Informatics Institute of Technology

In Collaboration With

The University of Westminster, UK

The University of Westminster, Coat of Arms

A Novel Approach to Time Series Forecasting using Liquid Time-constant Networks

A Project Proposal by

Mr. Ammar Raneez

W1761196 | 2019163

Supervised by

Mr. Torin Wirasingha

November 2022

This Project Proposal is submitted in partial fulfilment of the requirements for the BSc (Hons) Computer Science degree at the University of Westminster.

Contents

List of Tables

List of Figures

Acronyms

AI	Artificial Intelligence.
API	Application Programming Interface.
ARIMA	Autoregressive Integrated Moving Average.
BPTT	Back-Propagation Through Time.
BTC	Bitcoin.
CT-GRU	Continuous-time Gated Recurrent Unit.
CT-RNN	Continuous-time Recurrent Neural Network.
DL	Deep Learning.
GPU	Graphics Processing Unit.
LSTM	Long Short-Term Memory.
LTC	Liquid Time-constant.
ML	Machine Learning.
(s)MAPE	Symmetric Mean Absolute Product Error.
MASE	Mean Absolute Scaled Error.
MSE	Mean Squared Error.
N-BEATS	Neural Basis Expansion Analysis for interpretable Time Series
NLP	Natural Language Processing.
ODE	Ordinary Differential Equations.
POC	Proof-Of-Concept.
REST	Representational State Transfer.
RMSE	Root Mean Squared Error.
RNN	Recurrent Neural Network.
TS	Time Series.
SDE	Stochastic Differential Equations.

1. INTRODUCTION

In this document, the author aims to identify and provide the reader with an overview of the current issues in time series forecasting and highlight what a liquid time-constant network is and what it aims to solve. In detail, the author will define the problem and evaluate the necessary literature to arrive at a justifiable research gap and respective research challenges. The proposed methodology, deliverables, and work breakdown plan are also defined.

2. PROBLEM DOMAIN

2.1 Time series forecasting

TS forecasting is a significant business issue and an area where ML could create an impact. It is the foundation for contemporary business practices, including pivotal domains like customer management, inventory control, marketing, and finance. As a result, it has a comprehensive financial impact, with millions of dollars for subtle improvements in forecasting accuracy (Jain, 2017).

Having said that, although ML and DL have outperformed classical approaches for NLP and computer vision, the domain of TS still seems to be a struggle compared to classical statistical methodologies (Makridakis et al., 2018a;b). Most of the top-ranking methods in the M4 competition were ensembles of traditional statistical techniques (Makridakis et al., 2018b), while regular ML methods were nowhere near.

Therefore, this competition's winner was a hybrid model of LSTM and classical statistics (Smyl, 2020). Furthermore, they claimed that the only way to improve the accuracy of TS forecasting was by creating such hybrid models, which the author aspires to challenge in this research project.

2.2 Liquid Time-Constant (LTC) networks

LTCs are neural ODEs: hidden layers are not specified; instead, the derivative of hidden states is parameterized by neural networks (Chen et al., 2018). RNNs are effective algorithms for TS data modelling, if there exist continuous time hidden states determined by ODEs (Chen et al.,2018). Studies show that existing algorithms such as the CT-RNN (Funahashi and Nakamura 1993;

Rubanova, Chen and Duvenaud, 2019) and CT-GRU (Mozer, Kazakov and Lindsey, 2017) produce such performance. However, they have issues with expressivity and fixed behaviour once trained (Hasani et al., 2020). Therefore, the question arises: what would happen if there were unexpected changes to the characteristic of the inputs during inference? Additionally, these algorithms lose in generalization compared to even a simple LSTM network (Hasani et al., 2021), which raises another question, what is the point of defining a different and *'fancy'* approach if they cannot work well in real-world applications?

Hasani et al. state that LTCs can *"identify specialized dynamical systems for input features arriving at each time point"* (2020, p1). The ability to exhibit stable and bounded behaviour demonstrates that the proposed approach yields better expressivity than traditional implementations.

The LTC state and their respective time constant "*exhibit bounded dynamics and ensure the stability of the output dynamics*", which is a prominent factor when inputs increase relentlessly (Hasani et al., 2020).

2.3 Cryptocurrencies

The word 'crypto' has been an enormous buzzword recently, especially BTC. It has even come to the point where crypto and BTC are used interchangeably.

Cryptocurrencies are a fully decentralized digital currency form; it is a form of a peer-to-peer system without the need for a third party, thus enabling safer online transactions (S. Nakamoto, 2008). BTC is the first and the most popular digital currency to date, piquing many academic researchers' interest (Rahouti et al., 2018).

However, being at the forefront of the digital currency world, it has developed high volatility, making it difficult to predict future rates and a disadvantage for multiple investors (Kervanci and Akay, 2020). Despite that, recent advances in ML and statistics have shown acceptable results in the analysis and prediction of cryptocurrencies, yet the root cause of these algorithms persists: they are static.

3. PROBLEM DEFINITION

As of writing this report, there is no application of LTC networks in any domain since this novel neural ODE has only recently been announced (Hasani et al., 2020). Existing intelligent systems utilize more traditional approaches of neural nets developed some time ago.

Having mentioned that, most applications of ML available do perform quite well (ex: image classification, transfer learning, NLP), yet, as mentioned, the field of TS forecasting is subpar (Makridakis et al., 2018a;b). Existing TS forecasting algorithms cannot adapt to unforeseen changes in data streams; consequently, they could perform relatively poorly when used in areas of high volatility (in this case: the forecasting of BTC).

Building an algorithm inspired by the LTC architecture and its application on an ML domain that still can struggle could be the stepping stone for future intelligent systems by aiding further research. Additionally, as a supplement, it could provide hope to crypto investors for more straightforward predictions.

3.1 Problem statement

Existing TS forecasting systems cannot adapt to incoming data streams with unexpected changes and characteristics that are different from the data on which they were trained; implementing an algorithm capable of having the ‘liquid’ adaptability mentioned could be an advancement for more capable, accurate, and interpretable TS forecasting systems.

4. RESEARCH MOTIVATION

The field of AI, particularly neural networks, has been growing exponentially recently, alongside intriguing research. However, as mentioned by Hasani et al. (2020), the issue of networks being static and unable to adapt to varying characteristics could prove to be a limitation for the future of intelligent systems, TS in particular. Therefore, this research is expected to facilitate further exploration by trying to aid in driving the domain forward.

5. RELATED WORK

The author will break down the work towards general TS forecasting and its application in BTC forecasting.

Table 1: Related Work (*Self-Composed*)

Citation	Summary	Contributions	Limitations
TS forecasting (general)			
Hochreiter and Schmidhuber, 1997	LSTM. An algorithm that learns to bridge minimal time lags by enforcing constant error flows. It learns much faster, creates more successful runs, and can solve complex tasks that have not been solved before.	Improved performance for short-sequence predictions. Overcame error back-flow problems present in conventional BPTT, where they tended to blow up or vanish.	Prediction capacity limits long sequence performance, where the MSE and RMSE rise unacceptably. Therefore, there are better solutions for predictions of the distant future.
Box et al., 2015	ARIMA. A statistical analysis model for understanding the dataset or predicting future trends. This model depends on past values to predict the future and uses lagged moving averages to smoothen the data.	Improved performance for TS forecasting data that correlate with values ahead of time.	Does not handle well with nonlinear data and long-term forecasting. Furthermore, it performs best on univariate analysis.
Oreshkin et al., 2020	N-BEATS. An architecture that solves the univariate time series point forecasting problem. It carries some benefits,	Outperformed the M4 competition winner of the previous year and improved the statistical benchmark forecast.	Tailored specifically for univariate TS analysis, therefore, would not perform well on multivariate analysis.

	some of which are being understandable, easily applicable to multiple other fields, and being fast to train.		
Existing algorithms all exhibit static behaviour			
BTC forecasting			
Roy et al., 2018	Applied statistical analysis to predict the price of BTC using data from 2013 to 2017. Applied the ARIMA model and obtained an overall accuracy of 90% for deciding weighted cost volatility.	Improved insights obtained and added context to future predictions based on past values, scoring an overall lower RMSE than other ML solutions.	Trained on data only between 2013 and 2017, capable of 10 consecutive day predictions, and does not consider other input parameters.
Rizwan et al., 2019	Compared the usage of LSTM and ARIMA models for the prediction of BTC; however, found that these models could be more efficient. Used GRU and eventually gained higher overall accuracy.	Built a multivariate model using other features (high, low, volume) and improved existing models built using RNN and LSTM by producing better accuracy and lower MSE, alongside taking much less time to train.	Trained on data only between 2014 and 2019 and does not consider external factors (Twitter tweets & volume).
Fleischer et al., 2022	Focused on volatility and understanding the behaviour of cryptocurrencies. Trained an LSTM model using	Beat performance of ARIMA on more extended runtime training.	Limited to univariate: does not consider other input parameters and is capable of forecasting only one day.

	BTC close price values to predict future prices.		

Critique on time series algorithms

A drawback of the above algorithms is that they are static and lack adaptability (Hasani et al., 2020). TS data is volatile, ever-changing, and unpredictable. They can get unexpected characteristic changes to their inputs. Therefore, a fixed statistical model or neural network can struggle, which could be a reason for the identification of Makridakis et al. (2018b). Furthermore, statistical-based algorithms require a lot of domain knowledge to tune the parameters and demonstrate linear behaviour, which could be better (Maiti, Vyklyuk and Vukovic, 2020). Although DL algorithms introduce non-linearity, they lack expressivity and are deemed a 'black-box'.

Critique on bitcoin forecasting solutions

The work evaluated in the above table had not considered exogenous factors that could have an impact. Therefore, a significant concern is that they cannot adapt well; in other words, they are not robust. Factors that could influence the price are as follows:

- Tweet sentiment & volume
- Google Trends

Cryptocurrency forecasting is as reliable as palm reading since it is an open system. Hence, uncontrollable factors such as a country's law can affect the price. Despite this, researchers can consider certain factors, such as the ones mentioned above. Abraham et al. (2018) identified that the above factors correlate with the price of BTC; therefore, it is vital to consider them when building such systems in the future.

6. RESEARCH GAP

The literature defines only a single paper for the proposed algorithmic solution (Hasani et al., 2020). Where every other work is not directly related to the algorithm but is to the family of neural ODEs (CT-RNN (Funahashi and Nakamura 1993; Rubanova, Chen and Duvenaud, 2019) and

CT-GRU (Mozer, Kazakov and Lindsey, 2017)) and the secondary problem domain of cryptocurrencies and TS. Furthermore, no algorithmic solution exists for the proposed LTC architecture for model implementation.

Gap in existing forecasting algorithms

It is also worth noting that, because of this, existing forecasting solutions are all implemented using traditional deep neural net approaches (ex: LSTM (Hochreiter and Schmidhuber, 1997)) that are static and hence have the limitation of not being able to learn and adapt during inference (Hasani et al., 2020), which results in the model's accuracy degrading overtime – a '*data drift*' (Poulopoulos, 2021).

Gap in chosen algorithm

The proposed LTC architecture uses a sequence of linear ODEs, which are now considered obsolete and lack instantaneous adaptability. Recent advancements in this field suggest the usage of SDEs instead, as they are more flexible (Duvenaud, 2021). An additional issue is that ODEs model 'deterministic dynamics' – uncertainty, or any unobserved interactions cannot be modelled, which is inevitable in TS data.

7. RESEARCH CONTRIBUTION

In a nutshell, the author desires to answer the following hypothesis:

- **H**01 – Would a novel architecture built by a novel algorithm utilizing an LTC architecture with SDEs instead of ODEs be an advancement for TS forecasting?

7.1 Research domain contribution

An implementation of the LTC algorithm with the abovementioned change will be developed, following the proposed architecture, to facilitate the model creation. Additionally, the algorithm built will be generalized without being problem-specific so that it can be applied elsewhere to evaluate its performance and identify whether the LTC would also be an advancement to other domains.

Moreover, hypothesis **H**01 will be evaluated by identifying whether the developed architecture provides strong robustness and accuracy and outperforms currently existing TS forecasting approaches.

7.2 Problem domain contribution

Having understood the issues in the current literature, a solution capable of solving the mentioned issues could advance for future research. Adapting to unforeseen changes and being highly expressive could be the stepping stone within the TS forecasting community.

Moreover, based on the above critique, creating a more robust forecasting solution considering the mentioned factors (Twitter, Google Trends) could mean that the highly volatile market of cryptocurrencies could be predicted much more efficiently and be the way forward for investors.

8. RESEARCH CHALLENGE

Existing architectures scale up, and the LTC scales down - with more expressive nodes (Hasani et al., 2020). Having adapted to the "deeper is usually better" mindset of deep neural nets, a challenge opens up in identifying the requirement of scaling down and what a neural ODE aims to solve.

LTCs are a new approach with only a single research paper regarding its proposed solution. Currently, it is only in the experimental stage and utilizes a novel formulation compared to other existing neural ODEs (Hasani et al., 2020). The broader domain of neural ODEs (Chen et al., 2019) is also relatively new; hence the scarcity of references could create more challenges for further research or implementation of systems.

SDEs are an advanced topic in mathematics, and modelling them as neural SDEs have had a couple of research conducted; however, they were primarily for specific purposes. Therefore, no generic papers exist for neural SDEs, unlike neural ODEs, which would make modelling difficult.

Currently, existing TS forecasting systems are built using statistical ensemble methods (Makridakis et al., 2018b) or traditional neural net architectures (Hasani et al., 2021), which creates a new challenge where there is no reference implementation.

The chosen domain of application is an open system. Open system forecasting is usually poor and generally difficult to beat the naïve forecast (A naive forecast is not necessarily bad, 2014) since it can depend on any external factor. Therefore, there is the possibility of discouragement from continuing the research if the results are not as expected.

9. RESEARCH QUESTIONS

RQ1: What are the recent advancements in TS algorithms that can be considered when building the algorithm?

RQ2: How can the algorithm be used to implement a TS forecasting system, and how will the challenges faced be overcome?

RQ3: What contributions can be made to the chosen domain?

10. RESEARCH AIM

The aim of this research is to design, develop & evaluate a novel algorithm inspired by the LTC so that it can build intelligent systems by developing a novel architecture for TS forecasting, which could be the stepping stone to be further expanded to other domains as well.

Specifically, this research project will produce a TS forecasting system utilizing the said algorithm, focused on the forecasting of BTC.

The researched knowledge will be presented, and hypothesis **H**01 will be evaluated.

11. RESEARCH OBJECTIVES

Research objectives are milestones that the author must achieve for the research to succeed.

Table 2: Research Objectives (*Self-Composed*)

Objective	Description	Learning Outcomes	Research Questions

Literature Review	Collate relevant information by reading, understanding, and evaluating previous work. • **R**O1: Conduct preliminary studies and investigations on existing TS forecasting systems. • **R**O2: Analyze the requirement for specialized TS algorithms. • **R**O3: Conduct a preliminary study on neural ODEs, LTCs & SDEs. • **R**O4: Obtain deep insights into the architecture behind the LTC.	LO1, LO2, LO4, LO5	**R**Q1
Requirement Elicitation	Collect and analyze project requirements using appropriate tools and techniques. • **R**O1: Gather the requirements and architectures of LTCs, and SDEs. • **R**O2: Collate the most up-to-date details of BTC. • **R**O3: Design a schedule, associated deliverables, and the Gantt chart.	LO1, LO2, LO3	**R**Q1
Design	Design the architecture and a corresponding system capable of effectively solving the identified problems. • **R**O1: Design an efficient and novel LTC implementation. • **R**O2: Design an automated flow to update the built network with the latest data. • **R**O3: Design an ML pipeline for easy deployments.	LO1	RQ2

Implementation	Implement a system that is capable of addressing the research gaps. • **R**O1: Implement a novel LTC architecture. • **R**O2: Integrate the algorithm developed into a TS forecasting application. • **R**O3: Integrate the intelligent system into the prototype to display forecasts. • **R**O4: Consider any legal, social, ethical & professional issues upon implementation.	LO1, LO5, LO6, LO7	**R**Q2
Evaluation	Effectively test the algorithm implemented, the system, and the respective data science model using recommended techniques. • **R**O1: Create a test plan & test cases and perform unit, performance, and integration testing. • **R**O2: Evaluate the developed algorithm and the respective model against the mentioned evaluation metrics.	LO5	**R**Q2, **R**Q3
Documentation	Document the progression of the research project and inform about any challenges faced. • **R**O1: Document any legal, social, ethical & professional issues faced and how they were solved. • **R**O2: Create a coherent report of new skills obtained throughout the project plan and critical evaluations.	LO6, LO8	-

12. PROJECT SCOPE

Concerning the time granted for this research project, the scope is as follows.

12.1 In-scope

- Implementing a novel LTC architecture capable of being used as currently existing solutions and the corresponding creation of a system.
- Periodical updates of the model with the latest available data.
- Evaluate and compare the implemented system against existing solutions to validate or invalidate hypothesis **H**01.
- Ability to display a range of predictions for the chosen horizon.
- By combining them with the BTC historical data, consider Twitter sentiment, volume, and the ‘block reward size’ as external factors.

12.2. Out-scope

- Application of the algorithm implemented in other domains to justify whether it could be an advancement in those domains.
- Forecast multiple different cryptocurrencies.
- Use of live, on-demand data instead of daily data & incremental learning.
- Consider other external factors such as legislation and laws, fiat currencies, and country advertisements for handling digital currency.

12.3 Desirables

- Benchmark implementation against the M4 competition to further justify the future of TS forecasting algorithms.
- Evaluate other neural ODEs (CT-RNN, CT-GRU, Latent ODE) and SDEs (Latent SDE).
- Explainable AI for neural SDEs and neural ODEs.

12.4 Prototype diagram

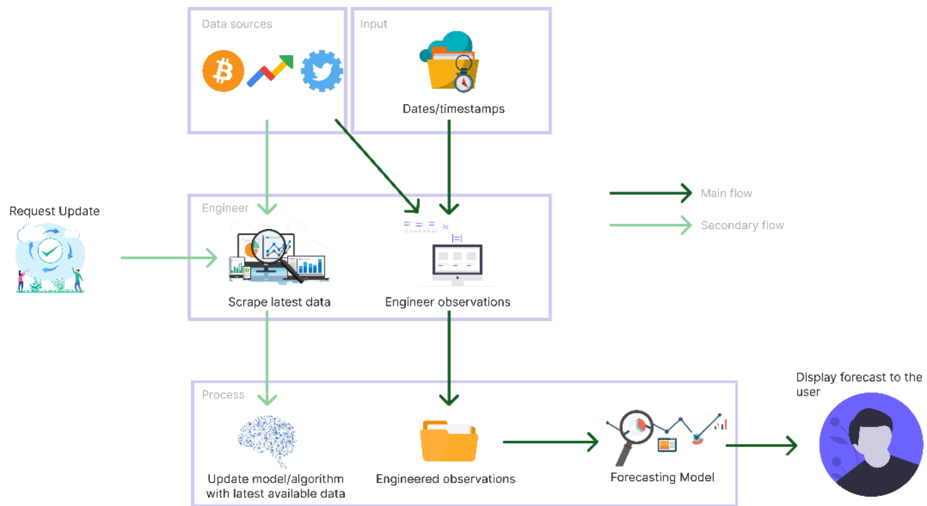

Figure 1: Prototype Feature Diagram (*Self-Composed*)

13. METHODOLOGY

13.1 Research methodology

Methodologies suitable for the research project have been evaluated and chosen from the predefined Saunders Research Onion Model (Saunders, Lewis and Thornhill, 2007, p102).

Table 3: Research Methodology

Philosophy	The **positivism** philosophy was chosen: although the data collected will be a collection of nominal and ordinal data, the data collected will be encoded into numbers. Additionally, as the outcome of this research, it is expected to validate/invalidate hypothesis **H**01 using necessary evaluation comparisons.
Approach	The **deductive** approach was chosen over the inductive since the final analysis and evaluation will be quantitative, which aims to deduce hypothesis **H**01.
Strategy	**Archival research** and **action research** were chosen as the primary data collection strategy. Since the research topic is modern, the principal source of data collection would be research documents. Action research will also be included since the research process will likely be an iterative approach of

	diagnosis, planning, action & evaluation. To assist the supplementary research taking place, **survey** was chosen to obtain insights from end users.
Choice	**Multi-method** will suit the proposed research project most since qualitative analysis would complement to the primary quantitative approach. However, it will not be used as a combination.
Time Horizon	The **cross-sectional** time horizon was chosen over the longitudinal time horizon. Even though the latest available data will have to be obtained often to update the model, there will be no interlinking between the times when the data is gathered as they will be independent.
Techniques and procedures	As a form of **data collection & analysis**, as many sources as possible will be used since there are finite resources. The primary mediums will be statistics, reports, journals, articles, and observations.

13.2 Development methodology

13.2.1 Life cycle model

Prototyping was chosen as the life cycle development methodology since heavy iterative building development and evaluation are required.

13.2.2 Requirement elicitation methodology

The author will utilize **surveys**, **interviews**, and the knowledge obtained from available **literature** to gather requirements to implement the associated software produced.

13.2.3 Design methodology

Structured System Analysis & Design Method (SSADM) was chosen as the design methodology since it is easier to understand and maintain. These are essential factors given that the project will be developed over a considerably long period. Additionally, the selected software requirements (Python & React) do not promote Object Oriented Programming (OOP) in nature, but rather, functions and modules. It is also worth mentioning that it will have little discernible benefit in the case of a data science project.

13.2.4 Software development methodology

Structured programming will accompany the chosen design methodology to facilitate development using modules and functions.

13.2.5 Evaluation methodology

A specialized version of the K-fold cross-validation: cross-validation on a rolling basis (Shrivastava, 2020), will be used as a means of evaluation. Validation is required to ensure the model is robust and does not overfit.

The evaluation metrics: MSE, RMSE, (s)MAPE, and MASE (Hyndman et al., 2021) will evaluate the system's performance to produce adequate comparisons against existing solutions and validate or invalidate hypothesis **H**01.

Benchmarking

As no previous system utilizing an LTC architecture exists, the author will not be able to conduct a comparative benchmarking analysis on the proposed system. However, conducting baselining to capture performance for future research benchmarking is feasible and will be carried out.

13.3 Solution methodology

As mentioned, a BTC forecasting prototype will be built to create justification.

A summarized workflow that will be followed when creating the model is depicted below.

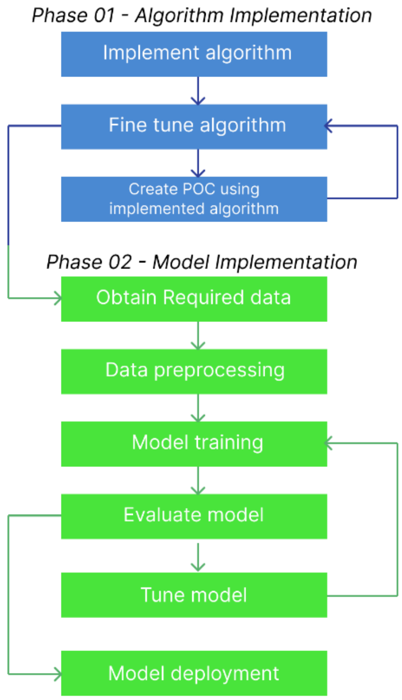

Figure 2: Model creation workflow (*Self-Composed*)

Phase 01

13.3.1 Implement the algorithm

The first and most crucial step is to implement the algorithm. This step is critical since it will give the author an idea of whether the development is feasible, whether pivoting is necessary, or even if the project must change entirely. The paper by Hasani et al. (2020) will be used as a guide alongside the latent SDE talk presented by Duvenaud (2021).

13.3.2 Fine-tune the algorithm

Once satisfactory progress has been made, the code must be cleaned and fine-tuned to be scalable and generalizable.

13.3.3 Create a POC

An example POC must be implemented to validate whether the supplemental forecasting application is feasible. This step is also essential since it will give the author an idea of how the software will be built.

Creating the POC and fine-tuning will be an iterative process since minor tweaks will be done while implementing.

Phase 02

13.3.4 Obtain the required data

As identified in the literature: existing systems had been trained on data that are outdated now. To address this limitation, the data used in this project will be scraped using APIs, which will be the most up-to-date.

Furthermore, to keep the model as updated as possible, the model will be retrained periodically with the existing new data.

13.3.5 Data preprocessing

Once the data has been fetched, it must be cleaned. The APIs return redundant & unneeded columns (ex: repeated features with different names) that must first be removed. The author must apply NLP techniques to extract sentiment scores and related information extracted. Once cleaned, they can be combined, creating a single set.

Data processing for TS forecasting applications is not the same as classification or regression problems since the data is temporal – therefore, the order must be given prominence.

Creating the train and test sets is unlike other problems, as random splits will not work. The data will be split sequentially, at a point in time such that the observations before it is the train data and after it the test data, a 'pseudo future.' Therefore, there is no 'leakage' between the two sets (Hyndman et al., 2021): past data must forecast the future.

Finally, the data must be 'windowed' to convert into a supervised learning problem and split into features and labels (BI4ALL, 2021), which is required since past windows will predict the future.

13.3.6 Model training

Once the data windows are ready, the model can be created. Here, the developed cell will be used within an RNN layer to provide a fair comparison against other existing cells like the LSTM.

13.3.7 Evaluation

Once the model has been trained, sufficient evaluation must be conducted to shed light on the model's performance. The model will be evaluated using the metrics discussed in the evaluation methodology.

13.3.8 Tuning

If the performance obtained is subpar, the model hyperparameters must be tuned (ex: number of epochs, batch size, learning rate, optimizer, activation function, number of units & layers). Tuning mentioned hyperparameters could cause a significant change in performance – even worsen the performance. However, this is an important step that must be carried out, as it could drastically improve performance.

Training, evaluation & tuning will be an iterative process, as it is unlikely to obtain the best-performing model in the first experiment. It will also be unexpectedly long since there exists no reference implementation. Therefore, common hyperparameter values are not documented.

13.3.9 Deployment

The final step in the implementation is to deploy the forecasting model so that it can be accessed from anywhere, in this case, especially the client application.

In addition, a deployment pipeline must be built to facilitate future automatic deployments whenever the model is updated periodically.

13.4 Project management methodology

The author will follow a combination of PRINCE2 and Agile. The project will require many iterations and improvements since the implementation is novel and no reference exists. Alongside multiple iterations, it is best implemented by being divided into multiple chunks and focusing on each chunk at a time with a plan-based approach.

13.4.1 Schedule

Gantt chart

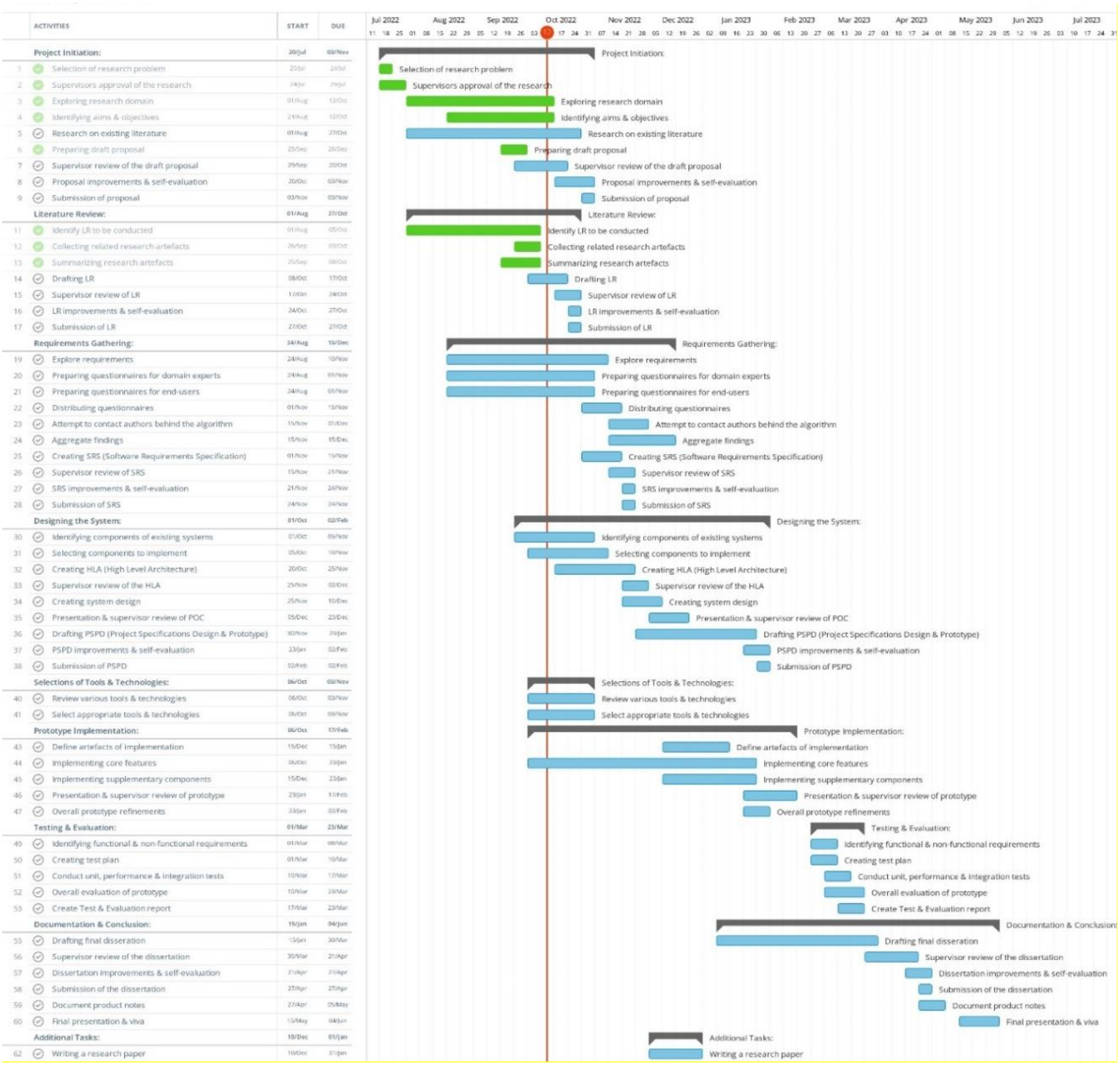

Figure 3: Gantt Chart (*Self-Composed*)

A clearer version can be found [Here]

Deliverables

Table 4: Deliverables & Dates (*Self-Composed*)

Deliverable	**Date**
Literature Review Critical analysis of related work & solutions.	27th October 2022
Project Proposal & Ethics Forms	3rd November 2022

The initial proposal of the research to be conducted.	
Software Requirement Specification Defines the requirements that must be met to prototype and collect data.	24th November 2022
Proof Of Concept & Implementation Presentation An initial implementation of the proposed system.	23rd December 2022
Project Specifications Design & Prototype A prototype of the system with the core features and an accompanying document specifying the design followed & an overview of the implemented algorithm.	2nd February 2023
Test & Evaluation Report Documentation of test findings and evaluations conducted on the prototype.	23rd March 2023
Draft Project Report A draft submission of the final thesis to get evaluations.	30th March 2023
Final Thesis Final submission of the thesis with complete documentation of the project's journey.	27th April 2023

13.4.2 Resource requirements

Software requirements

- **Operating System (Windows /** Linux **/** macOS**) -** Windows will be the default since it provides easy access to the required development environments and tools. Besides, the author does not have access to other operating systems.

- **Python /** R - to create the network & the respective model. Python, since it has a much simpler learning curve, provides easy integration with other mentioned software and is optimized for large-scale ML software. Meanwhile, R is more suitable for statistical data analysis (Python vs. R: What's the Difference?, 2021)
- **TensorFlow /** Torch – provides libraries that facilitate DL in Python & R. TensorFlow, due to its large developer community and seamless integration with Keras for higher-level API development. Additionally, multiple visualizations will be required, which is made simple by TensorBoard (Dubovikov, 2018).
- **Flask /** Node / Golang – for seamless communication between the client and the model. Flask will be the primary choice since the ML component will use Python, it is incredibly lightweight, and there is only a requirement for a minimal REST API. Node and Golang are secondary options if there are requirements to add additional features, such as authentication, that are not directly relevant to the research.
- **React** / Vue / Svelte – required to develop the client-side. A fast performant library is required to prevent lags and other performance issues. React will be the option because of the author's familiarity and large community. Svelte and Vue will be the second options if React is not performant enough. Angular is not considered since it is less performant and more suitable for larger-scale applications, which is not required. (React vs Vue vs Angular vs Svelte, 2020).
- **VSCode /** PyCharm – environment to facilitate application development. VSCode is the primary choice since it provides a general-purpose yet lightweight development experience with multiple plugins making it more developer-friendly. PyCharm will be a secondary option if there are issues with the Python environment or a need for a dedicated python development environment.
- **Jupyter Notebook /** Google Colab – development environment for building the forecasting model. Jupyter will be the primary choice as it has less risk: it runs locally. Therefore, in case of power failures, training would not be interrupted. Colab will be the backup choice if there is a requirement for a GPU to train the model.
- **Zotero** / Mendeley – manage references and research artefacts. Zotero is chosen due to the author's preference and is easy to use.

- **MS Office | Overleaf** – tools to create primary research reports. A combination of MS Office and Overleaf will facilitate the development of the project submission documents and the creation of professional research papers/surveys/reviews, if necessary.
- **Google Drive** / OneDrive – to backup research artefacts. Google Drive will be the primary option due to the unlimited storage availability provided by the university.
- **Figma | Canva | Draw.io –** tools to create figures and diagrams. Combining all three will streamline the design process, as each has its advantages. Figma - to design & prototype; Draw.io – to draw flow charts and other associated diagrams; Canva - to prepare attractive presentation slides.
- **GitHub** / Bitbucket **/** Gitlab – track, version & manage development code & research documents. GitHub will be the choice due to the author's familiarity, integrations with the development environments, and email notifications that could be significant.

Hardware requirements

- **Core i5 Processor (8th gen) /** Ryzen 5 **/** M1 – for long-running intensive workloads and managing multiple development environments. The author has access to an intel processor; therefore, it will be the CPU choice. However, the other options would suffice just as much.
- **8GB Ram or above** – to manage model training, multiple development environments & multitasking. Moreover, it is required to load large datasets and multitask.
- **Disk space of approx. 20GB** – to store application code & data.

If the available hardware does not meet the required criteria, a cloud-based development environment can be used (ex: GitHub codespaces, Google Colab, Zotero web, Google Drive).

Data requirements

- **BTC price observations & block reward size** – from a financial website (ex: investing.com, cmcmarkets.com, finance.yahoo.com).
- **BTC tweets** – from the Twitter API or a respective website that provides the required data (ex: bitinfocharts.com).
- **Google trends –** from a Python API (PyTrends) that supplies Google Trends data.

Skill requirements

- Creation of TS forecasting systems.
- Knowledge of ODEs, SDEs & respective solvers.
- Implementation of a raw neural ODE and SDE.
- Ability to create optimized & scalable DL models.
- Ability to develop optimized client-side charts & user interfaces that dynamically update.
- Research & academic writing skills.

13.4.3 Risk management

The following table identifies possible risks that the author could face and how they could mitigate them.

Table 5: Risk Management Plan (*Self-Composed*)

Risk Item	Severity	Frequency	Mitigation Plan
Lack of required knowledge	5	5	Get insights from domain experts and, if necessary, the author of the proposed algorithm.
Corrupted documentation	4	4	Store all necessary documentation on the cloud as well as external storage.
Lose access to development code	5	2	Backup code on source control and cloud storage.
Inability to deliver all expected deliverables	4	2	Follow a list of priorities and deliver accordingly.
Invalid hypothesis **H**01	3	2	Continue researching since the final output is a research contribution regardless.

REFERENCES

S. Nakamoto, (2020). *Bitcoin: A peer-to-peer electronic cash system*. Available from https://bitcoin.org/bitcoin.pdf [Accessed 25 September 2022].

Rahouti, M., Xiong, K. and Ghani, N. (2018). Bitcoin Concepts, Threats, and Machine-Learning Security Solutions. *IEEE Access*, 6, 67189–67205. Available from https://doi.org/10.1109/ACCESS.2018.2874539 [Accessed 25 September 2022].

Kervanci, I. sibel and Akay, F. (2020). Review on Bitcoin Price Prediction Using Machine Learning and Statistical Methods. *Sakarya University Journal of Computer and Information Sciences*. Available from https://doi.org/10.35377/saucis.03.03.774276 [Accessed 25 September 2022].

Makridakis, S., Spiliotis, E. and Assimakopoulos, V. (2018a). Statistical and Machine Learning forecasting methods: Concerns and ways forward. *PLOS ONE*, 13 (3), e0194889. Available from https://doi.org/10.1371/journal.pone.0194889 [Accessed 25 September 2022].

Makridakis, S., Spiliotis, E. and Assimakopoulos, V. (2018b). The M4 Competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, 34 (4), 802–808. Available from https://doi.org/10.1016/j.ijforecast.2018.06.001 [Accessed 25 September 2022].

Smyl, S. (2020). A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, 36 (1), 75–85. Available from https://doi.org/10.1016/j.ijforecast.2019.03.017 [Accessed 25 September 2022].

Hasani, R. et al. (2020). Liquid Time-constant Networks. Available from https://doi.org/10.48550/arXiv.2006.04439 [Accessed 25 September 2022].

Chen, R.T.Q. et al. (2019). Neural Ordinary Differential Equations. Available from https://doi.org/10.48550/arXiv.1806.07366 [Accessed 25 September 2022].

Poulopoulos, D. (2021). Is “Liquid” ML the answer to autonomous driving? *Medium*. Available from https://towardsdatascience.com/is-liquid-ml-the-answer-to-autonomous-driving-bf2e899a9065 [Accessed 25 September 2022].

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9 (8), 1735–1780. Available from https://doi.org/10.1162/neco.1997.9.8.1735 [Accessed 25 September 2022].

Oreshkin, B.N. et al. (2020). N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. Available from http://arxiv.org/abs/1905.10437 [Accessed 26 September 2022].

G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, Time series analysis: forecasting and control (John Wiley & Sons, 2015).

Roy, S., Nanjiba, S. and Chakrabarty, A. (2018). Bitcoin Price Forecasting Using Time Series Analysis. *2018 21st International Conference of Computer and Information Technology (ICCIT)*. December 2018. Dhaka, Bangladesh: IEEE, 1–5. Available from https://doi.org/10.1109/ICCITECHN.2018.8631923 [Accessed 25 September 2022].

Rizwan, M., Narejo, S. and Javed, M. (2019). Bitcoin price prediction using Deep Learning Algorithm. *2019 13th International Conference on Mathematics, Actuarial Science, Computer Science and Statistics (MACS)*. December 2019. Karachi, Pakistan: IEEE, 1–7. Available from https://doi.org/10.1109/MACS48846.2019.9024772 [Accessed 26 September 2022].

Fleischer, J.P. et al. (2022). Time Series Analysis of Cryptocurrency Prices Using Long Short-Term Memory. *Algorithms*, 15 (7), 230. Available from https://doi.org/10.3390/a15070230 [Accessed 26 September 2022].

Saunders, M.N.K., Lewis, P. and Thornhill, A. (2007). *Research methods for business students*, 4th ed. Harlow, England ; New York: Financial Times/Prentice Hall.

Shrivastava, S. (2020). Cross Validation in Time Series. *Medium*. Available from https://medium.com/@soumyachess1496/cross-validation-in-time-series-566ae4981ce4 [Accessed 12 October 2022].

BI4ALL. (2021). Supervised Machine Learning in Time Series Forecasting. *BI4ALL - Turning Data Into Insights*. Available from https://www.bi4all.pt/en/news/en-blog/supervised-machine-learning-in-time-series-forecasting/ [Accessed 12 October 2022].

Hyndman, R.J., & Athanasopoulos, G. (2021). *Forecasting: principles and practice*, 3rd edition, OTexts: Melbourne, Australia. Available from https://otexts.com/fpp3/. [Accessed on 30 Sep. 2022].

Hasani, R. et al. (2021). Liquid Neural Networks. *YouTube*. Available from https://www.youtube.com/watch?v=IlliqYiRhMU&t=350s. [Accessed on 30 Sep. 2022].

Mozer, M.C., Kazakov, D. and Lindsey, R.V. (2017). Discrete Event, Continuous Time RNNs. Available from https://doi.org/10.48550/ARXIV.1710.04110 [Accessed 14 October 2022].

Funahashi, K. and Nakamura, Y. (1993). Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Networks*, 6 (6), 801–806. Available from https://doi.org/10.1016/S0893-6080(05)80125-X [Accessed 14 October 2022].

A naive forecast is not necessarily bad. (2014). *The Business Forecasting Deal*. Available from https://blogs.sas.com/content/forecasting/2014/04/30/a-naive-forecast-is-not-necessarily-bad/ [Accessed 15 October 2022].

Dubovikov, K. (2018). PyTorch vs TensorFlow — spotting the difference. *Medium*. Available from https://towardsdatascience.com/pytorch-vs-tensorflow-spotting-the-difference-25c75777377b [Accessed 18 October 2022].

React vs Vue vs Angular vs Svelte. (2020). *DEV Community* 👩‍💻👨‍💻. Available from https://dev.to/hb/react-vs-vue-vs-angular-vs-svelte-1fdm [Accessed 18 October 2022].

Python vs. R: What's the Difference? (2021). Available from https://www.ibm.com/cloud/blog/python-vs-r [Accessed 18 October 2022].

Maiti, M., Vyklyuk, Y. and Vuković, D. (2020). Cryptocurrencies chaotic co-movement forecasting with neural networks. *Internet Technology Letters*, 3 (3). Available from https://doi.org/10.1002/itl2.157 [Accessed 16 October 2022].

Abraham, J., Higdon, D., Nelson, J. and Ibarra, J. (2018). Cryptocurrency Price Prediction Using Tweet Volumes and Sentiment Analysis. *SMU Data Science Review:* Vol. 1: No. 3, Article 1. Available at: https://scholar.smu.edu/datasciencereview/vol1/iss3/1

Rubanova, Y., Chen, R.T.Q. and Duvenaud, D. (2019). Latent ODEs for Irregularly-Sampled Time Series. Available from https://doi.org/10.48550/ARXIV.1907.03907 [Accessed 18 October 2022].

Chaman L. Jain. Answers to your forecasting questions. *Journal of Business Forecasting*, 36, Spring 2017.

Duvenaud, D (2021). Directions in ML: Latent Stochastic Differential Equations: An Unexplored Model Class. *YouTube*. Available from https://www.youtube.com/watch?v=6iEjF08xgBg. [Accessed on 30 Sep. 2022].