applications," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 5, 2017.

[32] S. Das and A. Endert, "Cactus: Detecting and resolving conflicts in objective functions," *arXiv:2103.07805 [cs.LG]*, vol. 1, pp. 1–13, 2021.

[33] K. Dashtipour, M. Gogate, A. Adeel, C. Leracitano, H. Larijani, and A. Hussain, "Exploiting deep learning for Persian sentiment analysis," *arXiv:1808.05077*, vol. [cs.CL], pp. 1–9, 2018.

[34] R. Debnath, S. Darby, R. Bardhan, K. Mohaddes, and M. Sunikka-Blank, "Grounded reality meets machine learning: A deep-narrative analysis framework for energy policy research," *Energy Research and Social Science*, vol. 69, 2020.

[35] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *arXiv:1810.04805*, vol. 2, pp. 1–16, 2018.

[36] V. Dignum, "Ethics in artificial intelligence: introduction to the special issue," *Ethics and Information Technology*, vol. 20, pp. 1–3, 2018.

[37] F. Feng, H. Chen, X. He, J. Ding, M. Sun, and T. S. Chua, "Enhancing stock movement prediction with adversarial training," *arXiv:1810.09936*, vol. [q-fin.TR], pp. 1–7, 2019.

[38] B. Fitzgerald and D. Howcroft, "Towards dissolution of the is research debate:from polarization to polarity," *Journal of Information Technology*, vol. 13, no. 4, pp. 313–326, 1988.

[39] M. D. Gall, W. R. Borg, and J. P. Gall, *Educational Research: An Introduction*. Longman Publishing, 1996.

[40] A. García-Durán, S. Dumančić, and M. Niepert, “Learning sequence encoders for temporal knowledge graph completion,” *Association for Computational Linguistics*, pp. 4816–4821, 2018.

[41] M. Glenski, T. Weninger, and V. Volkova, “Improved forecasting of cryptocurrency price using social signals,” *arXiv:1907.00558*, vol. [q-fin.ST], pp. 1–11, 2019.

[42] H. Gonen and Y. Goldberg, “Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them,” in *Conference of the North American Chapter of the Association for Computer Linguistics*, vol. 1, (Minneapolis, MN), pp. 609–614, 2019.

[43] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2016.

[44] I. J. Goodfellow, J. P. Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *arXiv*, vol. [stat.ML], no. 1406.2661, pp. 1–9, 2014.

[45] M. Goyal, R. Goyal, R. P. Venkatappa, and B. Lall, “Activation functions. in: Deep learning: Algorithms and applications,” *Studies in Computational Intelligence*, vol. 865, pp. 1–30, 2020.

[46] D. Graupe, *Deep Learning Neural Networks*. Singapore: World Scientific Publishing Company, 2016.

[47] A. Graves, S. Fernandez, and J. Schmidhuber, “Multi-dimensional recurrent neural networks,” *arXiv:0705.2011*, vol. [cs.AI], pp. 1–10, 2007.

[48] K. Greff, R. Srivastava, J. Koutnik, B. Steunebrink, and J. Schmidhuber, “LSTM: A search space odyssey,” *Transactions on Neural Networks and Learning Systems*, vol. 2, pp. 1–12, 2017.

[49] P. Gábor, “Recurrent neural networks for time series forecasting,” *arXiv:1901.00069*, vol. 1, pp. 1–23, 2019.

[50] U. Güclü and M. A. J. van Gerven, “Modeling the dynamics of human brain activity with recurrent neural networks,” *Frontiers in Computational Neuroscience*, vol. 11, no. 1551, 2017.

[51] Z. Han, J. Zhao, H. Leung, K. Ma, and W. Wang, “A review of deep learning models for time series prediction,” *(IEEE) Sensors Journal*, vol. 21, no. 1803.08375, pp. 7833–7848, 2021.

[52] B. V. Hemalatha and R. Vijayalatha, “Big data—Technologies, challenges and future,” *International Journal of Engineering Research in Computer Science and Engineering*, vol. 5, pp. 235–239, 2018.

[53] J. Hernandez-Orallo, F. Martınez-Plumed, S. Avin, and J. Whittlestone, “AI paradigms and AI safety: Mapping artefacts and techniques to safety issues,” *24th European Conference on Artificial Intelligence - ECAI 2020*, pp. 1–8, 2020.

[54] G. Hinton, S. Sabour, and N. Frosst, “Matrix capsules with EM routing,” *The Sixth International Conference on Learning Representations*, 2018.

[55] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *arXiv:1207.0580*, vol. [cs.NE], pp. 1–18, 2012.

[56] M. K. Ho, H. Darman, , and S. Musa, “Stock price prediction using ARIMA, neural network and LSTM models,” *Journal of Physics: Conference Series*, pp. 1–12, 2021.

[57] H. Hochheiser and B. Shneiderman, “Visual queries for finding patterns in time series data,” *DMSA*, pp. 1–8, 2002.

[58] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 10, pp. 1735–1780, 1997.

[59] S. Huang, D. Wang, X. Wu, and A. Tang, "DSANet: Dual self-attention network for multivariate time series forecasting," in *CIKM '19: Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, (Beijing), pp. 2129–2132, ACM, 2019.

[60] R. Huffaker, E. Berg, and M. Canavari, "Reconstructing deterministic economic dynamics from volatile time series data," *The Routledge Handbook of Agricultural Economics*, vol. 1, pp. 1–15, 2018.

[61] J. A. Hughes and W. W. Sharrock, *The Philosophy of Social Research*. Longman London and New York: Longman, 1997.

[62] L. M. Huglin, *The Relationship Between Personal Epistemology and Learning Style in Adult Learners*. University of Idaho, 2003.

[63] A. Husein, M. Arsyal, S. Sinaga, and H. Syahputa, "Generative adversarial networks time series models to forecast medicine daily sales in hospital," *SinkrOn*, vol. 3, pp. 1–112, 2019.

[64] M. E. Hussein, S. Hirst, V. Salyers, and J. Osuji, "Using grounded theory as a method of inquiry: Advantages and disadvantages. the qualitative report," *HCAS Journals*, vol. 19, no. 44, pp. 1–15, 2014.

[65] K. N. Igwe and I. Ahiaoma, "Imperative of cyber ethics education to cyber crimes prevention and cyber security in nigeria," *International Journal of ICT and Management*, vol. 10, pp. 102–115, 2014.

[66] S. N. Imenda, "Is there a conceptual difference between theoretical and conceptual frameworks?," *Journal of Social Sciences*, vol. 38, pp. 185–195, 2014.

[67] S. Jeble, S. Kumari, and Y. Patil, "Role of big data in decision making," *Operations and Supply Chain Management*, vol. 11, no. 1, pp. 36–44, 2018.

[68] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proceedings of the 32nd International Conference on Machine Learning*, (Lille), pp. 1–9, 2015.

[69] M. J. Kearns, A. Roth, and S. Sharifi-Malvajerdi, "Average individual fairness: Algorithms, generalization and experiments," *arXiv:1905.10607*, vol. [CoRR], pp. 1–46, 2019.

[70] J. D. Kelleher, *Deep Learning*. Cambridge, MA: The MIT Press, 2019.

[71] B.-T. Kim, Y. Kim, and S. Lee, "Prediction of lateral behavior of single and group piles using artificial neural networks," *KSCE Journal of Civil Engineering*, vol. 5, pp. 185–198, 2001.

[72] A. Koenecke, "Applying deep neural networks to financial time series forecasting," 2020.

[73] J. Koutnik, K. Greff, F. Gomez, and J. Schmidhuber, "A clockwork RNN," *arXiv*, vol. [cs.NE], no. 1402.3511, pp. 1–9, 2014.

[74] V. Kuleshov, N. Fenner, and S. Ermon, "Accurate uncertainties for deep learning using calibrated regression," *arXiv:1807.00263*, vol. [cs.LG], pp. 1–9, 2018.

[75] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[76] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," *2010 IEEE International Symposium on Circuits and Systems: Nano-Bio Circuit Fabrics and Systems*, pp. 253–256, 2010.

[77] J. Lee, H. Davari, J. Singh, and V. Pandhare, "Industrial artificial intelligence for industry 4.0-based manufacturing systems," *Manufacturing Letters*, vol. 18, pp. 20–23, 2018.

[78] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently recurrent neural network (IndRNN): Building a longer and deeper RNN learning," *arXiv*, vol. 1803.04831 [cs.CV], pp. 1–11, 2018.

[79] B. Lim and S. Zohren, "Time series forecasting with deep learning: A survey," *Philosophical Transactions of the Royal Society A*, pp. 1–13, 2020.

[80] J. Lin, E. Keogh, S. Lonardi, J. P. Lankford, and D. M. Nystrom, "VizTree: a tool for visually mining and monitoring massive time series databases," *KDD '04: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, no. 4, pp. 460–469, 2004.

[81] J. Liu, T. Zhang, G. Han, and Y. Gou, "TD-LSTM: Temporal dependence-based LSTM networks for marine temperature prediction," *Sensors*, vol. 18, no. 3697, pp. 1–13, 2018.

[82] I. E. Livieris, S. Stavroyiannis, E. Pintelas, and P. Pintelas, "A novel validation framework to enhance deep learning models in time-series forecasting," *Emerging applications of Deep Learning and Spiking ANN*, vol. 32, no. 1803.08375, pp. 17149—-17167, 2020.

[83] X. Ma, P. Karkus, D. Hsu, and W. Lee, "Particle filter recurrent neural networks," *arXiv:1905.12885*, vol. [cs.LG], pp. 1–16, 2019.

[84] M. Maggiolo and G. Spanakis, "Autoregressive convolutional recurrent neural network for univariate and multivariate time series prediction," *arXiv:1903.02540*, vol. [cs.LG], pp. 1–8, 2019.

[85] M. Mäkinen, J. Kanniainen, M. Gabbouj, and A. Iosifidis, "Forecasting of jump arrivals in stock prices: New attention-based network architecture

using limit order book data," *arXiv:1810.10845*, vol. [q-fin.TR], pp. 1–29, 2018.

[86] S. Makridakis, R. J. Hyndman, and F. Petropoulos, "Forecasting in social settings: The state of the art," *International Journal of Forecasting*, vol. 36, pp. 15–28, 2020.

[87] W. Marc, A. Marc, and M. Wolfgang, "Visualizing time-series on spirals," in *IEEE Symposium on Information Visualization 2001 (INFOVIS'01)*, IEEE, 2001.

[88] A. Mathew, P. Amudha, and S. Sivakumari, "Deep learning techniques: An overview," *Advanced Machine Learning Technologies and Applications*, vol. 1141, 2021.

[89] S. Meisenbacher, M. Turowski, K. Phipps, M. Rätz, MüKlerk, V. Hagenmeier, and R. Mikut, "Review of automated time series forecasting pipelines," *arXiv 2022.01712*, vol. [cs.LG], pp. 1–32, 2022.

[90] A. S. Mihaita, H. Li, Z. He, and M.-A. Rizoiu, "Motorway traffic flow prediction using advanced deep learning," *arXiv:1907.06356*, vol. [cs.LG], pp. 1–10, 2018.

[91] T. Mikolov, A. Joulin, S. Chopra, M. Mathieu, and M. Ranzato, "Learning longer memory in recurrent neural networks," in *International Conference on Learning Representations—Workshop Track*, (San Diego, Cal), pp. 1–9, ICLR, 2015.

[92] M. Miwa and M. Bansal, "End-to-end relation extraction using LSTMs on sequences and tree structures," *arXiv:1601.00770*, vol. [cs.CL], pp. 1–13, 2016.

[93] G. Mohamed, Z. Arkady, and K. Shonali, "Mining data streams: A review," *SIGMOD Record*, vol. 34, no. 10, pp. 18–26, 2005.

[94] S. Mohseni, N. Zarei, and E. D. Ragan, “A multidisciplinary survey and framework for design and evaluation of explainable AI systems,” *ACM Transactions on Interactive Intelligent Systems*, vol. 11, no. 24, pp. 1–45, 2021.

[95] A. Mujika and F. Meier, “Fast-slow recurrent neural networks,” in *31st Conference on Neural Information Processing Systems*, (Long Beach, Ca), pp. 5915–5924, NIPS, 2017.

[96] L. Murukesan, M. Murugappan, M. Iqbal, and K. Saravanan, “Machine learning approach for sudden cardiac arrest prediction based on optimal heart rate variability features,” *Journal of Medical Imaging and Health Informatics*, vol. 4, pp. 521–532, 2014.

[97] K. M. Myat and S. S. M. Win, “Analysis of outlier detection on structured data,” in *Proceedings of 2020 the 10th International Workshop on Computer Science and Engineering*, (Yangon, Myanmar), pp. 16–21, 2020.

[98] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, and E. Muharemagic, “Deep learning applications and challenges in big data analytics,” *Journal of Big Data*, vol. 2, no. 1, pp. 1–21, 2015.

[99] S. Narejo and E. Pasero, “Meteonowcasting using deep learning architecture,” *International Journal of Advanced Computer Science and Applications*, vol. 8, 2021.

[100] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, “Activation functions: Comparison of trends in practice and research for deep learning,” *arXiv:1811.03378*, vol. [cs.LG], no. 10, pp. 1–20, 2018.

[101] M. Peter, T. Diekötter, and K. Kremer, “Participant outcomes of biodiversity citizen science projects: A systematic literature review,” *Sustainability*, vol. 11, no. 10, pp. 1–18, 2019.

[102] E. Philippe and A. Carlos, "Time-series data mining," *ACM Computing Surveys (CSUR)*, vol. 45, pp. 18–26, 2012.

[103] P. P. Phyo and Y. Byun, "Deep learning for time series forecasting: A survey," *Symmetry*, vol. 13, pp. 1943–57, 2021.

[104] C. Pickering, J. Grignon, R. Steven, D. Guitart, and J. Byrne, "Publishing not perishing: how research students transition from novice to knowledgeable using systematic quantitative literature reviews," *Studies in Higher Education*, vol. 40, no. 10, pp. 1756–1769, 2015.

[105] X. Qian, "Financial series prediction: Comparison between precision of time series models and machine learning methods," *arXiv:1706.00948*, vol. [cs.LG], pp. 1–9, 2017.

[106] H. Qin, "Comparison of deep learning models on time series forecasting:a case study of dissolved oxygen prediction," *arXiv:1911.08414*, vol. [eess.SP], pp. 1–16, 2019.

[107] Y. Riahi and S. Riahi, "Big data and big data analytics: concepts, types and technologies," *International Journal of Research and Engineering*, pp. 1009–1032, 2018.

[108] A. Romanov, M. DeArteaga, H. M. Wallach, J. T. Chayes, C. Borgs, A. Chouldechova, S. C. Geyik, K. Kenthapadi, A. Rumshisky, and A. T. Kalai, "What's in a name? Reducing bias in bios without access to protected attributes," in *Conference of the North American Chapter of the Association for Computer Linguistics*, (Mineapolis, MN), pp. 1–10, 2019.

[109] J. D. Roode, *Implications for Teaching of a Process-based Research Framework for Information Systems*. Orlando, Florida, 1993.

[110] K. H. Rosen, *Discrete Mathematics and Its Applications*. New York, NY: McGraw-Hill Education, 8 ed., 2019.

[111] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, and S. Valaee, "Recent advances in recurrent neural networks," *arXiv:1801.01078 [cs.NE]*, vol. 1801.01078 [cs.NE], pp. 1–21, 2017.

[112] M. Sardelicha and S. Manandhar, "Multimodal deep learning for short-term stock volatility prediction," *arXiv:1812.10479*, vol. [q-fin.ST], pp. 1–40, 2018.

[113] I. H. Sarker, "Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions," *Springer Nature Computer Science*, vol. 2, no. 420, pp. 1–20, 2021.

[114] I. Sarker, "Machine learning: Algorithms, real-world applications research directions," *SN Computer Science*, pp. 1–27, 2021.

[115] S. Sarmah, "Concept of artificial intelligence, its impact and emerging trends," *International Research Journal of Engineering and Technology*, vol. 6, no. 11, pp. 2164–2019, 2019.

[116] M. Schaefer, F. Wanner, R. Kahl, L. Zhang, and T. Schreck, "A novel explorative visualization tool for financial time series data analysis," in *The Third International UKVAC Workshop on Visual Analytics*, (University College London, UK), pp. 1–4, 2011.

[117] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[118] K. Schwab, *The Fourth Industrial Revolution*. New York, NY: Crown Business, 2017.

[119] S. Sengupta, S. Basak, P. Saikia, S. Paul, V. Tsalavoutis, F. D. Atiah, V. Ravi, and R. A. Peters, "A review of deep learning with special empha-

sis on architectures, applications and recent trends," *arXiv*, vol. [cs.LG], no. 1905.13294, pp. 1–29, 2019.

[120] A. Shrestha and A. Mahmood, "Review of deep learning algorithms and architectures," *IEEE Access*, vol. 7, pp. 53040–65, 2019.

[121] K. Siau and W. Wang, "Artificial intelligence (AI) ethics: Ethics of AI and ethical AI," *Journal of Database Management*, vol. 31, no. 2, pp. 74–87, 2020.

[122] V. Sima, I. Gheorghe, J. Subic, and D. Nancu, "Influences of the industry 4.0 revolution on the human capital development and consumer behavior: A systematic review," *Sustainability*, vol. 12, pp. 1–28, 2020.

[123] D. Smihula, "Long waves of technological innovations," *Studia Politica Slovaca*, vol. 2, pp. 50–68, 2011.

[124] K. Sokol and P. Flach, "Explainability fact sheets: A framework for systematic assessment of explainable approaches," in *Fairness Accountability and Transparency (FAT20)*, (Barcelona, Spain), pp. 1–22, ACM, 2020.

[125] W. Souma, I. Vodenska, and H. Aoyama, "Enhanced news sentiment analysis using deep learning methods," *Journal of Computational Social Science*, vol. 2, pp. 33–46, 2019.

[126] C. J. Spoerer, P. McClure, and N. Kriegeskorte, "Recurrent convolutional neural networks: A better model of biological object recognition," *Front. Psychol.*, vol. 8, no. 1551, pp. 2–14, 2017.

[127] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[128] A. Strauss and J. M. Corbin, *Grounded Theory in Practice*. Thousand Oaks, Ca: SAGE Publications, 1997.

[129] H. Suresh and J. V. Guttag, "A framework for understanding sources of harm throughout the machine learning life cycle," in *Equity and Access in Algorithms, Mechanisms, and Optimization '21*, (NY, USA), pp. 1–9, 2021.

[130] K. Takamura and S. Yamane, "Improving minimal gated unit for sequential data," *2019 IEEE 8th Global Conference on Consumer Electronics (GCCE)*, pp. 696–698, 2019.

[131] Q. Tang, M. Yang, and Y. Yang, "ST-LSTM: A deep learning approach combined spatio-temporal features for short-term forecast in rail transit," *Journal of Advanced Transportation*, vol. 2019, pp. 1–8, 2019.

[132] C. A. Tapia Cortez, J. Coulton, C. Sammut, and S. Saydam, "Determining the chaotic behaviour of copper prices in the long-term using annual price data," *Palgrave Communications*, vol. 8, pp. 1–12, 2018.

[133] D. R. Thomas and I. D. Hodges, *Designing and Managing Your Research Project: Core Skills for Social and Health Research*. Thousand Oaks, Ca: SAGE Publications, 2010.

[134] J. Torres, D. Hadjout, A. Sebaa, F. Martínez-Alvarez, and A. Troncoso, "Deep learning for time series forecasting: A survey," *Big Data*, vol. 9, pp. 1–55, 2020.

[135] M. Travers, *Qualitative Research Through Case Studies*. Thousand Oaks, Ca: SAGE Publications, 2001.

[136] V. Umayaparvathi and K. Iyakutti, "Automated feature selection and churn prediction using deep learning models," *International Research Journal of Engineering and Technology*, vol. 4, pp. 1846–1854, 2017.

[137] E. van Laar, A. van Deursen, J. van Dijk, and J. de Haan, "The relationship between 21st century skills and digital skills: A systematic literature review," *Computers in Human Behaviour*, vol. 72, pp. 577–588, 2017.

[138] J. J. vanWijk and E. R. vanSelow, "Cluster and calendar based visualization of time series data," in *IEEE Symposium on Information Visualization (INFOVIS'99)*, (Yangon, Myanmar), pp. 1–6, 1999.

[139] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Łukasz. Kaiser, and I. Polosukhin, "Attention is all you need," in *31st Conference on Neural Information Processing Systems*, (Long Beach, CA), pp. 1–11, 2017.

[140] J. R. Venable, J. Pries-Heje, and R. L. Baskerville, "Choosing a design science research methodology," in *Australasian Conference on Information Systems (ACIS) 2017 Proceedings*, pp. 1–12, 2017.

[141] C. Vitor, T. Luís, and S. Carlos, "Machine learning vs statistical methods for time series forecasting: Size matters," *arXiv:1909.13316*, vol. [stat.ML], pp. 1–9, 2019.

[142] W. W Jiang, L. Ling, D. Zhang, R. Lin, and L. Zengg, "A time series forecasting model selection framework using CNN and data augmentation for small sample data," *Research Square*, pp. 1–21, 2022.

[143] K. Wang, C. Gou, Y. Duan, L. Yilun, X. Zheng, and F.-Y. Wang, "Generative adversarial networks: Introduction and outlook," *Journal of Automatica Sinica*, vol. 4, pp. 588–598, 2017.

[144] W. Wei, H. Wu, and H. Ma, "An autoencoder and LSTM-based traffic flow prediction method," in *Sensors*, (Beijing), pp. 1–16, 2016.

[145] J. Whittlestone, R. Nyrup, A. Alexandrova, and S. Cave, "The role and limits of principles in ai ethics: Towards a focus on tensions," in *Proceedings of the Conference on AI, Ethics, and Society (AIES '19)*, (Honolulu, HI), pp. 195–200, 2019.

[146] C. J. Willmott, K. Matsuura, and S. M. Robeson, "Ambiguities inherent in sums-of-squares-based error statistics," *Atmospheric Environment*, vol. 43, pp. 749–752, 2009.

[147] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 67–82, 1997.

[148] M. Woschank, E. Rauch, and H. Zsifkovits, "A review of further directions for artificial intelligence, machine learning, and deep learning in smart logistics," *Sustainability*, vol. 19, pp. 1–23, 2020.

[149] J. Xiao, H. Li, X. Wang, and S. Yuan, "Traffic peak period detection from an image processing view," *Journal of Advanced Transportation*, vol. 2018, pp. 1–10, 2018.

[150] Y. Xiao and M. Watson, "Guidance on conducting a systematic literature review," *Journal of Planning Education and Research*, vol. 39, no. 1, pp. 93–112, 2019.

[151] L. Xinyi, L. Yinchuan, Y. Hongyang, Y. Liuqing, and L. Yang, "DP-LSTM: Differential privacy-inspired LSTM for stock prediction using financial news," in *33rd Conference on Neural Information Processing Systems*, (Vancouver, Canada), pp. 1–9, ACM, 2019.

[152] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu, "Explainable AI: A brief survey on history, research areas, approaches and challenges," in *Natural Language Processing and Chinese Computing*, (Dalian, China), pp. 563–574, 2019.

[153] S.-S. Yu, S.-W. Chu, C.-M. Wang, Y.-K. Chan, and T.-C. Chang, "Two improved k-means algorithms," *Applied Soft Computing*, vol. 68, pp. 1–13, 2017.

[154] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," *arXiv:1610.00081*, vol. [cs.AI], pp. 1–8, 2017.

[155] Q. Zhang, R. Luo, Y. Yang, and Y. Liu, "Benchmarking deep sequential models on volatility predictions for financial time series," *arXiv:1811.03711*, vol. [cs.LG], pp. 1–14, 2018.

[156] F. Zhou, H.-M. Zhou, Z. Yang, and L. Yang, "EMD2FNN: A strategy combining empirical mode decomposition and factorization machine based neural network for stock market trend prediction," *Expert Systems with Applications*, vol. 115, pp. 136–151, 2019.

[157] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," *arXiv:2012.07436*, vol. [cs.LG], pp. 1–15, 2021.

Appendices

Appendix A

Selected Papers that Analyse Sequential Patterns

Table A.1 lists papers that analyse irregular sequences, found in a systematic literature review.

Table A.1: Behaviour of Irregular Sequential Patterns

List	Sequential datasets	Sequential artefacts	Evaluation metrics
1.	Daily exchange rates data from Australia, British Canada, Switzerland, China, Japan, New Zealand and Singapore from 1990 to 2016 by (Lai et al. 2017)	Attention based frameworks (At-LSTM)	Agreement Cohen's Kappa
2.	NASDAQ stock price dataset by Qin Y. et al (2017)	Hybrid attention based frameworks (At-LSTM)	Average negative log-likelihood (NLL)
3.	Appliances energy prediction dataset by Candanedo L. et al (2017)	Autoregressive models (AR)	Computational time spent by a model
4.	Air quality prediction (AIR De Vito S. et al (2008)	Hybrid autoregressive model	Copy memory loss and memory footprints
5.	Weather dataset by Liang X, et al (2015)	Back-propagation neural networks (BPNN)	Correlation coefficient (R^2)
6.	European G´EANT traffic data points	Bayesian based algorithms	Cosine proximity
7.	Telecom datasets from Cell2Cell	Bidirectional (Bi) based frameworks	Dynamic time warping (DTW)
8.	Crowd Analytix dataset	Bidirectional combined with attention (Att) mechanism	Empirical correlation coefficient (ECORR)
9.	Unstable social media dataset from Persian movie reviews from 2014 to 2016.	Bidirectional combined with GRU (BiGRU) and LSTM (BiLSTM)	F-Measure
10.	Standard benchmark ACL18 data for NASDAQ and NYSE markets from Jan 2014 to Jan 2016 by (Xu and Cohen, 2018)	Capsule neural network (CapsNet)	Hit ratio
11.	Standard KDD17 dataset by (Zhang et al., 2017)	Convolutional neural networks (CNNs)	Matthews correlation coefficient (MCC)
12.	Stock index data (DOW 30, S&P 500 and NASDAQ)	Deep autoencoder (DA)	Max absolute percentage error (MaxAPE)
13.	Ultra-high-frequency order book data from 5 liquid U.S NASDAQ's (Google, Microsoft, Apple, Intel and Facebook) financial stocks	Deep Bayesian neural networks (BNN)	Mean absolute error (MAE)
14.	Financial stock indices dataset (S&P 500, Dow Jones Industrial Average (DJIA), NASDAQ and Russel 2000)	Deep differential privacy-inspired LSTM (DP- LSTM)	Mean absolute percent errors (MAPE)
15.	Historical financial price data from Crypto-Compare for Bitcoin, Ethereum and Monero	Deep feed forward neural network (FFNN)	Mean absolute scaled error (MASE)
16.	Social data from publicly available social platforms (GitHub and Reddit).	Deep sequential spatio-temporal residual neural network (ST-ResNet)	Mean directional accuracy (MDA)
17.	Standard Penn Treebank (PTB) data	Denoising autoencoder (DAE)	Maximum error (ME)
18.	Standard WikiText-103 (WT103) data	Transformer neural network	Mean Error Percent (MEP)
19.	Financial news dataset from Reuters and Bloomberg on 473 Standard & Poor's 500 listed companies (Google, Amazon, Cisco, Microsoft, Apple, Intel, IMB, AMD, NVidia, Qualcomm, Walmart)	Transformer neural network combined with RNN and CNN	Mean prediction accuracy (MPA)
20.	Sydney motorway traffic flow data of 2017	TrellisNet	Mean relative error (MRE)
21.	Financial stock dataset from Bank of China (601988), Vanke A (000002) and Kweichou Moutai (600519).	Differentiable architecture (DARTS)	Mean square error (MSE)

Table A.1: Behaviour of Irregular Sequential Patterns (Cont.)

List	SOTA Sequential datasets	Sequential artefacts	Evaluation metrics
22.	UCI daily grocery sales datasets	Dilated recurrent neural network (DilatedRNN)	Mean squared percentage error (MSPE)
23.	Univariate (Daily values for Melbourne's minimum temperature and Zurich Sunspot) datasets	Dilated temporal convolutional network (TCN)	Mean symmetric mean absolute percentage error (SMAPE)
24.	Multi-variate (Energy production for 10 different photovoltaic power plants in California and SML2010 dataset containing internal and external measurements in a domestic house) datasets	Dual self-attention network (DSANet)	Median MASE
25.	Real time Yangtze River dissolved oxygen time series data automatically recorded from 2012 to 2016.	Dual-stage attention based recurrent neural network (DA-RNN)	Median SMAPE
26.	4 years sequential time series Uber dataset for 8 large cities in U.S. and Canada (Atlanta, Boston,Chicago, Los Angeles, New York City, San Francisco, Toronto, and Washington D.C)	Elmann recurrent neural networks (ERNN)	Normalized deviation (ND)
27.	Trajectory data (TaxiBJ from taxicab GPS data and meteorology data in Beijing (2013 – 2016) and Trajectory data (BikeNYC) from NYC bike system (2014)	Extension GARCH (EGARCH)	Normalized RMSE (NRMSE)
28.	Historical S&P 500 stock price data from the Yahoo Finance	Fast-slow recurrent neural network (FS-RNN)	Normalized root mean squared error (NRMSE)
29.	46. NLP sentimental news dataset from financial domain (CNBC.com, Reuters.com, WSJ.com, Fortune.com and Wall Street Journal)	Feed forward neural networks (FFNN)	Precision F1 score
30.	Daily revenue data from five gas stations companies	Generative adversary neural networks (GAN)	Precision jumps recall
31.	45 datasets of different time series lengths from random real world application domains which encompass Meteorology, Astronomy, Physiology, Acoustics, and others	Gated recurrent unit (GRU)	Proportion of variance R^2
32.	Real-world JD.com of China's (JD-demand and JD-shipment) data	Gated recurrent unit with hybrid architecture	Rank MASE
33.	Electricity consumption dataset for servers in a data centre by Flunkert et al.(2017)	Gaussian models (GP)	Rank SMAPE
34.	Traffic flows data by Lv et al. (2015)	General regression neural network (GRNN)	Regression coefficient (R^2)
35.	Internet traffic dataset for internet companies' by Kaggle (2017))	Generalized autoregressive conditional heteroscedasticity (GARCH)	Root mean square error (RMSE)
36.	Daily exchange rates data from Australia, British Canada, Switzerland, China, Japan, New Zealand and Singapore from 1990 to 2016 by (Lai et al. 2017)	Generalized linear regression (GLM))	Root mean squared logarithmic error (RMLSE)
37.	NASDAQ stock price dataset by Qin Y. et al (2017)	Hierarchical multi-scale recurrent neural network (HM-RNN)	Root mean squared percentage error (RMSPE)
38.	Appliances energy prediction dataset by Candanedo L. et al (2017)	Hierarchical neural network architecture	Root relative squared error (RRSE)
39.	Air quality prediction (AIR De Vito S. et al (2008)	Independently recurrent neural network (IndRNN)	Symmetric mean absolute percentage error (SMAPE)

Table A.1: Behaviour of Irregular Sequential Patterns (Cont.)

List	SOTA Sequential datasets	Sequential artefacts	Evaluation metrics
40.	Weather dataset by Liang X, et al (2015)	Large feedforward neural network (LFNN)	Trading profitability measures (cumulative return (CR), annualized return (AR), annualized volatility (AV), Sharpe ratio and (SR) and draw-down (DD))
41.	European G´EANT traffic data points	Logistic regression (LR)	
42.	Telecom datasets from Cell2Cell	Long short-term memory (LSTM)	
43.	Crowd Analytix dataset	Memory-based ordinal regression deep neural networks (MOrdReD)	
44.	Unstable social media dataset from Persian movie reviews from 2014 to 2016.	Momentum models (MOM)	
45.	Standard benchmark ACL18 data for NASDAQ and NYSE markets from Jan 2014 to Jan 2016 by (Xu and Cohen, 2018)	Mean reversion models (MR)	
46.	Standard KDD17 dataset by (Zhang et al., 2017)	Multilayer perception (MLP)	
47.	Stock index data (DOW 30, S&P 500 and NASDAQ)	Multivariate adaptive regression splines (MARS)	
48.	Ultra-high-frequency order book data from 5 liquid U.S NASDAQ's (Google, Microsoft, Apple, Intel and Facebook) financial stocks	Neural architecture search (NAS)	
49.	Financial stock indices dataset (S&P 500, Dow Jones Industrial Average (DJIA), NASDAQ and Russel 2000)	Particle filter recurrent neural networks (PF-RNNs)	
50.	Historical financial price data from Crypto-Compare for Bitcoin, Ethereum and Monero	Quasi-recurrent neural network (QRNN)	
51.	Social data from publicly available social platforms (GitHub and Reddit).	Radial basis neural networks (RBFNN)	
52.	Standard Penn Treebank (PTB) data	Random Classifier (RC)	
53.	Standard WikiText-103 (WT103) data	Random connectivity LSTM (RCLSTM)	
54.	Financial news dataset from Reuters and Bloomberg on 473 Standard & Poor's 500 listed companies (Google, Amazon, Cisco, Microsoft, Apple, Intel, IMB, AMD, NVidia, Qualcomm, Walmart)	Random forest (RF)	
55.	Sydney motorway traffic flow data of 2017	Recurrent highway network (RHN)	
56.	Financial stock dataset from Bank of China (601988), Vanke A (000002) and Kweichou Moutai (600519).	Recurrent neural network (RNN)	
57.	UCI daily grocery sales datasets	Rule-based regression (RBR)	
58.	Univariate (Daily values for Melbourne's minimum temperature and Zurich Sunspot) datasets	Sequence to sequence (Seq2seq) architectures or encoder-decoder models	
59.	Multi-variate (Energy production for 10 different photovoltaic power plants in California and SML2010 dataset containing internal and external measurements in a domestic house) datasets	Skip recurrent neural network (SkipRNN)	

Table A.1: Behaviour of Irregular Sequential Patterns (Cont.)

List	SOTA Sequential datasets	Sequential artefacts	Evaluation metrics
60.	Real time Yangtze River dissolved oxygen time series data automatically recorded from 2012 to 2016.	Small feedforward neural network (SFNN)	
61.	4 years sequential time series Uber dataset for 8 large cities in U.S. and Canada (Atlanta, Boston,Chicago, Los Angeles, New York City, San Francisco, Toronto, and Washington D.C)	Spatio-temporal long short-term network (ST-LSTM)	
62.	Trajectory data (TaxiBJ from taxicab GPS data and meteorology data in Beijing (2013 – 2016) and Trajectory data (BikeNYC) from NYC bike system (2014)	Squares support vector machine regression (LS-SVMR).	
63.	Historical S&P 500 stock price data from the Yahoo Finance	StockNet which uses a variational autoencoder (VAE)	
64.	46. NLP sentimental news dataset from financial domain (CNBC.com, Reuters.com, WSJ.com, Fortune.com and Wall Street Journal)	Support vector machine regression (SVMR)	
65.	Daily revenue data from five gas stations companies	Support vector machines (SVM)	
66.	45 datasets of different time series lengths from random real world application domains which encompass Meteorology, Astronomy, Physiology, Acoustics, and others	Temporal convolutional networks (TCN)	
67.	Real-world JD.com of China's (JD-demand and JD-shipment) data	Transformer networks	
68.	CIF 2016 Forecasting Competition Dataset	TrellisNet	
69.	NN5 Forecasting Competition Dataset	Variational LSTM	
70.	M3 Forecasting Competition Dataset		
71.	M4 Forecasting Competition Dataset		
72.	CIF 2016 Forecasting Competition Dataset		
73.	NN5 Forecasting Competition Dataset		

The full code and results can be found on GitHub at `https://bit.ly/3w622ok`

Appendix B

Irregular Sequential Patterns Identified by the SeLFISA Framework

Table B.1 lists 16 irregular sequences, found in a systematic literature review.

Table B.1: Behaviour of Irregular Sequential Patterns

List	Name	Length	Number of IQR Outliers
1	S&P500 from Jan–Dec-2011 [156] and [22]	251	0
2	NASDAQ from Jan–Dec-2011 [156] and [22]	251	6
3	DJI from 01-2008–12-2009 [22]	504	0
4	NASDAQ from 01-2008–12-2009 [22]	504	0
5	S&P from 01-2008–12-2009 [22]	504	0
6	Monero Crypto Currency Daily Rates from 2015–2018 [41]	1208	0
7	DJI from 10-2010–09-2016 [12] and [22]	1513	0
8	S&P500 from 10-2010–09-2016 [12] and [22]	1696	0
9	CAD_USD Daily Exchange Rate from 1990–2016 [23]	5000	0
10	CNY-USD Daily Exchange Rate from 1990–2016 [23]	5000	0
11	NZD_USD Daily Exchange Rate from 1990–2016 [23]	5000	0
12	SGD_USD Daily Exchange Rate from 1990–2016 [23]	5000	0
13	JPY_USD Daily Exchange Rate from 1990–2016 [23]	5000	24
14	AUD_USD Daily Exchange Rate from 1990–2016 [23]	5906	0
15	GBP_USD Daily Exchange Rate from 1990–2016 [23]	6135	639
16	SwiFranc_USD DailyExchange Rate from 1990–2016 [23]	7015	0

All the experimental code is given in the Jupyter Notebook files on the GitHub website at: `https://github.com/Dandajena/SDA/`.

Accessible at:

`https://github.com/Dandajena/SDA/blob/master/Database.csv.`

Appendix C

Variables identified and used in the SeLFISA Framework

Table C.1: Variables for SeLFISA Framework

No.	Variable Name	Feature Count	Feature List	Category	Implementation Remarks
1.	Domain of research	1	Deep Learning Framework for the Prediction of Discrete Irregular Patterned Sequential Environments	Combinatorial in identification and selection but permutative in implementation	This is the initial stage driven by the research challenges in sequential modelling.
2.	Domain Challenges classes	3	Major classes are within frameworks, datasets and evaluation	Combinatorial	Addressed all through a framework
3.	Specific prediction challenges	11	consistency or inconsistency, reliability, repeatability, straightforwardness transparency, explainability, sensitivity to outliers and extreme values, lack of well-established, explainable literature, poor comprehensive comparison analysis, lack of multidimensional performance evaluation on single framework, dominance of accuracy metrics, computational complexity.	Combinatorial	Focused on those that distort performance robustness
4.	Existing research sources	400	400 articles were the initial sources of literature research	Combinatorial	33 articles created nucleus articles based on a matrix specific selection, inclusion and analysis criteria.
5.	Implementation AI Platforms	8	Google AI Cloud Platform, Amazon AI Services (Amazon SageMaker), Google Cloud AutoML, MATLAB, Microsoft Azure (Machine Learning Studio), IBM Watson Machine Learning and Anaconda Enterprise.	Combinatorial	Anaconda Enterprise was our platform of choice because its open source versatility platform with a Python based IDE compatibility with many languages and notebooks. This avail the entire life cycle which prepare, build, validate, deploy and monitor AI models.
6.	Implementation Languages	5	Python, Java, Lisp, Prolog and R Programming	Combinatorial	Python was our language of choice since it is easy to learn, deploy and it integrates efficiently with a wide range of syntax

Table C.1: Variables for SeLFISA Framework (Cont.)

No.	Variable Name	Feature Count	Feature List	Category	Implementation Remarks
7.	Implementation environments	3	Jupyter Notebook, Kaggle and Google Colaboratory	Combinatorial	We created our environment based on Jupyter Notebook because of its interactive features that can mix code, script, inline graphs, interactive figures, into a shareable web document.
8.	Libraries and modules	22	Regular expressions, garbage collectors, operating systems, system-specific parameters, time, spacy, Keras, pickle, requests, math time, Matplotlib, NumPy, Pandas, progress bar TQDM library, math log2, Seaborn, sklearn, metrics, TensorFlow.	Combinatorial	We choose more than 22 libraries and modules that are already written in Python to set routines and functions. These libraries and modules were expanded from internal module through an "from main library import internal library"
9.	Computational Environment	4	High Performance Computing from CHPC, Google Cloud, Kaggle and On-Premise Core i7 Laptop.	Combinatorial on design and Permutative on installation and executions	CHPC High Performance Computing combined through on-Prem Laptop.
10.	Datasets domain	8	Weather, energy, finance, weather, astronomy, transportation, health and general domain benchmark datasets.	Combinatorial	Finance domain was our primary choice.
11.	Datasets	73	The 8 domains from 33 nucleus articles produced 73 accessible datasets.	Combinatorial	2 Financial market-daily currency exchange datasets were selected with high levels of irregular discrete properties.
12.	Selected dataset features	6	Date, price, open, high, low and change	Combinatorial	Pre-process before training.
13.	Data exploratory processes	10	More than 10 activities in the form of data wrangling, description, data pre-processing, data munching, data cleaning, and exploratory data analysis	Permutative	Pre-process before training.
14.	Dataset splitting ratio	3	Training, validation and testing (80%–20%, 90%–10% and 70%–30%) and respective window length to determine prediction horizon.	Combinatorial on ratio selection and permutative on execution	Pre-process before training. A window length of an array of 100 inputs were used to determine the next outcome.
15.	Algorithms and models	335	These architectures produced 335 algorithms and models based on statistical, probabilistic, gated, attention, bidirectional, general neural, encoders and decoders, transformer, vanilla, hybrid, ensemble, convolutional, classification and other	Combinatorial on selection and Permutative during execution	Focused on best performing through experimental deployment and application
16.	Algorithms learning technique	4	Supervised, Semi supervised, Unsupervised and Reinforcement	Combinatorial but learning process is permutative	Pre-process before training.
17.	Algorithms analysis types	4	Regression, classification, clustering and association	Combinatorial	Regression analysis was applied
18.	Evaluation criteria category	2	Quantitative and Qualitative	Combinatorial	Pre-process before training

Table C.1: Variables for SeLFISA Framework (Cont.)

No.	Variable Name	Feature Count	Feature List	Category	Implementation Remarks
19.	Evaluation criteria	9	Consistency, efficiency, accuracy, visualization sharpness, computational complexity. repeatability, straightforwardness and explainability	Combinatorial	Pre-process before training
20.	Activation function	24	ReLU. Leaky ReLU, Maxout, Tanh, linear / identity, Binary step, piece wise linear, Sigmoid, Complementary log-log, Bipolar, Bipolar Sigmoid, LeCun's Tanh, Hard Tanh, Absolute, Rectifier, Smooth Rectifier, Logit, Probit, Cosine, Softmax, Maxout, Multiquadratic and Inverse Multiquadratic.	Pre-process before training	
21.	Evaluation metrics category	3	Regression, binary classification and multi-class classification	Combinatorial	Regression was the choice of the research
22.	Evaluation metrics or Loss functions	12	Mean Error (ME),Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE),R Squared, Categorical Cross Entropy, Binary Cross Entropy ,Hinge Loss, Squared Hinge, Multi-Class Cross-Entropy Sparse Multiclass Cross-Entropy and Kullback Leibler Divergence.	Combinatorial	A hybrid approach was considered
23.	Weights	1	Randomly allocated using parameter optimisation techniques	Permutative	Automatically assigned
24.	Bias	1	Automatically selected using libraries.	Permutative	Guided by other factors
25.	Net input	1	Depend on the nature of the input features of the dataset.	Permutative	Guided by other factors
26.	Number of Neurons	1	Determined by a specific mathematical formula	Permutative	Guided by other factors
27.	Number of layers	1	Determined by a specific mathematical formula	Permutative	Guided by other factors
28.	Interconnections	2	Feed-forward and recurrent Combinatorial and permutative	Guided by other factors	
29.	Training process	2	Backpropagation and Backpropagation through time		Automatically implemented through Python libraries.
	Number of subvariables	≥ 410			

Appendix D

Visualisation of Models Predicting JPY vs. USD

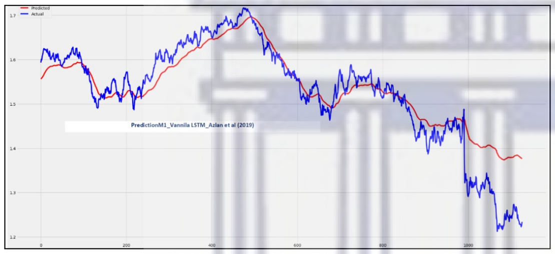


Figure D.1: Model 1
LSTM (32) + Dropout (0.2) + Dense (1) suggested by Azlan et al(2019) [8], Li et. al (2019) [90] and Glenski et. al (2019) [41]

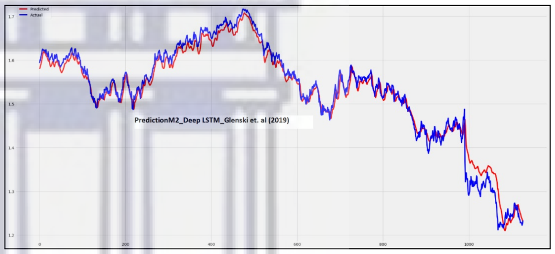


Figure D.2: Model 2
LSTM(32) + LSTM(64)+Dropout(0.2) + LSTM(128) + Dropout(0.5) + Dense(1) by Deep LSTM Model based implemented by Glenski et. al (2019) [41] and Chalvatzisa et. al (2019) [22]

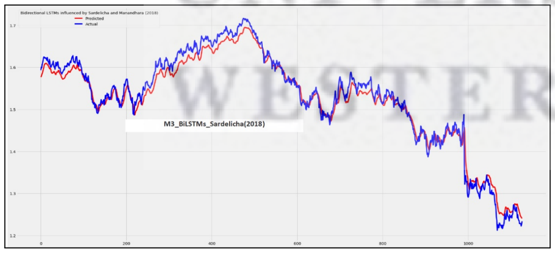


Figure D.3: Model 3
BiD(LSTM(50)) + Dense(10) + Dense(10) + Dense(1) influenced by Sardelicha and Manandhara (2018) [112]

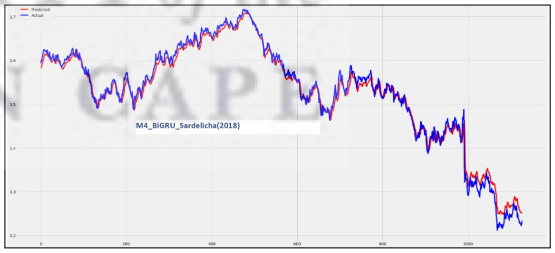


Figure D.4: Model 4
BiD(GRU(50)) + Dense(10) + Dense(10) + Dense(1) by Sardelicha and Manandhara (2018) [112]

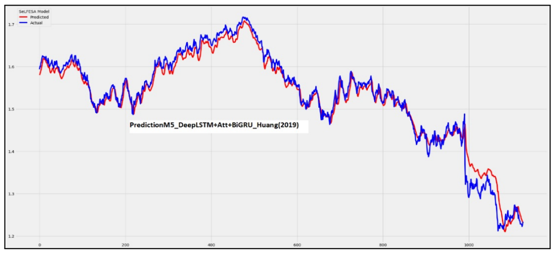


Figure D.5: Model 5
LSTM(100) + Dropout(100) + Attention(SeqSelfAttention) + LSTM(16) + Dense(10) + Dense(10) + Dense(1) by Deep LSTM by Huang (2019) [59]

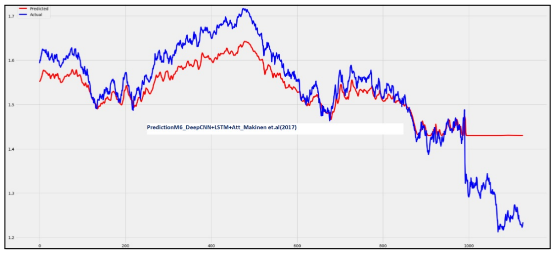


Figure D.6: Model 6
LSTM(32) + Conv1D(32) + Dropout(0.2) + Conv1D (16) + Conv1DTr(16) + Dropout(16) + Conv1DTr(32) + Conv1D (16) + AttSeqSelf(1) + LSTM(16) + Dropout(0.2) + Dense (1) by Makinen et. al (2017) [85]SeqSelf(1) + LSTM(16) + Dropout(0.2) + Dense (1) attention by Huang (2019) [59]

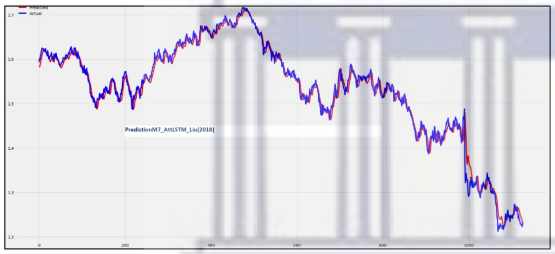


Figure D.7: Model 7
LSTM(32) + Dropout(100) + Attention (SeqSelf (32)) + LSTM (16) + Dense(10) + Dense(10) + Dense(1) by Liu (2018) [81]

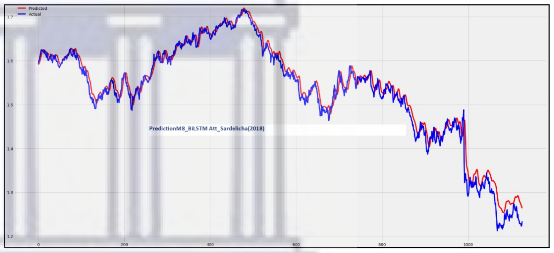


Figure D.8: Model 8
LSTM(32)+Dropout(0.2) + Attention (SeqSelf)(32) + Bidirection(LSTM(32)) + Bidirection(LSTM(32)) + Dense(10) + Dense(1) by by Sardelicha and Manandhara (2018) [112]

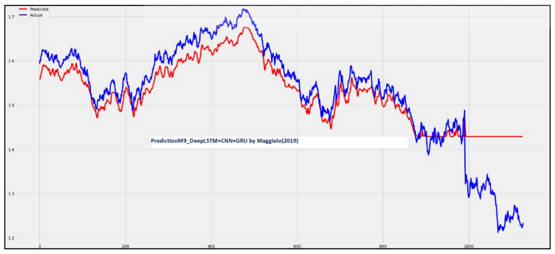


Figure D.9: Model 9
LSTM(32) + Conv1D(32) + Dropout(0.2) + Conv1D(16) + Conv1DTranspose(16) + Dropout(0.2) + Conv1DTranspose(32) + Conv1DTranspose(1) + GRU(32) + Dropout(0.5) + Dense(1) by Maggiolo and Spanakis (2019) [84]

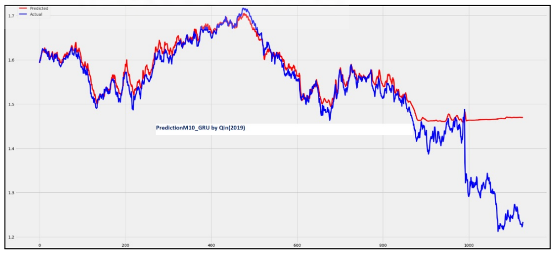


Figure D.10: Model 10
GRU(32) + GRU(64) + Dropout(0.2) + GRU(128) + Dense(1) by GRU by Qin(2019) [106]

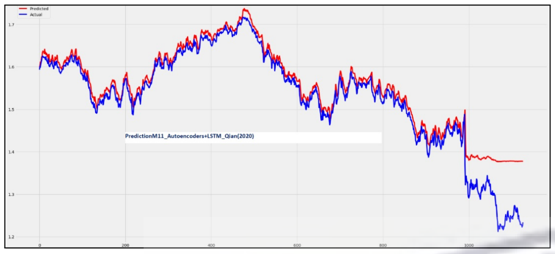


Figure D.11: Model 11
LSTM(32) + LSTM(64) + RepeatVector(64) + LSTM(64) + TimeDist(1) + LSTM(128) + Dropout(128) + Dense(1) by Qin(2019) [106]

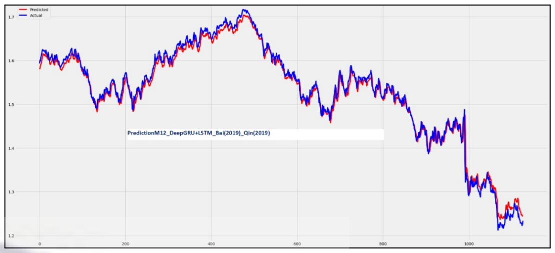


Figure D.12: Model 12
LSTM(50) + Dropout + LSTM(100) + Dropout (0.5)+ GRU(100) + LSTM(100) + Dropout(0.5) + LSTM(100) + Dropout(0.5) + Dense(100) + Dense(10) + Dense(10) + Dense(1) by Bai(2019) [10]

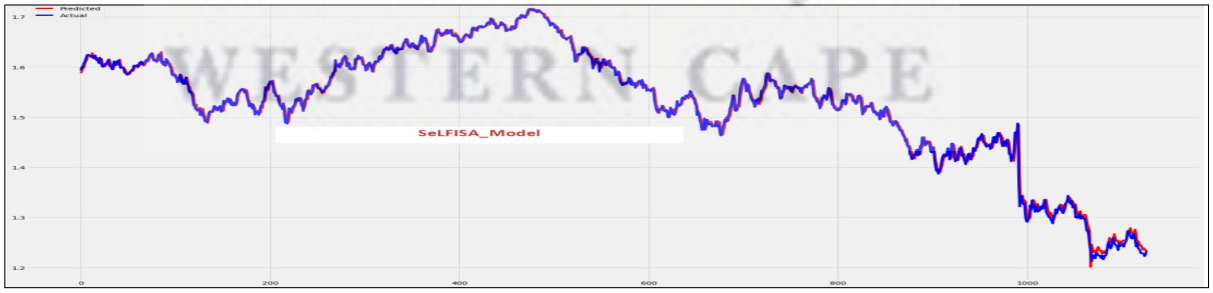


Figure D.13: SeLFISA Model
BiD(GRU(32)) + SeqSelfAtt(att_width=30) + Dropout(0.2) + BiD(LSTM(32)) + BiD(GRU(32)) + BiD(LSTM(32)) + BiD (GRU(32)) + LSTM (32) + GRU(32) + Dense(1)