Informatics Institute of Technology

In Collaboration With

The University of Westminster, UK



*The University of Westminster, Coat of Arms*

GenSum

A Generalized Text Summarization System using Optimized

Transformers

A dissertation by

Mr. Nazhim Kalam

W1761265 | 2019281

Supervised by

Mr. Torin Wirasingha

April 2023

Submitted in partial fulfilment of the requirements for the

BSc (Hons) Computer Science degree at the University of Westminster.

**DECLARATION**

I affirm that this dissertation, including its sub-components, is the product of my own original research endeavors. Furthermore, I confirm that I have not previously submitted or presented any of this content, in whole or in part, as part of any other degree or qualification program at any other university or institution. Any factual information obtained from credible external sources has been duly acknowledged through proper citation.

**Student Name**: Nazhim Kalam

**Registration Number**: w1761265 -- 2019281

**Signature: Date:** April 27, 2023

****

**ABSTRACT**

Abstractive text summarization systems have been integrated with various application in the world to perform text summarization and its nothing new to the field. However, with the prior research it found that in the domain of movies the need for performance improvement is required using latest approaches than the current traditional ML & DL methods, movie review summarization plays a major role in helping users to make better decisions by matching their interest with the reviews of the movie, this saves a lot of time and also improves businesses in their sales.

In 2017 researches from Google Brain introduced NLP Transformers, which is a latest approach to solve NLP problems and its increasingly been known and used nowadays over traditional ML & DL approaches like using basic LSTM, RNN approaches. The author explored ways in which to get an optimal solution using Transformer for abstractive text summarization and yet making a generalized solution which can be adapted with respect to any domain (be it hotels, movies, restaurants) and increase its performance as the system gets used over with time.

The author was able to experiment with few of the top tier transformer architectures to filter out the optimal model and integrated an automated hyperparameter searching mechanism which will find the best set of hyperparameters to train the model. **ROUGE1 of 80.8, ROUGE2 of 79.42, ROUGEL of 80.8, ROUGELSUM of 80.8** was the optimal evaluation metric result achieved from the BART model giving the best result.

**Keywords:** Natural Language Processing (NLP), Machine Learning (ML), Deep Learning (DL), Recall-Oriented Understudy for Gisting Evaluation (ROUGE), Inductive logic programming (ILP)

**Subject Descriptors:**

* Computing methodologies 🡪 Artificial intelligence 🡪 Natural language processing 🡪 Natural language generation
* Theory of computation 🡪 Theory and algorithms for application domains 🡪 Machine learning theory 🡪 Semi-supervised learning.
* Information systems 🡪 Information systems applications 🡪 Management and querying of encrypted data.
* Security and privacy 🡪 Database and storage security 🡪 Data mining.

# **PUBLICATIONS**

1. **A Review on Creating a Performance Adaptive Generalized Abstractive Text Summarization Using Optimized Transformers**

Conference: SmartNets 2023 - BDA-ML

Status: Submitted, In-Review

Venue: Asia, Colombo

Date: April 08, 2023

# **ACKNOWLEDGEMENT**

I am grateful for the supportive individuals in my life who have empowered me to exceed my previous limitations and achieve my current level of success, including the completion-of this final year research project.

I would like to express my deepest gratitude to Mr. Guhanathan Poravi, my esteemed module leader, for his consistent review and invaluable guidance that has tremendously improved my reports. His feedback has instilled in me the confidence to conduct research at any level. I am also grateful to my supervisor, Mr. Torin Wirasingha, for his contribution in refining my project idea and expanding its scope. I extend my appreciation to the evaluators of this project, who took the time to thoroughly review my work and provide valuable advice for its enhancement. Furthermore, I would like to thank all the interviewees and survey respondents, including industry and academia experts, as well as my senior and junior peers at IIT, for their invaluable input in the requirement gathering process.

I express my heartfelt appreciation to the esteemed lecturers at the Informatics Institute of Technology, as well as the supportive management, for the invaluable opportunities, unwavering support, and invaluable guidance they have provided me throughout my undergraduate degree. I would also like to acknowledge my fellow friends from university, whose continued support and positive influence have played a pivotal role in my growth as a developer/researcher. Their encouragement and camaraderie have fueled my rapid progress over the span of four years. Lastly, I extend my deepest gratitude to my parents for their unwavering support and encouragement in pursuing my dreams, and to my beloved family, friends, and significant other, whose unwavering encouragement has been a beacon of strength during challenging times.

**CONTENTS**

[**PUBLICATIONS** Iii](#_Toc132325735)

[**ACKNOWLEDGEMENT** Iv](#_Toc132325736)

[**LIST OF TABLES** Xii](#_Toc132325737)

[**LIST OF FIGURES** Xiv](#_Toc132325738)

[**LIST OF ABBREVIATIONS** Xvi](#_Toc132325739)

[**CHAPTER 01. INTRODUCTION** 1](#_Toc132325740)

[**1.1 Chapter Overview** 1](#_Toc132325741)

[**1.2 Problem Domain** 1](#_Toc132325742)

[**1.2.1 Time Series Forecasting** 1](#_Toc132325743)

[**1.2.2 Liquid Time-Constant (LTC) Networks** 1](#_Toc132325744)

[**1.2.3 Cryptocurrencies** 2](#_Toc132325745)

[**1.3 Problem Definition** 3](#_Toc132325746)

[**1.3.1 Problem Statement** 3](#_Toc132325747)

[**1.4 Research Motivation** 3](#_Toc132325748)

[**1.5 Research Gap** 3](#_Toc132325749)

[**1.6 Contribution To The Body Of Knowledge** 4](#_Toc132325750)

[**1.6.1 Research Domain Contribution** 5](#_Toc132325751)

[**1.6.1 Problem Domain Contribution** 5](#_Toc132325752)

[**1.7 Research Challenge** 5](#_Toc132325753)

[**1.8 Research Questions** 6](#_Toc132325754)

[**1.9 Research Aim** 6](#_Toc132325755)

[**1.10 Research Objectives** 7](#_Toc132325756)

[**1.11 Chapter Summary** 9](#_Toc132325757)

[**CHAPTER 02. LITERATURE REVIEW** 10](#_Toc132325758)

[**2.1 Chapter Overview** 10](#_Toc132325759)

[**2.2 Concept Map** 10](#_Toc132325760)

[**2.3 Problem Domain** 10](#_Toc132325761)

[**2.3.1 Time Series Forecasting** 10](#_Toc132325762)

[**2.3.2 Cryptocurrencies (Bitcoin)** 11](#_Toc132325763)

[**2.3.2.1 Opportunities Of Cryptocurrencies** 11](#_Toc132325764)

[**2.3.2.2 Challenges Of Cryptocurrencies** 12](#_Toc132325765)

[**2.3.2.3 Why Have Cryptocurrencies Taken The World By Storm?** 12](#_Toc132325766)

[**2.3.2.4 Cryptocurrency Exchanges** 13](#_Toc132325767)

[**2.4 Existing Work** 13](#_Toc132325768)

[**2.4.1 Time Series Forecasting** 13](#_Toc132325769)

[**2.4.2 Bitcoin Forecasting** 13](#_Toc132325770)

[**2.4.2.1 Factors That Affect The Rate Of Bitcoin (BTC)** 14](#_Toc132325771)

[**2.4.3 Open Market Forecasting** 14](#_Toc132325772)

[**2.4.4 Bitcoin Twitter Analysis** 15](#_Toc132325773)

[**2.5 Technological Review** 16](#_Toc132325774)

[**2.5.1 Statistical-Based Forecasting Techniques** 17](#_Toc132325775)

[**2.5.1.1 Autoregressive Integrated Moving Average (ARIMA)** 17](#_Toc132325776)

[**2.5.1.2 Seasonal Autoregressive Integrated Moving Average (SARIMA)** 17](#_Toc132325777)

[**2.5.1.3 Generalized Autoregressive Conditional Heteroskedasticity (GARCH)** 17](#_Toc132325778)

[**2.5.1.4 Prophet** 17](#_Toc132325779)

[**2.5.2 Deep Learning-Based Forecasting Techniques** 18](#_Toc132325780)

[**2.5.2.1 Long Short-Term Memory (LSTM)** 18](#_Toc132325781)

[**2.5.2.2 Gated Recurrent Unit (GRU)** 18](#_Toc132325782)

[**2.5.2.3 Neural Basis Expansion Analysis For Interpretable Time Series (N-BEATS)** 19](#_Toc132325783)

[**2.5.2.4 Temporal Fusion Transformer (TFT)** 19](#_Toc132325784)

[**2.5.3 Concerns About Existing Used Techniques** 19](#_Toc132325785)

[**2.5.3.1 Issues In Statistical Models** 19](#_Toc132325786)

[**2.5.3.2 Issues In Deep Learning Models** 20](#_Toc132325787)

[**2.5.4 Neural Ordinary Differential Equations (Odes)** 20](#_Toc132325788)

[**2.5.5 Approach Proposed By A Liquid Time-Constant (LTC)** 21](#_Toc132325789)

[**2.5.6 Neural Stochastic Differential Equations (Sdes)** 22](#_Toc132325790)

[**2.5.7 Traditional Required Techniques** 23](#_Toc132325791)

[**2.5.7.1 Data Preprocessing** 23](#_Toc132325792)

[**2.5.7.2 Hyperparameter Tuning** 24](#_Toc132325793)

[**2.5.7.3 Validation** 24](#_Toc132325794)

[**2.5.8 NLP Techniques That Can Assist** 25](#_Toc132325795)

[**2.5.8.1 Sentiment Analysis** 25](#_Toc132325796)

[**2.5.8.2 Named Entity Recognition (NER)** 25](#_Toc132325797)

[**2.5.9 Proposed Architecture** 26](#_Toc132325798)

[**2.6 Evaluation** 26](#_Toc132325799)

[**2.6.1 Evaluation Approaches** 26](#_Toc132325800)

[**2.6.2 Benchmarking** 28](#_Toc132325801)

[**2.6.2.1 System Benchmarking** 28](#_Toc132325802)

[**2.6.2.2 Algorithm Benchmarking** 28](#_Toc132325803)

[**2.6.3 Research Justification** 28](#_Toc132325804)

[**2.7 Chapter Summary** 28](#_Toc132325805)

[**CHAPTER 03. METHODOLOGY** 29](#_Toc132325806)

[**3.1 Chapter Overview** 29](#_Toc132325807)

[**3.2 Research Methodology** 29](#_Toc132325808)

[**3.3 Development Methodology** 30](#_Toc132325809)

[**3.3.1 Life Cycle Model** 30](#_Toc132325810)

[**3.3.2 Requirement Elicitation Methodology** 30](#_Toc132325811)

[**3.3.3 Design Methodology** 30](#_Toc132325812)

[**3.3.4 Software Development Methodology** 30](#_Toc132325813)

[**3.4 Project Management Methodology** 30](#_Toc132325814)

[**3.4.1 Schedule** 31](#_Toc132325815)

[**3.4.1.1 Gantt Chart** 31](#_Toc132325816)

[**3.4.1.2 Deliverables** 32](#_Toc132325817)

[**3.5 Resources** 32](#_Toc132325818)

[**3.5.1 Software Requirements** 32](#_Toc132325819)

[**3.5.2 Hardware Requirements** 34](#_Toc132325820)

[**3.5.3 Technical Skills** 34](#_Toc132325821)

[**3.5.4 Data Requirements** 34](#_Toc132325822)

[**3.6 Risks & Mitigation** 35](#_Toc132325823)

[**3.7 Chapter Summary** 35](#_Toc132325824)

[**CHAPTER 04. SOFTWARE REQUIREMENTS SPECIFICATION** 36](#_Toc132325825)

[**4.1 Chapter Overview** 36](#_Toc132325826)

[**4.2 Rich Picture** 36](#_Toc132325827)

[**4.3 Stakeholder Analysis** 37](#_Toc132325828)

[**4.3.1 Stakeholder Onion Model** 37](#_Toc132325829)

[**4.3.2 Stakeholder Viewpoints** 38](#_Toc132325830)

[**4.4 Selection Of Requirement Elicitation Methodologies** 39](#_Toc132325831)

[**4.5 Discussion Of Findings** 40](#_Toc132325832)

[**4.5.1 Literature Review** 40](#_Toc132325833)

[**4.5.2 Observations** 40](#_Toc132325834)

[**4.5.3 Interviews** 41](#_Toc132325835)

[**4.5.4 Survey** 42](#_Toc132325836)

[**4.5.5 Prototyping** 47](#_Toc132325837)

[**4.5.6 Summary Of Findings** 49](#_Toc132325838)

[**4.6 Context Diagram** 50](#_Toc132325839)

[**4.7 Use Case Diagram** 50](#_Toc132325840)

[**4.8 Use Case Descriptions** 51](#_Toc132325841)

[**4.9 Requirements** 52](#_Toc132325842)

[**4.9.1 Functional Requirements** 52](#_Toc132325843)

[**4.9.2 Non-Functional Requirements** 53](#_Toc132325844)

[**4.10 Chapter Summary** 54](#_Toc132325845)

[**CHAPTER 05. SOCIAL, LEGAL, ETHICAL & PROFESSIONAL ISSUES** 55](#_Toc132325846)

[**5.1 Chapter Overview** 55](#_Toc132325847)

[**5.2 SLEP Issues And Mitigation** 55](#_Toc132325848)

[**5.3 Chapter Summary** 55](#_Toc132325849)

[**CHAPTER 06. DESIGN** 56](#_Toc132325850)

[**6.1 Chapter Overview** 56](#_Toc132325851)

[**6.2 Design Goals** 56](#_Toc132325852)

[**6.3 System Architecture Design** 57](#_Toc132325853)

[**6.3.1 Architecture Diagram** 57](#_Toc132325854)

[**6.3.2 Discussion Of Tiers Of The Architecture** 58](#_Toc132325855)

[**6.4 Detailed Design** 59](#_Toc132325856)

[**6.4.1 Choice Of Design Paradigm** 59](#_Toc132325857)

[**6.5 Design Diagrams** 59](#_Toc132325858)

[**6.5.1 Data Flow Diagrams** 59](#_Toc132325859)

[**6.5.1.1 Level 01 Data Flow Diagram** 59](#_Toc132325860)

[**6.5.1.2 Level 02 Data Flow Diagram** 60](#_Toc132325861)

[**6.5.2 Algorithm Design** 61](#_Toc132325862)

[**6.5.2.1 Liquid Time-Stochasticity (LTS) Algorithm** 61](#_Toc132325863)

[**6.5.2.2 Tweet Sentiment Weighting Algorithm** 63](#_Toc132325864)

[**6.5.3 LTS Algorithm Analysis** 63](#_Toc132325865)

[**6.5.4 UI Design** 64](#_Toc132325866)

[**6.5.5 System Process Activity Diagram** 64](#_Toc132325867)

[**6.6 Chapter Summary** 64](#_Toc132325868)

[**CHAPTER 07. IMPLEMENTATION** 65](#_Toc132325869)

[**7.1 Chapter Overview** 65](#_Toc132325870)

[**7.2 Technology Selection** 65](#_Toc132325871)

[**7.2.1 Technology Stack** 65](#_Toc132325872)

[**7.2.2 Selection Of Data** 66](#_Toc132325873)

[**7.2.3 Selection Of Programming Language** 66](#_Toc132325874)

[**7.2.4 Selection Of Development Framework** 67](#_Toc132325875)

[**7.2.4.1 Deep Learning (DL) Framework** 67](#_Toc132325876)

[**7.2.4.2 User Interface (UI) Framework** 67](#_Toc132325877)

[**7.2.4.3 Application Programming Interface (API) Web Framework** 67](#_Toc132325878)

[**7.2.5 Other Libraries & Tools** 67](#_Toc132325879)

[**7.2.6 Integrated Development Environment (IDE)** 68](#_Toc132325880)

[**7.2.7 Summary Of Chosen Tools & Technologies** 68](#_Toc132325881)

[**7.3 Implementation Of Core Functionalities** 69](#_Toc132325882)

[**7.3.1 Algorithm Implementation** 69](#_Toc132325883)

[**7.3.2 Data Fetchers** 71](#_Toc132325884)

[**7.3.3 Preprocessing** 72](#_Toc132325885)

[**7.4 User Interface** 72](#_Toc132325886)

[**7.5 Chapter Summary** 72](#_Toc132325887)

[**CHAPTER 08. TESTING** 73](#_Toc132325888)

[**8.1 Chapter Overview** 73](#_Toc132325889)

[**8.2 Testing Objectives & Goals** 73](#_Toc132325890)

[**8.3 Testing Criteria** 73](#_Toc132325891)

[**8.4 Model Testing & Evaluation** 74](#_Toc132325892)

[**8.4.1 Model Testing** 74](#_Toc132325893)

[**8.4.2 Model Evaluation** 75](#_Toc132325894)

[**8.5 Benchmarking** 75](#_Toc132325895)

[**8.6 Functional Testing** 76](#_Toc132325896)

[**8.7 Module & Integration Testing** 76](#_Toc132325897)

[**8.8 Non-Functional Testing** 76](#_Toc132325898)

[**8.9 Limitations Of The Testing Process** 77](#_Toc132325899)

[**8.10 Chapter Summary** 77](#_Toc132325900)

[**CHAPTER 09. EVALUATION** 78](#_Toc132325901)

[**9.1 Chapter Overview** 78](#_Toc132325902)

[**9.2 Evaluation Methodology & Approach** 78](#_Toc132325903)

[**9.3 Evaluation Criteria** 78](#_Toc132325904)

[**9.4 Self-Evaluation** 79](#_Toc132325905)

[**9.5 Selection Of Evaluators** 81](#_Toc132325906)

[**9.6 Evaluation Results & Expert Opinions** 81](#_Toc132325907)

[**9.7 Limitations Of Evaluation** 81](#_Toc132325908)

[**9.8 Evaluation Of Functional Requirements** 82](#_Toc132325909)

[**9.9 Evaluation Of Non-Functional Requirements** 82](#_Toc132325910)

[**9.10 Chapter Summary** 82](#_Toc132325911)

[**CHAPTER 10. CONCLUSION** 84](#_Toc132325912)

[**10.1 Chapter Overview** 84](#_Toc132325913)

[**10.2 Achievement Of Research Aim & Objectives** 84](#_Toc132325914)

[**10.2.1 Achievement Of The Research Aim** 84](#_Toc132325915)

[**10.2.2 Achievement Of Objectives** 84](#_Toc132325916)

[**10.3 Utilization Of Knowledge From The Degree** 84](#_Toc132325917)

[**10.4 Use Of Existing Skills** 85](#_Toc132325918)

[**10.5 Use Of New Skills** 85](#_Toc132325919)

[**10.6 Achievement Of Learning Outcomes** 86](#_Toc132325920)

[**10.7 Problems And Challenges Faced** 86](#_Toc132325921)

[**10.8 Deviations** 87](#_Toc132325922)

[**10.9 Limitations Of The Research** 87](#_Toc132325923)

[**10.10 Future Enhancements** 87](#_Toc132325924)

[**10.11 Achievement Of The Contribution To The Body Of Knowledge** 88](#_Toc132325925)

[**10.11.1 Research Domain Contributions** 88](#_Toc132325926)

[**10.11.2 Problem Domain Contributions** 88](#_Toc132325927)

[**10.11.3 Additional Contribution** 88](#_Toc132325928)

[**10.12 Implementation Code** 88](#_Toc132325929)

[**10.13 Concluding Remarks** 88](#_Toc132325930)

[**REFERENCES** I](#_Toc132325931)

[**APPENDIX A – INTRODUCTION** XII](#_Toc132325932)

[**A.1. Prototype Feature Diagram** XII](#_Toc132325933)

[**A.2. Project Scope** XII](#_Toc132325934)

[**APPENDIX B – LITERATURE REVIEW** XIV](#_Toc132325935)

[**B.1. Analysis Of Forecasting Algorithms** XIV](#_Toc132325936)

[**B.2. Studies Associated With These Algorithms** XVII](#_Toc132325937)

[**APPENDIX C – SRS** XIX](#_Toc132325938)

[**C.1. Requirement Elicitation Methodologies** XIX](#_Toc132325939)

[**C.2. Interview Analysis** XIX](#_Toc132325940)

[**C.3. Survey Analysis** XXI](#_Toc132325941)

[**C.4. Use Case Descriptions** XXIII](#_Toc132325942)

[**C.5. Functional Requirements** XXV](#_Toc132325943)

[**APPENDIX D – DESIGN** XXVI](#_Toc132325944)

[**D.1. LTS Algorithm Intuition** XXVI](#_Toc132325945)

[**D.2. Tweet Sentiment Weighting Algorithm Intuition** XXX](#_Toc132325946)

[**D.3. LTS Algorithm Complexity Analysis** XXXI](#_Toc132325947)

[**D.4. UI Wireframes** XXXII](#_Toc132325948)

[**APPENDIX E – IMPLEMENTATION** XXXIV](#_Toc132325949)

[**E.1. Selection Of Programming Language** XXXIV](#_Toc132325950)

[**E.2. Selection Of Deep Learning (DL) Framework** XXXV](#_Toc132325951)

[**E.3. Selection Of User Interface (UI) Framework** XXXV](#_Toc132325952)

[**E.4. Selection Of Application Programming Interface (API) Framework** XXXVI](#_Toc132325953)

[**E.5. Fetch Data** XXXVII](#_Toc132325954)

[**E.6. Preprocessing** XLI](#_Toc132325955)

[**E.7. User Interface** XLIV](#_Toc132325956)

[**APPENDIX F – TESTING** XLIX](#_Toc132325957)

[**F.1. Functional Testing** XLIX](#_Toc132325958)

[**F.2. Non-Functional Testing** L](#_Toc132325959)

[**APPENDIX G – EVALUATION** LV](#_Toc132325960)

[**G.1. Expert Evaluators** LV](#_Toc132325961)

[**G.2. Evaluation Of Functional Requirements** LV](#_Toc132325962)

[**G.3. Evaluation Of Non-Functional Requirements** LVII](#_Toc132325963)

[**APPENDIX H – CONCLUSION** LIX](#_Toc132325964)

[**H.1. Status Of Research Objectives** LIX](#_Toc132325965)

[**H.2. Achievement Of Learning Outcomes** LXI](#_Toc132325966)

[**APPENDIX I – CONCEPT MAP** LXIII](#_Toc132325967)

[**APPENDIX J – EXTENDED REVIEW PAPER** LXIV](#_Toc132325968)

[**J.1. Acceptance Notification** LXIV](#_Toc132325969)

[**J.2. Extended Review Paper** LXV](#_Toc132325970)

# **LIST OF TABLES**

[Table 1: research objectives 7](#_Toc132182700)

[Table 2: Evaluation metrics for TS forecasting systems 27](#_Toc132182701)

[Table 3: Research methodology 29](#_Toc132182702)

[Table 4: Deliverables & dates 32](#_Toc132182703)

[Table 5: Risk management plan 35](#_Toc132182704)

[Table 6: Stakeholder viewpoints 38](#_Toc132182705)

[Table 7: Requirement elicitation methodologies 39](#_Toc132182706)

[Table 8: Literature review findings 40](#_Toc132182707)

[Table 9: Observations findings 40](#_Toc132182708)

[Table 10: Interview thematic analysis codes, themes & conclusions 41](#_Toc132182709)

[Table 11: Survey analysis 42](#_Toc132182710)

[Table 12: Prototyping findings 47](#_Toc132182711)

[Table 13: Summary of findings 49](#_Toc132182712)

[Table 14: Use case description UC:01; UC:02 51](#_Toc132182713)

[Table 15: Functional requirements 52](#_Toc132182714)

[Table 16: Non-functional requirements 53](#_Toc132182715)

[Table 17: SLEP issues & mitigation 55](#_Toc132182716)

[Table 18: Design goals of the proposed system 56](#_Toc132182717)

[Table 19: Dataset sources 66](#_Toc132182718)

[Table 20: Chosen libraries & tools 67](#_Toc132182719)

[Table 21: Chosen IDEs 68](#_Toc132182720)

[Table 22: Summary of chosen tools & technologies 68](#_Toc132182721)

[Table 23: Univariate model evaluation 75](#_Toc132182722)

[Table 24: Multivariate model evaluation 75](#_Toc132182723)

[Table 25: Module & integration testing 76](#_Toc132182724)

[Table 26: Evaluation criteria 78](#_Toc132182725)

[Table 27: Self-evaluation of the author 79](#_Toc132182726)

[Table 28: Categorization of selected evaluators 81](#_Toc132182727)

[Table 29: Thematic analysis of expert feedback 81](#_Toc132182728)

[Table 30: Knowledge utilized from the degree 84](#_Toc132182729)

[Table 31: Problems and challenges faced 86](#_Toc132182730)

[Table 32: Analysis of forecasting algorithms XIV](#_Toc132182731)

[Table 33: Few studies associated with these algorithms XVII](#_Toc132182732)

[Table 34: Stakeholder groups XIX](#_Toc132182733)

[Table 35: Interview participant details XIX](#_Toc132182734)

[Table 36: Interview thematic analysis themes, conclusions & evidence XX](#_Toc132182735)

[Table 37: Survey thematic analysis codes, themes & conclusions XXI](#_Toc132182736)

[Table 38: Use case description UC:03; UC:04; UC:05 XXIII](#_Toc132182737)

[Table 39: Use case description UC:07 XXIII](#_Toc132182738)

[Table 40: ‘MoSCoW’ technique of requirement prioritization XXV](#_Toc132182739)

[Table 41: Complexities of BPTT and adjoint sensitivity XXXI](#_Toc132182740)

[Table 42: Selection of data science language XXXIV](#_Toc132182741)

[Table 43: Selection of DL framework XXXV](#_Toc132182742)

[Table 44: Selection of UI framework XXXV](#_Toc132182743)

[Table 45: Selection of web framework XXXVI](#_Toc132182744)

[Table 46: Functional testing XLIX](#_Toc132182745)

[Table 47: Non-functional testing LIII](#_Toc132182746)

[Table 48: Selected expert evaluator details LV](#_Toc132182747)

[Table 49: Evaluation of the implementation of functional requirements LV](#_Toc132182748)

[Table 50: Evaluation of the implementation of non-functional requirements LVII](#_Toc132182749)

[Table 51: Evaluation of the achievement of design goals LVIII](#_Toc132182750)

[Table 52: Status of research objectives LIX](#_Toc132182751)

[Table 53: Achievement of learning outcomes LXI](#_Toc132182752)

# **LIST OF FIGURES**

[Figure 1: Global market share (Fortune Business Insights, 2021) 11](#_Toc132182753)

[Figure 2: Latest trends (Fortune Business Insights, 2021) 12](#_Toc132182754)

[Figure 3: Proposed architecture (*Self-Composed*) 26](#_Toc132182755)

[Figure 4: Gantt chart (*Self-Composed*) 31](#_Toc132182756)

[Figure 5: Rich picture diagram (*Self-Composed*) 36](#_Toc132182757)

[Figure 6: Stakeholder onion model (*self-Composed*) 37](#_Toc132182758)

[Figure 7: Context diagram (*Self-Composed*) 50](#_Toc132182759)

[Figure 8: Use case diagram (*Self-Composed*) 50](#_Toc132182760)

[Figure 9: Three-tiered architecture (*Self-Composed*) 57](#_Toc132182761)

[Figure 10: Data flow diagram - level 01 (*Self-Composed*) 60](#_Toc132182762)

[Figure 11: Data flow diagram - level 02 (*Self-Composed*) 60](#_Toc132182763)

[Figure 12: System process activity diagram (*Self-Composed*) 64](#_Toc132182764)

[Figure 13: Tech stack (*Self-Composed*) 65](#_Toc132182765)

[Figure 14: Initialize algorithm (*Self-Composed*) 69](#_Toc132182766)

[Figure 15: Build algorithm (*Self-Composed*) 70](#_Toc132182767)

[Figure 16: Algorithm – sensory, stochastic and leakage variables (*Self-Composed*) 70](#_Toc132182768)

[Figure 17: Algorithm – forward propagation (*Self-Composed*) 70](#_Toc132182769)

[Figure 18: Algorithm – define weights and biases (*Self-Composed*) 71](#_Toc132182770)

[Figure 19: Algorithm – Euler-Maruyama SDE solver (*Self-Composed*) 71](#_Toc132182771)

[Figure 20: Univariate model testing (*Self-Composed*) 74](#_Toc132182772)

[Figure 21: Multivariate model testing (*Self-Composed*) 74](#_Toc132182773)

[Figure 22: TensorBoard loss curve (*Self-Composed*) 74](#_Toc132182774)

[Figure 23: Prototype feature diagram (*Self-Composed*) XII](#_Toc132182775)

[Figure 24: Algorithm intuition (*Self-Composed*) XXVI](#_Toc132182776)

[Figure 25: Understanding what an SDE solves XXVI](#_Toc132182777)

[Figure 26: UI wireframes – Home (*Self-Composed*) XXXII](#_Toc132182778)

[Figure 27: UI wireframes – News (*Self-Composed*) XXXII](#_Toc132182779)

[Figure 28: UI wireframes – Cryptocurrencies (*Self-Composed*) XXXII](#_Toc132182780)

[Figure 29: UI wireframes – Cryptocurrency (*Self-Composed*) XXXII](#_Toc132182781)

[Figure 30: UI wireframes – Admin login (*Self-Composed*) XXXIII](#_Toc132182782)

[Figure 31: UI wireframes – Admin metrics (*Self-Composed*) XXXIII](#_Toc132182783)

[Figure 32: UI wireframes – Forecast (*Self-Composed*) XXXIII](#_Toc132182784)

[Figure 33: Fetch historical prices (*Self-Composed*) XXXVII](#_Toc132182785)

[Figure 34: Fetch Twitter volume (*Self-Composed*) XXXVIII](#_Toc132182786)

[Figure 35: Fetch block reward size (*Self-Composed*) XXXVIII](#_Toc132182787)

[Figure 36: Fetch google trends (*Self-Composed*) XXXVIII](#_Toc132182788)

[Figure 37: Scrape tweets (*Self-Composed*) XXXIX](#_Toc132182789)

[Figure 38: Clean tweets (*Self-Composed*) XL](#_Toc132182790)

[Figure 39: Analyze sentiments (*Self-Composed*) XLI](#_Toc132182791)

[Figure 40: Combine and condense tweets (*Self-Composed*) XLII](#_Toc132182792)

[Figure 41: Combine all datasets (*Self-Composed*) XLIII](#_Toc132182793)

[Figure 42: GUI - Home (*Self-Composed*) XLIV](#_Toc132182794)

[Figure 43: GUI - News (*Self-Composed*) XLV](#_Toc132182795)

[Figure 44: GUI - Cryptocurrencies (*Self-Composed*) XLV](#_Toc132182796)

[Figure 45: GUI - Cryptocurrency (*Self-Composed*) XLVI](#_Toc132182797)

[Figure 46: GUI - Admin login (*Self-Composed*) XLVII](#_Toc132182798)

[Figure 47: GUI - Admin metrics (*Self-Composed*) XLVII](#_Toc132182799)

[Figure 48: GUI - Forecast (*Self-Composed*) XLVIII](#_Toc132182800)

[Figure 49: Lighthouse home page (*Self-Composed*) LI](#_Toc132182801)

[Figure 50: Lighthouse login page (*Self-Composed*) LI](#_Toc132182802)

[Figure 51: Lighthouse cryptocurrencies page (*Self-Composed*) LI](#_Toc132182803)

[Figure 52: Lighthouse cryptocurrency page (*Self-Composed*) LI](#_Toc132182804)

[Figure 53: Lighthouse news page (*Self-Composed*) LI](#_Toc132182805)

[Figure 54: Lighthouse metrics page (*Self-Composed*) LI](#_Toc132182806)

[Figure 55: CodeFactor - Algorithm repository (*Self-Composed*) LII](#_Toc132182807)

[Figure 56: CodeFactor - Application repository (*Self-Composed*) LII](#_Toc132182808)

[Figure 57: CodeQL - Algorithm repository (*Self-Composed*) LII](#_Toc132182809)

[Figure 58: CodeQL - Application repository (*Self-Composed*) LIII](#_Toc132182810)

# **LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| **AI** | Artificial Intelligence. |
| **API** | Application Programming Interface. |
| **AD** | Automatic Differentiation. |
| **ARIMA** | Autoregressive Integrated Moving Average. |
| **BPTT** | Back-Propagation Through Time. |
| **BTC** | Bitcoin. |
| **CT-GRU/RNN** | Continuous-time Gated Recurrent Unit / 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. |
| **MVP** | Minimal Viable Product. |
| **N-BEATS** | Neural Basis Expansion Analysis for interpretable Time Series. |
| **NER** | Named Entity Recognition. |
| **NLP** | Natural Language Processing. |
| **POC** | Proof-Of-Concept. |
| **REST** | Representational State Transfer. |
| **RMSE** | Root Mean Squared Error. |
| **RNN** | Recurrent Neural Network. |
| **SOTA** | State Of the Art. |
| **SDE**; **ODE** | Stochastic Differential Equations; Ordinary Differential Equations. |
| **SGD** | Stochastic Gradient Descent. |
| **TS** | Time Series. |
| **UI** | User Interface. |
| **VADER** | Valence Aware Dictionary for Sentiment Reasoning. |
| **XAI** | Explainable Artificial Intelligence. |

# **CHAPTER 01. INTRODUCTION**

# **1.1 Chapter Overview**

In this chapter, a series of top-tier pretrained transformer designs are optimized using automated search hyperparameter optimization in an effort to improve the performance of abstractive text summarization for movie reviews while developing a generalized solution that may be used in other domains. Along with a review of previous studies and a presentation of the anticipated project timetable, the research problem, gap, challenge, and method will be discussed in the work plan.

# **1.2 Problem Domain**

## **1.2.1 Movie User Reviews**

A growing number of websites, like Amazon and the Internet Movie Database (IMBD), a website for movie reviews, allow users to publish reviews for things they are interested in, along with the growth of Web 2.0, where user interaction is prioritized. (Khan, Gul, Zareei, et al., 2020)

Online movie reviews are evolving into an important information source for users, with the continuous increase in data on the web (M and Mehla, 2019). However, online users post a significant number of movies reviews every day, hence making it difficult for them to manually summarize the reviews and determine their interest in the film. One of the challenging problems in natural language processing is mining and summarizing movie reviews. (Khan, Gul, Uddin, et al., 2020).

Text summary assist users or business decision-makers by compiling and analyzing a significant number of online reviews. (Alsaqer and Sasi, 2017).

These days, the majority of people research a film's reviews before selecting or watching it on any platform, such Netflix or Amazon Prime, but we also come across conflicting reviews that can be either good or bad. While most reviews are detailed and require a significant amount of time to review, this develops a problem where users aren't able to make quicker decisions. Therefore, by summarizing the review makes it easier and faster for users to make decisions. This can also help streaming services like Netflix quickly discover the viewing habits or preferences of their users (Dashtipour et al., 2021)

## **1.2.2 Text Summarization**

Today, there is a lot of textual material available, including news stories and reviews. Text summarizing helps us quickly find the key elements of the full piece by minimizing the quantity of text. (Mahajan et al., 2021).

Extractive summarization and abstractive summarization are typically the two methods of text summarization. When extractive summarizing, the most important lines from the context or article are plucked out without being altered in any way. Meanwhile, abstractive summarizing aims to create the sentences on its own and creates the summary; this is superior than extractive summarization since it is more meaningful to generate our own phrases inside the context rather than to utilize selected sentences from the context without any change. (Etemad, Abidi and Chhabra, 2021).

**1.2.3 Transformers**

Transformers in NLP is a novel architecture that aims to solve sequence-to-sequence tasks while handling long range dependencies with ease. It has surpassed competing neural models like CNN (Convolutional Neural Nets) and RNN (Recurrent Neural Nets) in terms of performance to appear as the dominant architecture for natural language processing (Wolf et al., 2020).

Transformers uses self-attention mechanism to target on selected areas of the input sentence followed by the encoder and decoder architecture (Etemad, Abidi and Chhabra, 2021).

# **1.3 Problem Definition**

In the domain of movie review summarization, currently there are no researches done using the latest deep learning approaches (***such as Transformers***) to solve this problem, standard machine & deep learning algorithms such as Naïve Bayes, RNN have been used, the usage of advanced deep learning approaches can be utilized in order to enhance the quality/accuracy of the text summarization.

Deep learning models take longer to train but they provide greater accuracy since they can simultaneously automate feature extraction and classification, whereas machine learning algorithms require feature selection at first. Therefore, applying deep learning techniques will help to improve the quality of text summarization and help the user in making better decisions (Etemad, Abidi and Chhabra, 2021).

## **1.3.1 Problem statement**

No prior research has looked into applying cutting-edge deep learning methods like Transformers to produce abstractive summaries from movie reviews, which can improve text summarization. This solution aims to be generic and accessible to any sector. (Khan, Gul, Zareei, et al., 2020).

# **1.4 Research motivation**

The identified problem can also be applied to several other domains which requires to improve the quality abstractive text summarization using the advanced approaches of deep learning, not only specific movie reviews, this is why a generalized solution was thought of initially (Kouris, Alexandridis and Stafylopatis, 2019).

As mentioned in the work of (Etemad, Abidi and Chhabra, 2021), syntactic and semantic issues with text summarization were the main issues that researchers were concerned on solving. and with respect to their research by exploring multiple deep learning techniques, they concluded that Transformer based models (T5 model) outperformed in all NLP tasks, this encourages the author to go deeper into the field of transformers optimization in order to enhance the quality of text summarization and address the constraints associated with the summarizing of movie reviews.

# **1.5 Research Questions**

**R**Q1: What are the top tier transformer architectures widely used and know for NLP problems related to text summarization?

**R**Q2: How can a pretrained transformer architecture be fine-tuned to get the optimal hyper parameters and to automate it for model retraining?

**R**Q3: What kind of evaluations should we perform after fine-tuning to filter out the best transformer architecture?

**R**Q4: How can domain generalization be integrated for system?

# **1.6 Research Aim & Objectives**

## **1.6.1 Research Aim**

***The aim of this research is to design, develop and evaluate an optimal adaptive generalized transformer architecture from a range of popularly used architectures by fine-tuning via automated hyperparameter optimization, therefore obtaining the recommended architecture's optimum performance***

A fully working system that can be utilized to perform abstractive text summarization based on the user input from any domain (movie, hotel, ecommerce etc.…) will be created by this research project. The quality of the resulting text summary or performance optimization will be the **main points** of emphasis. To get the best result, the usage of data preparation, data analysis, conducting hyperparameter tuning, and evaluating the models will be investigated.

Components will be built, necessary information will be gathered and researched, and performance will be assessed. The system may be utilized for private or public purposes on both a hosted server and a local browser. The data science models' source code will be made available in a public repository for future research and use. A research paper will be published and it end of this study.

## **1.6.2 Research Objectives**

For the research to be considered successful, its goals must be fulfilled

Table 1: Research Objectives (*Self-Composed*)

|  |  |  |  |
| --- | --- | --- | --- |
| **Objective** | **Description** | **LO** | **RQ** |
| Literature Review | Complete a thorough critical review of earlier related work.  **RO1:** Make a preliminary investigation on existing abstractive text summarization using deep learning approaches.  **RO2:** Make a preliminary investigation on why transformers architecture was the chosen deep learning choice for this research.  **RO3:** Analyze the top tier transformer architectures widely used.  **RO4:** Analyzing how the models can be fine-tuned via hyperparameter optimization.  **RO5:** Analyzing the different approaches used for model evaluation.  **RO6:** Analyze how the model can be generalized for every other domain. | LO1, LO4,  LO8 | [**R**Q1](#researchq1),[**R**Q2](#researchq1),[**R**Q3](#researchq1),[**R**Q4](#researchq1) |
| Methodology Selection and SLEP Framework | This defines the outline structure for the requirement analysis and the design process followed by the social legal ethical and professional issues.  **RO1**: Analyzing the Research Methodology approaches.  **RO2**: Analyzing the Development Methodology approaches.  **RO3**: Analyzing the Project Management Methodology approaches.  **RO4**: Analyzing the Solution Methodology approaches.  **RO5**: Analyzing the Social, Legal Ethical and Professional Issues which could develop during the phase of the project. | LO2,  LO6 | [**R**Q4](#researchq1),[**R**Q2](#researchq1),[**R**Q1](#researchq1) |
| Requirement Elicitation | Defining the project's needs utilizing relevant approaches and tools in order to solve the projected research gaps and obstacles based on prior related research.  **RO1:** Gathering information related to the expected metadata required for the dataset to contain for the model training.  **RO2:** Gathering the requirements of transformer architectures for fine-tuning and understand the end to end user expectations.  **RO3:** Getting insights from domain experts to build a suitable system.  **RO4:** Gathering the requirements for handling generalization. | LO1, LO3, LO5 | [**R**Q4](#researchq1),[**R**Q2](#researchq1),[**R**Q1](#researchq1) |
| Design | Considering the following when developing the suggested system:  **RO1:** Design a component to preprocess the dataset for the respective model inputs.  **RO2:** Design a component to store the top tier transformer models with their respective metadata, to use throughout.  **RO3:** Design a hyperparameter tuning component that can improve accuracy of the transformer model.  **RO4**: Design high-level architecture for the system. | LO1, LO5 | [**R**Q2](#researchq2) |
| Implementation | Setting up a mechanism capable of addressing the gaps that were intended to be covered.  **RO1**: To develop data preprocessing component.  **RO2**: To develop a component that handles and stores the top tier transformer architectures for fine-tuning.  **RO3**: To develop the automated hyperparameter search component that handles all the top tier architectures assigned.  **RO4**: To develop a component for the model evaluations for the measured hyperparameters | LO1, LO5, LO7 | [**R**Q2](#researchq2),[**R**Q3](#researchq2) |
| Evaluation | Testing and evaluating the developed system (including the data science models with the suitable metrices)  **RO1**: Performing unit test, integration and performance testing along with a test plan created.  **RO2**: Evaluating all the transformer architectures used for fine-tune experimentations, using recommended scores such as (ROUGE or BLEU SCORE). | LO1,  LO5 | [**R**Q3](#researchq3) |
| Documentation | Keeping track of and documenting the study project's ongoing progress and any challenges encountered. | LO6, LO8 | - |
| Publication | Ensure that the documentation, reports, and papers are well-structured and include a critical analysis of the research.  **RO1**: To publish a research paper on the related work done.  **RO2**: To publish the testing & evaluation results of the work done.  **RO3**: To publish the code implementation repository as public to be access by future research investigations, along with the models and datasets | LO4,  LO8 | - |

# **1.7 Novelty of the Research**

## **1.7.1 Problem Novelty**

The problem novelty of this research is, the lack of attempt to increase transformer performance in order to get better textual summarizing outcomes (Khan, Gul, Zareei, et al., 2020).

## **1.7.2 Solution Novelty**

The solution novelty for this problem is performing an automated approach for hyperparameter tuning & creating a retraining mechanism with newly exposed data to enhance its performance further using the optimal transformer (Etemad, Abidi and Chhabra, 2021).

# **1.8 Research Gap**

Based on previous work done (Khan, Gul, Zareei, et al., 2020) related to abstractive text summarization on movie reviews, the literature identifies for the need of using advanced deep learning approaches to improve the performance of text summarization for this movie domain over traditional machine learning approach.

This project focuses on Empirical gap in the Movie Domain, as well as Theoretical and Performance gaps in the area of transformer optimization. Transformers plays a major role in the field of deep learning especially at problems related to Natural Language Processing, by performing hyperparameter optimization on several transformer architectures we can contribute to the enhanced quality of abstractive text summarization and create a generalized model which can be adapted with the respective domains usage and improve the performance

# **1.9 Contribution to the body of knowledge**

Improving the performance of an existing solution is very common in the field of data science, as we can explore new algorithms or fine-tuning existing algorithms to get better results. The contributions for this project can be classified as theoretical contributions and domain contributions.

The following is a summarization of the authors contribution:

* ***Abstractive Text Summarization:*** *Automated Hyperparameter optimization + Model Retraining + Transformers + Deep Learning*
* ***Movie User Review & Generalization:*** *Research domain target is for Movie reviews, in addition the author makes the system generalized to adapt to any domain area*.

## **1.9.1 Research Domain Contribution**

There are various deep learning techniques that can be used to handle abstractive text summarization, however with respect to previous researches done, (Zhang, Xu and Wang, 2019) it is found that transformers outperform most of the other deep learning approaches as of today but there was no much research on optimizing them for a much better performance.

This research will be focused on creating a generalized solution by achieving the optimized transformer architecture from a couple of the top tier existing architectures, via fine-tuning and performing hyperparameter optimization along with handling abstractive text summarization (Liu and Wang, 2021), therefore we are able to maximize the performance of the recommended architecture. The author plans out to make use of generalization where any domain when used the model will be optimizing and adapting towards their respective domain.

## **1.9.2 Problem Domain Contribution**

Neural Networks makes up the backbone of deep learning algorithms which enables them to process complex unstructured data over normal means of machine learning algorithms (Mahajan et al., 2021). It is found that, the need for using advanced deep learning approaches has not been explored in the domain of movie review summarization.

Given that transformers perform well in this field, the proposed solution for this domain will be finding the recommended architecture along with hyper-parameter optimization, to reach its best performance. An additional contribution will be that, the proposed solution will be generalized to any other domain linked with the field of NLP text summarization.

# **1.10 Research Challenge**

The main objective of this research is to achieve the generalized optimal transformer architecture for the field of NLP abstractive text summarization. Transformers were introduced in 2017 by a team at Google Brain and are the most used choice for NLP problems replacing RNN models, given that this architecture was introduced not much longer back brings to a point where there is a lack of research done in the area of transformer optimization for the purpose of abstractive text summarization. (Wolf et al., 2020).

Therefore, creating and finding the recommended transformer architecture along with the optimal parameters which also handles generalization becomes a challenge with very fewer resources to look up to.

Additionally, identifying suitable datasets for this domain (Movie Reviews Summarization) and Generalization is challenging and necessitates a substantial amount of effort in data preprocessing where it is important since we are dealing with NLP and performance optimization related domain.

# **1.11 Chapter summary**

The author outlined the research effort, explained why the research and problem were innovative, and discussed potential challenges that could develop while attempting to tackle them in this chapter. Additionally, the key goals that must be attained for the research to be deemed successful were outlined and connected to the necessary learning outcomes for the degree.

# **CHAPTER 02. LITERATURE REVIEW**

# **2.1 Chapter Overview**

In this chapter, the author presents critiques on prior relevant work about the use of abstractive text summarization in the domain of movie review summarization, along with the usage of advanced deep learning approaches such as transformers. Additionally, the author tries to create a generalized model that will handle several other domains in addition, not just to only the movie domain.

Finally, the author determines the optimal transformer design that has been improved in order to produce the greatest outcomes by obtaining the optimum set of hyperparameters by model fine-tuning, performing hybridization and adaptive model retraining with new data and hyperparameters.

# **2.2 Concept Map**

The concept map illustrates the project scope that will be addressed in this literature review, and the nodes that are highlighted correspond to the project's primary study areas. The concept map was created to ensure that all necessary literature was covered. The concept map can be found in [**APPENDIX I**](#_APPENDIX_I_–).

# **2.3 Problem Domain**

The simplicity of selling products or services to customers is growing along with the usage of technology and the internet. Sellers utilize customer feedback to better decide how to improve sales and so attain customer satisfaction (Boorugu, Ramesh and Madhavi, 2019). When it comes to movies, people typically find it quite challenging to quickly determine whether a movie meets their demands by reading the reviews, which may occasionally be very lengthy and time-consuming (Khan et al., 2020).

## **2.3.1 User Reviews**

A user/customer review is typically referred to be written feedback from a customer who has used a product or service. Consumers frequently use user ratings and reviews to drive their purchasing decisions. Because the review data is unstructured, it becomes more challenging for consumers to compare and understand lengthier reviews (Lackermair, Kailer and Kanmaz, 2013).

User and customer reviews are extremely important to major corporations like tourism and hospitality as they constitute the primary engine for the country's economic growth and development. where tourists from over the world may blog about their experiences and share their reviews online in numerous formats (Mukherjee et al., 2020).

**2.3.2 Corporate Advantage**

It is also known that it costs at least five times as much time and money to acquire a new customer as it does to keep an existing one, so it is important to learn how to foster customer loyalty to the brand, business, or service that is being offered. Customer satisfaction is essential to the survival of corporate industries. Understanding client expectations through their feedback or reviews helps business industries grow and fix faults (Pizam and Ellis, 1999).

On the other hand, companies like Netflix or Amazon Prime can use the movie summaries to help users and understand the watching pattern or their interest. Likewise, the movie-related industries need to allow the customers to quickly scan the summary and quickly decide whether they should be watching it or not (Khan et al., 2020).

**2.3.3 Text Summarization**

With the massive accumulation of information/data on the internet nowadays, it is extremely difficult to extract relevant information from a large number of textual documents. The goal of text summarizing is to provide a condensed yet meaningful version of a lengthy textual content (Shi et al., 2020).

We all know that text summarization has several uses in a variety of internet-based fields, including search engines that are used for querying and e-commerce sites that utilize sentiment analysis to determine client satisfaction with items (Etemad, Abidi and Chhabra, 2021).

However, in the movie industry, consumers may utilize text summarization to simplify customer reviews of movies, which are often lengthy and time-consuming to read. This enables users to make better decisions when they decide whether or not to watch a certain movie (Khan et al., 2020).

**2.3.4 Abstractive and Extractive Techniques**

Generally, text summarization is classified into two which are; abstractive text summarization and extractive text summarization, however the approach for creating a hybrid model for text summarization is possible (Alsaqer and Sasi, 2017). The abstractive text summarization technique aims to produce the sentences on its own and then uses them to provide a coherent summary. Therefore, the summary's content will vary from the original context yet still convey the same idea (Mahajan et al., 2021). Additionally, it is well recognized that a strong abstractive summary encompasses the input's key details and is linguistically fluent (Zhang et al., 2020).

The extractive text summarizing method focuses on picking out key phrases or groups of phrases from the original input content and combining them to produce a concise yet insightful text summary. It is determined which sentences should be included as parts of the summary based on the statistical and linguistic characteristics of the sentences (Gupta and Lehal, 2010). A hybrid system is one that combines various strategies to produce a single system. However, hybrid text summarizing systems do exist, for instance, using a combination of extractive and abstractive summarization can be utilized to generate a hybrid system that uses encoder-decoders (Kirmani et al., 2019; Abolghasemi, Dadkhah and Tohidi, 2022).

*Table 1: Comparison of Text Summarization Techniques*

|  |  |
| --- | --- |
| **Abstractive** | **Extractive** |
| Paraphrases content like humans do, meaning it creates its own context (Mahajan et al., 2021) | Doesn’t create its own context but uses the best possible phrases from the original document (Gupta and Lehal, 2010) |
| A vast number of datasets are available to experiment working in this domain. | Capable of visualizing sentence scores and investigating gradient-based ways to calculating the contribution of each input token to score prediction (Pai, 2014) |
| There is a probability of creating information which may be faulty or that gives a different in meaning compared to the original text. | There is a possibility that the combined sentences made from the extracted sentences will contain errors. |

**2.3.5 NLP with Deep Learning**

NLP is a method for computers to intelligently and effectively analyze, comprehend, and derive meaning from human language, as opposed to other approaches that only focus on the interactions between human language and computers. Deep learning techniques are increasingly being used in the field of AI compared to traditional machine learning approaches due to their success rates in handling difficult high computing learning tasks (Lopez and Kalita, 2017; Mahajan et al., 2021).

In today's NLP, machine learning is prominent, but for the most part it only involves numerically optimizing the weights of characteristics and representations that have been created by humans. Deep learning aims to investigate how computers can utilize data to create features and representations suitable for challenging interpretation tasks (Socher, Bengio and Manning, 2012).

**2.3.6 Transformers**

Open-source library Transformers contains modern transformer architectures that have been thoroughly developed and are integrated by a common API. Pretraining has enabled the efficient use of this capacity for a wide range of activities, and these designs have permitted the construction of higher-capacity models. Transformers are designed to be easy for practitioners, expandable for researchers, and quick and reliable in industrial deployments (Wolf et al., 2020).

It has been demonstrated that the modern generation of pre-trained language models based on transformers is rather competent at identifying syntactic signals like noun modifiers, possessive pronouns, prepositions, or co-referents, as well as semantic cues like entities and relations (Brasoveanu and Andonie, 2020).

Hugging Face Hub offers a variety of transformer designs, including BERT, GPT2, T5, PEGASUS, and many others. The figure below represents the daily average for unique downloads of the pretrained transformer model architectures between Oct 2019 to May 2020 (Wolf et al., 2020).

*Figure 3.1 – Transformer Architecture Downloads Rate (Wolf et al., 2020).*



(Etemad, Abidi and Chhabra, 2021) research compares various other researchers approaches taken in order to perform abstractive text summarization, these techniques includes the use of transformers and other neural network approaches such as CNN and LSTM RNN networks. The research comparison table below only includes the approaches of transformers used taken abstractive text summarization.

**2.3.7 Hyperparameter Tuning**

Finding the ideal collection of parameter values to train an algorithm using in order to build a model relevant to the dataset is known as hyperparameter tuning (Liu and Wang, 2021). The calculation of the performance improvement that may be obtained by changing the value of each of the considered hyperparameters from the original value to the value indicated in the target configuration set by the tuning strategy is where hyperparameters make the biggest contribution to improving algorithm performance (Joy and Selvan, 2022).

There are several hyperparameters that play a significant role in performance enhancement; however, not all of the parameters do so; just a select handful do, for example, learning rate, weight decay, number of epochs, batch size, and warmup ratio. As a result, giving critical hyperparameters a higher priority is crucial (aws.amazon.com, 2022).

Automated framework tools, such as Optuna, an open-source framework for hyperparameter optimization built on the Python programming language, does hyperparameter tweaking. The application of numerous hyperparameter optimization techniques, including Grid Search, Random Search, TPE, and CMA-ES algorithms, was made easier by this framework (Joy and Selvan, 2022).

**2.3.8 Generalization**

Generalization now plays a significant part in resolving issues in numerous fields that are linked to the same issue. The capacity of a model to generalize to new, previously unobserved data that comes from the same distribution as the model's original data is known as generalization (Neyshabur et al., 2017).

Generalization is a useful strategy for starting with the foundation and improving or specializing in one's field as more unseen domain data becomes available. Therefore, the generalized solution will be able to adapt to even unseen domain data, making this solution to solve a common problem in multiple domain (Zhou et al., 2021).

**2.3.9 Data Expansion**

The quality of a machine learning or deep learning model depends on a number of factors, one of which is the amount and quality of data fed during model training. There are several approaches to increase or expand your available data one of which is data augmentation (making use of existing data points to create new data points). Making using of new data from the users end by saving as the model is used is another way of exposing new data for model retraining (Shorten and Khoshgoftaar, 2019).

When generalized models are required to adapt to become domain specific, model retraining will be considered with new data used by the specialized domain as the application is used.

# **2.4 Existing work**

There have been several works done on abstractive text summarization for the field of movie reviews, mainly using the traditional machine learning algorithms. However, there are several limitations which created the need for recent deep learning approaches in order to improve the **system’s performance.**

## **2.4.1 Text Summarization Systems**

There were multiple studies done previously in the area of text summarization, regarding both abstractive and extractive text summarization. (Khan et al., 2020) research is related to the domain of movie reviews summarization which is the same as this project domain, where the author has developed an automatic approach to summarize lengthy movie reviews along with the feature where the users are allowed to quickly recognize the positive and negative aspects of the movie with respect to the review processed with. The text summarization approach taken by the author is **extractive approach**, where sentence score ranking plays a major role in creating the summary.

The study of (Boorugu, Ramesh and Madhavi, 2019) is towards the domain of ecommerce but yet related to text summarization for customer reviews on the products they sell, so purpose being that, allowing other customer make better purchasing decisions on products, therefore the hassle of going through all the reviews to making a purchasing decision can be reduced to save time, **abstractive approach** is considered to create the summary, which is a better choice of approach.

The research of (Mukherjee et al., 2020) is another **extractive approach** for text summarization where the author develops a solution for generating personalized aspect-based opinion summaries using a dataset which consists of a large collection of online tourist reviews. In addition, the author has gone a step further to personalize the summary's qualities by using the user's interest. However, using abstractive summarization would be a more effective strategy but also challenging when user interest customization is considered because the sentences have been created using own words rather than with any sentence ranking technique.

(Gupta et al., 2021) research is a comprehensive comparison study with benchmarking results of various pretrained transformer architectures such as BART, BERT, T5, PEGASUS etc... for abstractive text summarization which is an **abstractive approach.** This study includes the various types of datasets been used to explore each model with they evaluations as benchmarking results. The author has also concluded the best performing transformer architecture as **T5** by comparing the evaluation results of the study.

The study conducted by (Mahajan et al., 2021) is also an **abstractive approach** to text summarization with the addition of proper grammar and no repeated words used using a deep learning approach with RNN and likewise (Etemad, Abidi and Chhabra, 2021)research also relates to an experimenting study with various deep learning approaches for abstractive text summarization along with the evaluation benchmarking with a goal in search for the best deep learning approach for the problem.

## **2.4.2 Algorithmic approaches for Text Summarization**

As described in this literature review's Problem Domain section, deep learning approaches are mostly given priority to than traditional machine learning approaches as they can handle highly computational tasks. The author came across of multiple deep learning techniques used aswell as machine learning techniques used for handling abstractive text summarization.

The study of (Khan et al., 2020) starts by first focusing on feature extraction, then transforming reviews into vector spaces, and applying the Naive Bayes machine learning method for review classification utilizing an undirected weighted graph-based ranking algorithm to rank score for each review phrase in graph and then, in order to construct the extractive summary, the highest scoring sentences are selected. However, the author has limited the use of sophisticated deep learning algorithms to improve performance by solely using standard machine learning approaches to tackle the problem.

(Boorugu, Ramesh and Madhavi, 2019) research made use of seq2seq model for text summarization along with the attention mechanism for improved accuracy and the Concept net Number batch word embedding model, which is superior than Glove. Utilizing a 1D convolutional layer, a max pooling layer, an LSTM layer, and finally a fully connected layer at the very end. However, the author's use of generic deep learning algorithms to handle this problem introduces a new constraint that prevents performance from being improved using the most recent deep learning strategy for NLP-related problems, transformers.

The research of (Mukherjee et al., 2020) liked mentioned earlier is an extractive method text summarization based on integer linear programming (**ILP** [Unsupervised method]) to choose an informative subset of opinions centered on the identified aspects. Utilize ROUGE-based criteria to assess and contrast the summaries and get competitive outcomes. Since the dataset is also constrained, extractive summaries could not be particularly insightful; thus, utilizing an abstractive technique might produce superior results, despite the dataset's constrained size.

The study of (Mahajan et al., 2021) focus of the authors' study is utilizing the **encoder-decoder** model with the attention layer to produce text summaries with good syntax and no repeated words. the creation of an encoder-decoder model with gated recurrent units and training it to provide an abstract summary of a piece of writing. Although the author employed deep learning, its application in production required real-time training so that it could be updated with the most recent content over time.

## **2.4.3 Usage of Transformers**

(Gupta et al., 2021) research employed pretrained models such Pipeline BART, BART modified, T5, and PEGASUS to deal with text summarization as a part of the comparison study done. The ROUGE Scores were used as the evaluation measures. During the experiments, the author employed transformer designs; however, the **hyperparameters** used were **default** and might be tuned for a better performance. The constraints consist of concentrating on developing more reliable models that can further expand the method to produce summaries of varying length and applicable for multi-document summarization.

(Etemad, Abidi and Chhabra, 2021)The author explores with deep learning methods in the broad text summarization domain to determine which method—among a collection that includes RNN, CNN, and Transformers—performs best. The author also considers metrics for model evaluations including BLEU and ROUGE, despite using sophisticated deep learning algorithms, the author was unable to undertake **hyperparameter** tuning to improve the method and obtain a better outcome.

# **2.5 Technological review**

There are many applications for text summarization systems today, especially when researching papers. Users may choose from a variety of contexts, such as research paper materials, customer reviews, etc., much more easily by using summaries to comprehend the context and pinpoint the key concepts. Text summarization tools assist researchers in frequently writing an abstract of their findings. With this technique, text summaries may be extracted or abstracted. In contrast to extractive text summarizing, the abstractive text summarization approach creates its own context, which is a far more logical or human-like written language, and can help with problem solving (Barna and Heickal, 2022). Text summarizers may be quite helpful in highlighting the key elements of reviews by providing a summary of user reviews, which can sometimes be very extensive and descriptive.

Traditional machine learning and deep learning approaches has been widely used for text summarization for the domain of movies reviews, however advanced deep learning approaches such as Transformers has not been explored for the domain of movie reviews but yet been used in other case scenarios. Even though traditional machine learning and deep learning approaches performed well, there was a limitation to push the boundaries with new approaches. That’s where transformer optimized was considered via repeated hyperparameter tuning with exposure to new data and making this generalized to any domain.

## **2.5.1 Proposed architecture for the Generalized Text Summarization System.**

Figure 3.2 – Proposed Generalized Abstractive Summarization System Process Flow



## **2.5.2 Machine Learning Text Summarization Techniques**

(Boorugu, Ramesh and Madhavi, 2019) points out a previous research where a system was built that uses a hybrid classifier approach with machine learning algorithm combination of SVM and Naïve Bayes in sync with fuzzy logic and they also concluded that with the increase in the classifier count the accuracy can also be increased. They also made use of supervised ML algorithms such as KNN for the classification of the reviews which then combining appropriate words for identifying the features of the product.

(Khan et al., 2020) proposed system was for the movies domain using the customer reviews, the author broke down proposed methodology into segments of which is preprocessing, feature extraction, review classification and finally review summarization. The Nave Bayes (NB) classification method, which is regarded as a robust classifier and may achieve greater accuracy, was used to categorize the reviews from negative to positive using supervised ML classification technique, It is clear that an extractive summarization approach was used because the text summarization phase was completed in several stages, starting with the creation of a graph from classified reviews, followed by the ranking of graph nodes and the selection of the top rank sentences for the summary generation.

Initially, these machine learning methodologies were given a lot of significance, but as time has progressed on, new technologies and techniques have emerged that can utilize deep learning techniques like RNN, CNN, etc. to perform better.

## **2.5.3 Deep Learning Text Summarization Techniques**

Numerous studies have been conducted on deep learning methods for abstractive text summarization, such as with the usage of CNN, LSTM-CNN, Convolutional Seq2Seq, Sequence to Sequence RNN, Convolutional Sequence to Sequence, Transformers, T5, BART, BERT etc.… which were trained on a general dataset such as from Gigaword, DUC 2002, DUC 2004, CNN Daily Mail, DUC, Xsum, Newsroom such datasets, in order to get an evaluation comparison on which outperforms the rest and eventually the T5 Transformer outperformed the rest of the other techniques in the case of abstractive text summarization (Etemad, Abidi and Chhabra, 2021).

(Shi et al., 2020) has conducted a thorough analysis of latest developments in seq2seq models for the task of abstractive text summarizing. The author's analysis includes a full review of several distinct seq2seq models for abstractive summarization.

Out of which transformers are the advanced deep learning approach for text summarization which is an encoder-decoder model with attention layer which helps it to generate better results than a traditional simple RNN architecture (Mahajan et al., 2021).

## **2.5.4 Available Datasets for generalized text summarization**

There are two datasets that the author will be exploring throughout the development of this project. One of which is the Amazon movie reviews dataset from Stanford University Education, which contains data within the span period of more than 10 years including 8 million review data records (McAuley and Leskovec, 2013).

This dataset will be used to test out the solution for the problem domain which is abstractive text summarization for movies. Given that the author is able to create the solution for the domain of movies then, the author then plans to generalize the solution using another dataset named as Gigaword which is from TensorFlow datasets which was used previously for creating generalized content for text summarization (Kouris, Alexandridis and Stafylopatis, 2019).

## **2.5.5 Available Datasets for Generalized Text Summarization**

There are two datasets that the author will be exploring throughout the development of this project. One of which is the Amazon movie reviews dataset from Stanford University Education, which contains data within the span period of more than 10 years including 8 million review data records (McAuley and Leskovec, 2013).

This dataset will be used to test out the solution for the problem domain which is abstractive text summarization for movies. Given that the author is able to create the solution for the domain of movies then, the author then plans to generalize the solution using another dataset named as Gigaword which is from TensorFlow datasets which was used previously for creating generalized content for text summarization (Kouris, Alexandridis and Stafylopatis, 2019).

## **2.5.6 Preprocessing Techniques used in Text Summarization**

Text preprocessing is very important when it comes to dealing with text related data. In earlier studies, a variety of text preprocessing approaches were utilized for text summarization.

Sentence segmentation is a fundamental step in NLP applications including IR, machine translation, semantic role labeling, and summarization. It is the process of identifying boundaries within a document that divides the document's text into sentences, typically from a strong point of punctuation like (full stop, explanation mark, question mark, etc.), Tokenization and stop words removal will then be performed. Tokenization will be carried out by the tokenizer program to split the sentences into distinct words by splitting them at whitespaces such as blanks, tabs, and any strong punctuation. Stop word removal is also used to remove frequently used words in the document such as "I," "an," and "a" because these words carry little meaning and are best removed from the document (Khan et al., 2020).

Other researchers have incorporated a variety of other techniques, including noise removal, which eliminates unnecessary text from the input document, such as the header and footer, and named entity recognition (NER), which recognizes words in the input text as names of things like people, places, and things, among others (Barna and Heickal, 2022).

Datasets may also contain unwanted records, null records, or redundant records that are absolutely useless. These records or rows with null values are eliminated, unnecessary HTML tags and URL links are also filtered off from the text as a part of text preprocessing. Contraction mapping is crucial and this will be handling which are converting short word formats into longer such as “aren’t” into “are not”. Converting the entire text content into a single case most preferably to lowercase, therefore further character filtration would become very simpler (Mahajan et al., 2021).

# **2.6 Evaluation Techniques**

## **2.6.1 Evaluation Approaches**

A machine learning model's performance, as well as its advantages and disadvantages, are understood through the process of model evaluation, which employs many evaluation measures. During the early stages of research, it's critical to evaluate models to determine their efficiency.

The table below shows the available measure and the metrics that can been used to **quantitatively** evaluate the text summarization system.

Table 2: Evaluation metrics for abstractive text summarization

|  |  |  |  |
| --- | --- | --- | --- |
| **Measure** | **Description** | | **Objective Orientation** |
| **Metric: 01** | | | |
| ROUGE | ROUGE also known as Recall-Oriented Understudy for Gisting Evaluation. Measures are made by comparison between an automatically generated summary/translation against a group of reference summaries (generally human created summaries) (Lin, 2004). ROUGE measures the **recall,** (according to how frequently the terms from the summaries created by humans appeared in those computer - generated.) | | Positively oriented.  Higher, the better |
| **Equations** | | | |
| **Rouge-1** | | | |
| Refers to the system summary and reference summary's overlap of unigrams (one-word sequence)*.* | | | |
| **Rouge-2** | | | |
| Refers to the bigram (two-word-sequence) overlap between the system and the reference summaries | | | |
| **Rouge-L** | | | |
| Measures the longest matching word sequence | | | |
| **Metric: 02** | | | |
| BLEU | | BLEU also known as Bilingual Evaluation Understudy is a metric used for evaluation for the quality of machine generated text by comparing it with a reference text that is supposed to be generated. (Steinberger and Jezek, 2009). BLEU measures the precision (as to how much words in the generated summaries appeared in the human generated summaries) | Positively oriented.  Higher, the better |
| **Equations** | | | |
|  | | | |

Different versions of ROUGE exist, including ROUGE-1, ROUGE-N, ROUGE-L, and ROUGE-S. For example, ROUGE-L considers the longest common sequence, whereas ROUGE-S and ROUGE-SU consider skip sequences. (Etemad, Abidi and Chhabra, 2021). Out of which ROUGE-1, ROUGE-2 and ROUGE-L is considered as the least number ways to get a proper evaluation of the model and the scores lies between 0 to 1. Higher the score, the better.

(Steinberger and Jezek, 2009) Out of both of these evaluation metrics ROUGE score demonstrates the best performance for text summarization as compared to BLEU. (Lin, 2004)claims that they introduced ROUGE, an evaluation package for summarization, and carried out thorough evaluations of the automated measures present in the ROUGE package using three years' worth of DUC data.

## **2.6.2 Benchmarking**

The table given below is the benchmarking results of training transformers with generalized datasets for abstractive text summarization. The author will be also following the same measures for the evaluation benchmarking for the prototype, so it can be comparable.

*Table 3.1 – Comparison table for abstractive text summarization using transformers (Etemad, Abidi and Chhabra, 2021).*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Researcher | Year | Type of model | Rouge 1 | Rouge 2 | Rouge L | Dataset |
| Haoyu Zhang et al. | 2019 | Transformer with BERT | 41.71 | 19.49 | 38.79 | CNN-Daily Mail |
| Andrew Hoang et al | 2019 | Transformer | 39.01  36.73  40.87 | 17.87  14.93  28.59 | 36.17  29.66  37.62 | CNN Daily Mail  Xsum  Newsroom |
| Kaiqiang Song et al. | 2019 | Transformer | 40.89  45.93 | 19.11  24.14 | 37.60  42.51 | Gigaword,  Newsroom |
| Mike Lewis et al. | 2019 | BART | 44.16  45.14 | 21.28  22.27 | 40.90  37.25 | CNN Daily Mail  Xsum |
| Itsumi Saito et al. | 2020 | RoBERTa Base | 45.80  45.42 | 22.53  22.13 | 42.48  36.92 | CNN Daily Mail  Xsum |
| Beliz Gunel et al. | 2020 | Transformer XL | 34.273 | 13.018 | 32.048 | CNN Daily Mail |
| Colin Raffel et al. | - | T5 | 43*.*52 | 21*.*55 | 40*.*69 | CNN Daily Mail |

# **2.7 Chapter summary**

The problem, technology domains, prior work, and assessment strategies were all broken down in a concept map at the beginning of this chapter. Then, these four areas were further divided into subtopics and examined based on the research and ideas presented in earlier works of literature. A critical analysis of all the literature has been conducted, contrasting the likes and dislikes of earlier research, potential future work described in the literature, and unique approaches that the author of this study proposes as potentials not before explored.

# **CHAPTER 03. METHODOLOGY**

# **3.1 Chapter overview**

To make sure the research process runs smoothly, it's important to have a clear definition of the methodology used. In this chapter, the author explains the selected methodologies and provides detailed explanations for their choices. Additionally, the chapter presents the project's needs, potential risks, plans to mitigate those risks, timeline, a plan for dividing work, and the expected results.

# **3.2 Research methodology**

When determining the quality of a project, there are a number of important factors to consider, including the cost incurred, the amount of time required, and the weight given to the project's scope. These factors must be effectively managed throughout the project's lifespan, which is when methodologies are required.

The table listed below are the chosen methodologies for the project, where Saunders Researched Onion model has been used (Saunders, Lewis and Thornhill, 2007).

Table 3: Research methodology

|  |  |
| --- | --- |
| Research Philosophy | The author will explore and experiment with numerous techniques as part of a combined strategy to determine which is most effective for reaching the research aim, therefore the **pragmatism** approach was chosen among the positivism, pragmatism, realism, and interpretivism approaches. |
| Research Approach | This research experiments with several approaches to figure out the best, the **deductive approach** was taken into consideration this was because the research aims at applying a combination of existing model architectures to fine-tune and get the best. As the data analysis **qualitative method** were chosen. |
| Research Strategy | This area focuses on data collection with respect to the research questions created. **Survey and experiments** were the strategies considered to address the research questions. Both of these strategies are expected as an approach for the **quantitative result** at evaluation. |
| Research Choice | Weather the research is concerned with qualitative or quantitative aspects depends on the choice of methodology.  Even though we ultimately prioritize quantitative findings mainly, **multi-method** was taken into consideration for this study. This is partly because determining the qualitativeness of the data utilized for development is important since, in the end, it will influence the quantitative outcomes. |
| Time Horizons | **Cross-sectional** will be used since only during the requirement engineering and evaluation phase the data will be gathered and therefore not repeatedly collection over time. |
| Techniques and procedures | Here, data collecting and analysis methods are considered.  We'll utilize sources including internet news, discussions, reports, surveys, publications and organizational records. |

# **3.3 Development methodology**

## **3.3.1 Life cycle model**

The project's research development methodology of choice was the **Agile** Software Development Life Cycle. This is a result of the project's reliance on an iterative development method.

## **3.3.2 Requirement elicitation methodology**

Conducting Surveys via questionnaires, review more previous research done, experimenting with various transformer architectures and brainstorming will be the approaches taken in-order to communicate and gather **insights** for the projects need.

## **3.3.3 Design methodology**

The system's simplicity of expansion and future growth was considered by the author to support incremental methodology, therefore **SSADM (Structured Systems Analysis and Design Method)** was chosen as the Design Methodology for the project.

## **3.3.4 Software development methodology**

**Functional programming** methodology will be used for the development methodology for the project, this is due to the project's ease of future developer enhancement, making it simpler.

# **3.4 Project management methodology**

Prince2 is the chosen approach for project management. This aids the author's ability to be extremely flexible as well as operate in controlled environments.

## **3.4.1 Schedule**

### **3.4.1.1 Gantt chart**

The project’s schedule is presented as a Gantt chart below.

****

Figure 4: Gantt chart (*Self-Composed*)

### **3.4.1.2 Deliverables**

The deliverables and respective dates are specified in the table below.

Table 4: Deliverables & dates

|  |  |
| --- | --- |
| Deliverable | Date |
| **Project Proposal Document + Ethics Forms**  Initial proposal of the project. | 9th November 2022 |
| **Software Requirement Specification**  Documentation outlining the requirements that must be met, designed as the ultimate prototype, including data collection methods. | 24th November 2022 |
| **Proof of Concept with Implementation Presentation**  Performing a presentation regarding the implementation along with the proof of concept | 23rd December 2022 |
| **Project Specifications Design & Prototype**  A functional prototype with all its main features included as stated. Along with a documentation of the design approach followed. | 2rd February 2023 |
| **Test & Evaluation Report**  Documented Evaluation Report conducted on the Prototype. | 23rd March 2023 |
| **Draft Project Reports**  A draft thesis submission, in order to get supervisors feedback | 30th March 2023 |
| **Final Thesis**  Final report detailing the research and project decisions | 27th April 2023 |
| **Review Research Paper**  A review paper reviewing published existing systems in handling abstractive text summarization. | 2nd May 2023 |
| **Final Research Paper**  A research paper about the experimentations done with the transformers hyperparameters. | 15th May 2023 |
| **Public project repository**  A publicly accessible project repository to setup and test the development | 30th April 2023 |

# **3.5 Resources**

## **3.5.1 Software requirements**

* ***Operating System* –** Microsoft Windows OS will be used for the research, documentation and for the complete project implementation (end to end), due to its flexibility compared to another operating system.
* ***Python* –** Machine learning & Deep learning model development and APIs creation to serve the models and handle logic will be implemented by using the Python language. Python is a general-purpose language that has been used most widely in data-science related projects and in backend frameworks link Flask and Django.
* ***Flask* –** Backend web framework for API development for the prototype. This will be used to access/transfer data to and from the data science models developed.
* ***TensorFlow/Scikit learn Python packages* –** Libraries that will be used during the development of the data science models.
* ***Jupyter Notebook / Google Colab* –** Used for Machine-learning/Deep learning model development in this project, it’s an Integrated development environment for programming.
* ***TypeScript (React)* –** JavaScript framework which is used for the development of the frontend application interface of the project. Here is where the user will be able to input and view their data.
* ***Vscode* –** The project's development environment. This will be utilized while creating the codebase for the backend API and frontend development.
* ***Zotero* –** Referencing software that keeps a copy of all the articles as well as managing the references for research papers
* ***MS Office/ Google Docs/ Figma* –** Software & tools which will be used to create figures, reports and handle documentations.
* ***Google Drive/ GitHub* –** Backup platform and code management system to help keep backup of all documents and code.
* ***Git* –** Version control system which will be used to keep track of the changes made in the project code and manage code changes.
* ***Firebase* –** Application development platform which helps to build and grow apps, its also known as the Backend As a Service.

## **3.5.2 Hardware requirements**

* **Core i5x Processor (8th generation) or above –** Above average processing power required to perform high resource intensive tasks (such as model training).
* **Nvidia MX130 GPU or above –** To handle data science model training processes.
* **16GB RAM or above –** Sufficient amount of RAM needed to run multiple applications (client + server), model training also consumes a lot of CPU and RAM.
* **Disk space of 30GB or above –** To store project data and applications.

## **3.5.3 Technical skills**

* Good understanding about machine learning and deep learning concepts.
* Good understanding about Natural Language Processing and its data preprocessing methods.
* Good understanding about transformers and how to work with hyperparameters in general along with the knowledge of its use.
* Research writing skills

## **3.5.4 Data requirements**

* Amazon Movie review data **–** From Stanford University Education.
* Gigaword, Xsum & CNN Daily News dataset – From TensorFlow datasets which will be used for generalization model.

# **3.6 Risks & mitigation**

The table given below defines the possible risks which can be encountered during the process of the project development along with the possible mitigation steps.

Table 5: Risk management plan

|  |  |  |  |
| --- | --- | --- | --- |
| Risk | Magnitude | Frequency | Mitigation Plan |
| Losing the development project codebase/repository | 5 | 4 | Using GitHub and external backup to keep a latest copy of the project codebase. |
| Personal computing breaks down during the project timeline progress. | 5 | 4 | Upload the complete backup to GitHub and Google Drive, use University Lab service to continue project work, till personal machine recovery. |
| Unable to complete all mentioned project deliverables on time | 4 | 4 | Prioritize and create a timeline to complete the deliverables. |
| Project documentation corruption | 5 | 3 | Use a dedicated folder under the same GitHub repository and push all latest documentation changes & use cloud-based documentation approach |
| Insufficient knowledge on the project domain | 5 | 3 | Performing an intensive research on the problem domain along with the research domain. |
| Any unavoidable personal health risk – Sickness | 3 | 1 | Create weekly goals to complete and keep them updated. |

# **3.7 Chapter summary**

In this chapter, the author explained the methodology they used and provided an explanation for their choice. The author also gave detailed descriptions of the research, development, and project management methodologies they followed. Additionally, the chapter outlined the project requirements, work division plan, expected outcomes, and identified potential risks along with their mitigation strategies.

# **CHAPTER 04. SOFTWARE REQUIREMENTS SPECIFICATION**

# **4.1 Chapter overview**

In this chapter, the author describes how to identify the essential needs and how to gather them. To carefully record the engagement of possible stakeholders, their interaction points, and their separate responsibilities, a rich picture diagram and stakeholder onion model are used. The chapter also discusses the methods used for requirement gathering and the results that were used to create functional and non-functional requirements, use case diagrams, and prototypes.

# **4.2 Rich picture**



Figure 5: Rich picture diagram (*Self-Composed*)

The diagram above depicts a bird's-eye view of the surrounding region, as well as how certain stakeholders might interact with the system and profit from it. Along with the knowledge gained by the researcher to improve the system, the potential negative impacts on the design and prospective critical analyses are also identified.

# **4.3 Stakeholder analysis**

The section that follows acknowledges significant stakeholders involved with the system, their relationships, and their individual roles. The stakeholder onion model represents this information, and stakeholder perspectives elaborate on it.

## **4.3.1 Stakeholder onion model**



Figure 6: Stakeholder onion model (*self-Composed*)

## **4.3.2 Stakeholder viewpoints**

The table below provides information about the stakeholders, their functions, and the actions related to them.

Table 6: Stakeholder viewpoints & Requirements (self-Composed)

|  |  |  |
| --- | --- | --- |
| **Stakeholder** | **Role** | **Benefits/Description** |
| Developer | Functional beneficiary | Works on developing the system |
| Investors | Profit is generated through system investment and money from marketing and user subscriptions. |
| Product Owner | Owner & Operational admin | Owns the system and has control over the system |
| Data Scientists | Quality Control Regulator | Provides performance enhancements for the models and algorithms used in data science. |
| Data Engineers | Gives guidance on potential data that may be used to generate the best suggestions possible. |
| AI Researchers | Conduct research in the specified area to enhance and implement reliable text summarizing models. |
| NLP Experts | Offers specialized guidance and insights on the field  knowledge, to enhance the functionality of the system. |
| Domain Specific Manager | Operational Beneficiary | Text reviews are used as inputs for abstractive summarization, and the model is retrained with prior inputs as new data to increase performance. |
| General Users | Unless specifically assigned or retrained, typical users will utilize a general abstractive summarization model. |
| Operational Staff | Ensures that the system is up and functioning while responding to user requests and problems. |
| DevOps Engineers | Product Deployment & Maintenance | Makes ensuring the system is up and running in the cloud and is serving users without being throttled |
| Hackers | Negative Stakeholder | May manipulate the review data stored in the database which will affect the retraining process. |
| Competitors | May build competing systems that may outperform the existing system. |
| Evaluators | Quality Inspector | Checks to see if the system is ready for production use and puts it through its paces. |
| Supervisor | Checks to see if the system development is progressing well without any issues. |

# **4.4 Selection of Requirement Elicitation Methodologies**

There were several requirement elicitation approaches used to collect needs for the creation of the research project. The approaches selected for this were literature review, survey, interviews, prototyping, brainstorming and self-evaluation. The following is a discussion of the rationales behind selecting the mentioned requirement elicitation approaches.

Table 7: Requirement elicitation methodologies (*Self-Composed*)

|  |  |
| --- | --- |
| **Method** | **Description** |
| Literature Review | To determine research gaps in the chosen domain of interest and the intended topic of study at the project's outset, the author conducted a thorough literature analysis. Current systems were researched together with comparable technologies that might be applied to the existing systems that were referenced in literature in order to discover research gaps available in technologies that can be used. |
| Survey | A questionnaire was utilized as a survey instrument to obtain requirements and opinions from possible users of the suggested system. The author will benefit from this sort of poll in understanding people's thought processes and expectations for the prototype. It will also enable the author to explain whether or not the targeted users will benefit from the suggested solution. |
| Interviews | Interviews were performed to gain expert insight into domain-specific requirements and to determine the best method to address the issue at hand while adding to the body of knowledge through research. Interviews were determined to be the greatest source of information because the field is new and the technical expertise needed is very precise. Additionally, this technique allowed for the qualitative evaluation of the suggested system, allowing for the identification of any shortcomings or difficulties that could need to be resolved during prototyping. |
| Prototyping | The project was chosen to follow the Agile Software Development Life-cycle, thus prototyping would allow the author to test and evaluate the prototype while iteratively trying out several alternative implementations to find any potential areas for improvement. |
| Brainstorming | Whether you're attempting to come up with a broad subject before you start your research, you're trying to focus more specifically, or you're determining what evidence to use for a particular paragraph, brainstorming is a useful technique to produce ideas at every step of the process. In order to assess the system for personally, the author has a number of brainstorming sessions with his colleagues at various project stages. |
| Self-Evaluation | Self-evaluation is done in order to examine the currently available applications, do competitor analyses on the current systems, and get insight into how negative stakeholders, such as hackers, can breach the system and find a way around to protect the data and the system. |

# **4.5 Discussion of Findings**

The relevant key stakeholders are split up into groups where the chosen best methodology was used for each group. [**APPENDIX C.1**](#_C.1._Requirement_elicitation) contains a complete breakdown of these stakeholders.

## **4.5.1 Literature review**

Table 8: Literature review findings (*Self-Composed*)

|  |  |
| --- | --- |
| **Discussion of Findings** | **Citation** |
| In the completion of the literature review on the existing work done, it was identified that abstractive text summarization systems for customer reviews helps users to make better and quicker decisions on their actions let it be on buying products or watching a movie, user review summarization proves to save time for customers. | (Boorugu, Ramesh and Madhavi, 2019) |
| When exploring technologies that can be applied to achieve the required outcome, it was clear that traditional machine learning and deep learning approaches were only used for abstractive text summarization in the domain of movie reviews. Leaving the usage of advanced deep learning approaches such as Transformers untouched for this domain. | (Khan et al., 2020) |
| It was identified that transformer optimization has not been looked into when working with transformers in abstractive text summarization domain in general and not specific to the movie domain. | (Gupta et al., 2021) |
| Dataset related to working with model generalized has been used previously and is recommended to be used if researchers are willing to work with the idea of generalization for the domain of abstractive text summarization. | (Kouris, Alexandridis and Stafylopatis, 2019) |

## **4.5.2 Brainstorming**

The author engaged in brainstorming across various project phases. These were carried out both with the authors' colleagues and supervisors as well as through a self-analysis process.

Table 9: Observations findings (*Self-Composed*)

|  |  |
| --- | --- |
| **Criteria** | **Discussion of Findings** |
| Able to figure out several other research gaps/ limitations which can be fit into the current project domain in order to increase the magnitude of research effort. | Multiple ideas were brought up as the result of the brainstorming session. The concept of creating a performance adaptive generalization model was brought up by the authors supervisor, along with several other approaches to increase the performance of the system exponentially such like making use of the new data from the domain users for retraining and combine all data with the common domain for retraining since the data count increases with respect to the common domain user. |

## **4.5.3 Interviews**

Interviews with experts and researchers in the relevant domains were performed to obtain insights on the technical domain competence. To determine the project requirements, experts and researchers in ML and abstractive text summarizing systems were chosen. 2 PhD candidate in ML and Computational Linguistics, 1 NLP Researcher, 2 Software Architects, 1 Software Engineer and 1 Lecturer with MSc completion were interviewed. The interview outcomes were processed to a thematic analysis based on the following themes. Interview participant details can be found at over [**APPENDIX B.3**](#_B.3._Interview_analysis)

Table 14: Interview thematic analysis (*Self-Composed*)

|  |  |
| --- | --- |
| **Code** | **Theme** |
| Data handling | Data Collection & Data Preprocessing |
| Transformer architectures | Best performing transformer architectures |
| Generalization | Handling adaptive generalization |
| Research scope | Research gap and scope |
| Hyperparameter tuning | Automatic hyperparameter tuning & model retraining |
| Hybrid transformers | Looking into hybrid transformer combinations |
| Custom transformers | Customizing the transformer architecture |
| Prototype | Prototype features and suggestions |
| Business benefits | Understanding which and how businesses would benefit |
| Evaluations | Understanding the importance and evaluation ways |

|  |  |
| --- | --- |
| **Theme** | **Conclusion** |
| Data handling | Data accessibility and data preparation techniques are crucial considerations for a data science project. Since every domain would initially employ the same model, PhD candidates proposed utilizing validated and well-researched datasets for the field of generalization to ensure the quality of data. Since text data may contain characters from other languages unless the project is restricted to English language support alone, NLP experts raised worry about the language of the text included in the project scope. |
| Transformer architectures | The interviewees stated that NLP tasks like text summarization and sentiment analysis may be successfully solved using transformer designs like BERT, GPT-2, Roberta, T5, and others. They advised using the most recent version of these models because they are frequently upgraded and developed in new, improved forms. To keep track of these upgrades, they advised examining daily analytics from websites like Hugging Face, such as download and like counts. |
| Generalization | Due to their scalability and performance advantages, the software engineers and architects suggested employing NoSQL databases like MongoDB or Firebase for storing data related to domain specific managers. |
| Research scope | Experts agree that using optimized transformers to solve the issue is a brilliant idea, but given the project's time constraints, they advise prioritizing the movie domain and postponing the task of developing a broader adaptive solution. |
| Hyperparameter tuning | The NLP researchers and Lectures suggested several ways of using tools and libraries to help with hyperparameter tuning since doing this manually is very time consuming and unnecessary effort. |
| Hybrid transformers | The hybrid transformer combination with ensemble techniques was well-loved by PhD applicants, but many believed the project scope was growing too large and risky for the time available. |
| Prototype | Interviewees are interested in the novel domain-specific retraining system and recommended incorporating a pretrained model for sentiment analysis if time allows. |
| Business benefits | Most of the interviewees suggested the Movie domain, Tourism, Ecommerce, Book, Researchers would find this useful in summarizing their customer reviews on their businesses. |
| Evaluations | PhD candidates and NLP experts emphasized the need for evaluations in the adaptive generalization model and suggested limiting the project to a maximum of three domains for easier comparison and clearer demonstration |

## **4.5.4 Survey**

Table 12: Survey analysis (*Self-Composed*)

|  |  |
| --- | --- |
| **Question** | Have you ever realized that reading lengthy reviews takes a significant amount of time? |
| **Aim of question** | To determine whether the audience as a whole considers reading lengthy reviews to be a time-consuming activity. |
| **Findings & Conclusion**  It can be concluded that a large part of the audience (more than 90% of the audience) finds that’s reading lengthy reviews is a time-consuming hassle which also proves that they would appreciate if there would be a quicker approach for this problem, like a summarization. This also concludes to see a positive correlation from the results which was expected from the author of the project. | |
|  | |
| **Question** | Do you believe that developing a generic system for all domains would be a wise course of action? |
| **Aim of question** | Ensuring that developing a generic system would be beneficial in all domains |
| **Findings & Conclusion**  It can be concluded that most of the participants (more than 90% of the audience) agrees that developing a generalized system which can adapt to the domain as they use, is beneficial and worth the effort to process with the project research. This also concludes to see a positive correlation from the results which was expected from the author of the project | |
|  | |
| **Question** | Who do you think will most benefit from this system? |
| **Aim of question** | Getting to know about the thoughts of the participants about to whom the system would mostly benefit from? |
| **Findings & Conclusion**  It can be concluded that a majority of the participants (more than 60%) finds that this system will benefit the movie, restaurant, tourist, hotel, ecommerce domains (these domains were considered since they are mostly interacted with the users on a daily bases and uses customer reviews for their domain as a part of their business) as well as the general users. | |
|  | |
| **Question** | How much do you think that this system would benefit you? |
| **Aim of question** | Getting to know how much the system would benefit the general participants which are NOT domain specific |
| **Findings & Conclusion**  From the statistics graph, it can be concluded that roughly 75% of the audience finds that the system would benefit them for their general work or needs given that it’s not domain specific to them, which is a positively correlated result from the achieved statistics. | |
| **Question** | How much do you think that this system would benefit businesses? |
| **Aim of question** | Getting to know from the participants as to how much the system would benefit businesses/domains in solving this problem. |
| **Findings & Conclusion**  From the statistics graph, it can be concluded that roughly 84% of the audience finds that the system would benefit the businesses, which is a positively correlated results from the achieved statistics and that’s what the author expected to achieve. | |
|  | |
| **Question** | Before making a reservation or booking a movie or a hotel, do you read the customer reviews? |
| **Aim of question** | Getting an idea from the audience if in general they give importance to customer/user reviews to any domain before consuming their product or services. |
| **Findings & Conclusion**  It can be concluded that a majority of the participants (more than 95% of the audience) agrees that they value and read customer reviews before they consume one’s product or service. Therefore, making customer reviews a major contributing factor for business growth. | |
|  | |
| **Question** | How much you think customer reviews are important with respect to any domain? |
| **Aim of question** | Getting an idea from the audience to see how much they value customer reviews. |
| **Findings & Conclusion**  From the statistics graph, it can be concluded that roughly 90% of the audience finds that customer/user reviews are very important irrelevant to the domain, which is a positively correlated results from the achieved statistics and that’s what the author expected to achieve. | |
|  | |
| **Question** | Which additional features would you want to see in this system. |
| **Aim of question** | To identify the systems non-functional requirements which could potentially improve the system. |
| **Findings & Conclusion**  The majority of participant responses were concerned with classifying the review text's sentiment after it had been summarized and with managing a list of review uploads so as to add filtering for the summarized review text based on the sentiment, whether it was positive or negative, along with the sentiment score. | |

Table 13: Survey thematic analysis (*Self-Composed*)

|  |  |
| --- | --- |
| **Code** | **Theme** |
| Convenience | User-friendly |
| Adjustability | Flexibility |

|  |  |
| --- | --- |
| **Theme** | **Conclusion** |
| Convenience | A group of participants required to upload more than one review and a time/bulk at once. |
| Adjustability | A majority of the participants requested for sentiment of the summary and the sentiment score to be also included with the output. |

## **4.5.5 Self Evalutaion**

Comparing similar products from competitors and existing products gives the author an idea of making the project more unique and distinguish new approaches to solve the problem (**Competitor Analysis**). The author will also self-evaluate as to what data needs to be protected and how from the hackers. Few of the abstractive text summarization tools which are out there are listed and is available at [**APPENDIX B.4**](#_B.4._Self-Evaluation_(Competitor)

In the case of hackers stealing data from the database, **data encryption** can be applied therefore database will only contain the encrypted text data which will be then later decrypted from the decryption key when need, this will be most needed when performing the model retraining.

## **4.5.6 Prototyping**

Table 12: Prototyping findings (*Self-Composed*)

|  |  |
| --- | --- |
| **Criteria** | **Discussion of findings** |
| In-order to look into the feasibility of continuing the project research a prototype was planned to be worked on. | During the iterative prototype process, the author encountered several requirements and obstacles, including the challenge of finding a suitable dataset with desired metadata in the movie domain. After intensive evaluation, the author discovered a large dataset with 8 million entries, but preprocessing it was difficult due to its size and noisy text. To automate the hyperparameter search, the author experimented with a framework called "Optuna." The system will be retrained using new data from the domain user, and the author plans to study at least three top-tier transformer designs to choose the best one. |

## **4.5.6 Summary of findings**

Table 13: Summary of findings (*Self-Composed*)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Id** | **Finding** | **LR** | **Survey** | **Self-Evaluation** | **Interview** | **Brainstorming** | **Prototyping** |
| 1 | The proposed system would benefit businesses (domain specific users) and general users (not domain specific) |  | ✓ |  |  | ✓ |  |
| 2 | For the movie domain the limit of abstractive text summarization can be further pushed using optimized transformers to increase performance this being the existing limitation | ✓ |  |  | ✓ | ✓ |  |
| 3 | It’s clear that customer/user reviews are valued and reviewed mostly by a vast majority of the audience before they consume or use any product or service (applies to any domain) | ✓ | ✓ |  | ✓ | ✓ |  |
| 4 | It’s clear that users spend lot of time review long reviews and they would like it being short to save time and make quicker decisions. | ✓ | ✓ |  |  | ✓ |  |
| 5 | Hyperparameter tuning is one way to increase the performance of the system and it can be done both manually or by automated tools like Raytune, Optuna etc.… | ✓ |  |  | ✓ |  | ✓ |
| 6 | Data preprocessing for the domain of Movies and Generalization is requires a lot of effort since the datasets are mostly raw data difficult to find specially in the case of movie reviews (with expected metadata) | ✓ |  |  |  |  | ✓ |
| 7 | Additional features such as sentimental and sentimental score of the review summary is mostly expected from the users. |  | ✓ |  |  |  |  |
| 8 | Creating a hybrid transformer model to further increase the performance is a suggested improved. |  |  |  | ✓ | ✓ |  |
| 9 | It’s clear on what are the suitable evaluation metrics to be used for abstractive text summarization. | ✓ |  |  | ✓ |  |  |
| 10 | It’s clear on what the top tier transformer architecture that could be explored. | ✓ |  |  | ✓ |  |  |
| 11 | Making use of larger new data for retraining for a specific domain, from companies/businesses who uses data which are of the same domain. (e.g.: - 50 different restaurants data can be combined for retraining give that the domain is “Restaurants”) |  |  |  | ✓ | ✓ |  |
| 12 | Making use of data encryption to protect the data from hackers breaking into the database to steal data. |  |  | ✓ |  | ✓ |  |

# **4.6 Context Diagram**

The boundaries and interactions of the system should be established before development. The graphic below shows how the system is situated.



Figure 7: Context diagram (*Self-Composed*)

# **4.7 Use case Diagram**



Figure 8: Use case diagram (*Self-Composed*)

# **4.8 Use case Descriptions**

Usecase diagrams with the highest importance are given below, the rest of the Usecase descriptors are available at [**APPENDIX C.4**](#_C.4._Use_case).

Table 8: Use case description UC:07 (*Self-Composed*)

|  |  |  |
| --- | --- | --- |
| Use Case Name | View Summary | |
| Use Case Id | UC:07 | |
| Description | Displays a summarized version of the uploaded review text from the domain user’s end. | |
| Primary Actor | General User, Domain Specific User | |
| Pre-Conditions | The text review data must go through specific text preparation techniques before the summary can be produced. | |
| Extended use cases | None | |
| Included use cases | UC10, UC02 | |
| Trigger | A user selects to summarize a given customer/user review text. | |
| Main flow | **Actor** | **System** |
| 1. The user enters the review text on the text field from the GUI. 2. Clicks on “Generate Summary” from the GUI | 1. The system does the data preprocessing for the input review text. 2. Loads the generalized transformer model. 3. Generates the summary using the model. 4. **(If Domain Specific User)** stores the input review and summary into the database. 5. Returns the summary response back to the GUI |
| Alternative flows | None | |
| Expectational flows | Displays an error message if the network request fails (server is down, or internet issues from client). | |
| Post Conditions | Success end condition: The user is presented with the summarized review text. | |

Table 9: Use case description UC:03 (*Self-Composed*)

|  |  |  |
| --- | --- | --- |
| Use Case Name | Retrain Model | |
| Use Case Id | UC:03 | |
| Description | Performs model retraining with the new data from the database, to find the new best set of hyperparameters. | |
| Primary Actor | Domain Specific User | |
| Pre-Conditions | The actor should be a Domain Specific User and have an account created. | |
| Extended use cases | None | |
| Included use cases | UC05, UC06, UC07 | |
| Trigger | The Domain Specific User clicks on the “Perform model retraining” button | |
| Main flow | **Actor** | **System** |
| 1. Domain Specific logs into their account 2. Clicks on “Perform model retraining” from the GUI | 1. The system pulls all the data with respect to the user id from the database. 2. Combines data of the common domains (only if user consent is given to use their data) 3. Finds new set of hyperparameters for the model with respect to new data. 4. Trains the model using the new hyperparameters. 5. Saves the model with the user Id 6. Updates the status in the database if succeed/fails |
| Alternative flows | None | |
| Expectational flows | Displays an error message if the network request fails (server is down, or internet issues from client). | |
| Post Conditions | Success end condition: The user will be able to see the recent status of the model if the retraining is successful or failed | |

# **4.9 Requirements**

## **4.9.1 Functional Requirements**

Based on the significance of the system demands, the ‘MoSCoW’ approach was utilized to identify their priority levels. The details related to the priority levels are detailed at [**APPENDIX B.6**](#_B.5._Functional_requirements).

The Usecase description along its mapping id is also listed at [**APPENDIX B.7**](#_B.5._Functional_requirements).

Table 15: Functional requirements (*Self-Composed*)

|  |  |  |  |
| --- | --- | --- | --- |
| **FR ID** | **Requirement** | **Priority Level** | **Use Case** |
| FR1 | Both general and domain specific users must be able to enter a review text from the GUI considering as the starting point of the summary generation. | M | UC01 |
| FR2 | Only Domain Specific Users should be able to sign up and create an account after entering the necessary details required | S | UC02 |
| FR3 | The system could allow the ability to update the account details of the domain user after creating the account | C | UC02 |
| FR4 | The system must undergo model retraining with the new data stored in the database for the specific domain user, when its triggered from the GUI with the user’s consent. | M | UC03 |
| FR5 | The system could be able to perform model retraining automatically during off peak hours every day. | C | UC03 |
| FR6 | The system must be able to find the new set of best hyperparameters with the usage of the new data. | M | UC04 |
| FR7 | The system must be able to able to retrain the model with the new best hyperparameters and create the model | M | UC05 |
| FR8 | The system must be able to pull the new data from the database to recreate the new dataset for retraining. | M | UC06 |
| FR9 | The system should be able to combine all the data from a common group of domains when creating the dataset only given that the consent is approved to use their data | C | UC06 |
| FR10 | The system must be able to process the review text and display the summary output on the GUI | M | UC07 |
| FR11 | The system must be able to use the latest trained model to generate the summary for the review text | M | UC08 |
| FR12 | The system could also find the sentiment of the generated summary if its positive or negative and return the result. | C | UC08 |
| FR13 | The system could make use of a hybrid model for the text summarization. | C | UC08 |
| FR14 | The system must store the entered user review and generated summary to be stored in the database for retraining purposes. | M | UC09 |
| FR15 | The system should encrypt the data when saving into the database (both the review and summary) | S | UC09 |
| FR14 | The system could allow the domain users to delete the reviews from the database. | C | UC10 |

## **4.9.2 Non-Functional Requirements**

The non-functional requirements are prioritized into two level of which are “Important” and “Desirable”

Table 16: Non-functional requirements (*Self-Composed*)

|  |  |  |  |
| --- | --- | --- | --- |
| **NFR ID** | **Requirement** | **Specification** | **Priority Level** |
| NFR1 | The system needs to be simple enough for non-technical individuals to utilize without much effort. | Usability | Important |
| NFR2 | Meaningful error messages should be displayed if anything goes wrong | Usability | Desirable |
| NFR3 | Summary generation should be done within 3000ms | Performance | Important |
| NFR4 | Following coding standards and best practices | Maintainability | Important |
| NFR5 | Any domain users are able to use the application and model performance will adapt with respect to the domain | Generalization | Important |
| NFR6 | The system should protect against data corruption by attackers, and testing can ensure this. | Security | Desirable |
| NFR7 | The prototype can be used by several domains and multiple businesses under a single domain, then the system may have to support many concurrent user-requests. | Scalability | Desirable |

# **4.10 Chapter summary**

To illustrate how the system interacts with society and the system stakeholders in this chapter, a Rich Picture Diagram was developed. The Saunder's Onion model, which took the influence flow from each stakeholder into account, was used to depict the stakeholders. Requirement gathering techniques were utilized to obtain all the essential data and the opinions of possible system stakeholders. Finally, the use cases, functional requirements, and non-functional requirements for the system were specified using the knowledge gathered from the requirement elicitation methodologies.

# **CHAPTER 05. SOCIAL, LEGAL, ETHICAL & PROFESSIONAL ISSUES**

# **5.1 Chapter overview**

In this chapter, the social, legal, ethical, and professional concerns that emerged during the course of this project are delineated, along with the measures that were implemented to address them.

# **5.2 SLEP issues and mitigation**

The table presented below provides a detailed analysis of the identified social, legal, ethical, and professional (SLEP) issues, as well as the corresponding mitigation strategies that were employed.

Table 17: SLEP issues & Mitigation

|  |  |
| --- | --- |
| **Social** | **Legal** |
| * The dissertation refrains from mentioning the names of the interviewees. * During the conducted survey, no personal details were collected from the respondents to ensure their anonymity. The respondents were kept anonymous throughout the survey process. * The information obtained during the research was not saved or stored elsewhere, ensuring that the data remained secure and confidential. | * All the datasets used in the research are freely available for the public to access. * The tools and technologies utilized in the research are either open source or provided to students free of charge, thereby enabling wider access and use by researchers and students. * The codebase utilized in the research is available on GitHub under the open-source MIT license, allowing others to access and use it freely. |
| **Ethical** | **Professional** |
| * The participants were provided with clear information regarding the intended use of the data that was collected from them. * The work presented in this report is entirely original and has not been reproduced elsewhere. All the ideas taken from external sources have been appropriately cited, and credit has been given where it is due. | * Throughout the project, the team made every effort to adhere to professional standards to ensure the quality and integrity of the work. * The responses and evaluations received during the research were not tampered with, and any limitations encountered during the study were clearly stated and acknowledged. |

# **5.3 Chapter summary**

In this chapter, the author has expounded upon the social, legal, ethical, and professional issues that were encountered during the research process, and has outlined the measures taken to address and resolve them.

# **CHAPTER 06. DESIGN**

# **6.1 Chapter overview**

This chapter discusses the design decisions made to establish an appropriate architecture for implementation based on the requirements obtained. High-level design, low-level design, design diagrams, and UI wireframes have been used to describe how the design goals are meant to be achieved while demonstrating the reasoning for selected design decisions.

# **6.2 Design goals**

The design goals that should be achieved by the system are specified in the table below.

Table 5: Design goals of the proposed system (*Self-Composed*)

|  |  |
| --- | --- |
| **Design Goal** | **Description** |
| Performance | To find the new set of hyperparameters with the new data, model retraining requires a significant amount of time. As a result, the newly created dataset (with unseen data) should be accurately made, and it is best if it takes the least amount of time to query the data from various businesses within the same domain to create the dataset. Moreover, other core functionalities should be designed effectively to increase overall performance. |
| Correctness | The correctness & quality of the output should be of the highest possible level utilizing the optimized transformer architecture. Since several approaches are considered in order to get the optimized solution the expected output should of the best possible form. |
| Usability | The system's usability must be straightforward for users of all levels of knowledge because its primary function is to summarize review text for any domain, including movies and general users. |
| Adaptability | Adopting new features or components need to be a simple procedure. The system shouldn't be broken if a component is added or removed, and it shouldn't be affected overall. |
| Scalability | In a production environment, the system may need to accommodate a large number of concurrent user requests. This should be manageable by the backend. The system should be easily expandable to accommodate new data. |

# **6.3 High Level Design**

## **6.3.1 Architecture Diagram**

The system has a three-tier architecture that separates the data, logic, and presentation levels. The research contribution of the system includes generalization and domain-specific adaptive hyperparameter tuning and data preprocessing.



Figure 9: Three-tiered architecture (*Self-Composed*)

## **6.3.2 Discussion of tiers of the architecture**

**Data Tier**

1. Model Storage - The text summarization models which will be used for both generalized text summarization and domain specific text summarization will be stored here.
2. Generalized Model – The model which will be used by general users to generated review summarized, this model will be hyperparameter tuned for genialized purpose.
3. Domain Specific Model – The model will be used by domain specific users for review summarization, this model will be replaced whenever the model retraining is triggered from the domain user.
4. Dataset Storage – The data which is required for model training will be available.
5. Generalized Dataset – The data which is used for creating the generalized model will be stored for retraining when it comes to domain specific model retraining.
6. New Review Data – The data stored here are from the domain users when they use the application, the data will be storage and used for retraining along with the generalized dataset.
7. User Profile Data Storage – The metadata data related to the domain specific user when creating the user profile will be stored, for updating and profile deletion.

**Logic Tier**

1. User Profile Creation – Allowing to create unique user profiles for each domain user, main purpose comes when working with model retraining to figure out the data to be used.
2. API Proxy – Interface which allows the frontend to communicate with the backend services via HTTP calls/ request.
3. Data Preprocessors – The text data that will be used as input for the text summarizer must be cleaned using the preprocessing code.
4. NLP Text parser – Responsible for cleaning the input text review when received from the API endpoint.
5. Models – The model which will be responsible in generating the summary from the input review and find the sentiment of the summary generated.
6. Generalized Summarization Transformer – This is the summarization model which will be used, an adaptive model depending on the domain and type of user interacting with the model with optimized hyperparameters.
7. Sentiment Analysis Transformer – This model will be used to classify the generated summary into positive or negative sentiment.
8. Data Encryption – Data encryption is in charge of data protection/safety, keeping domain data extremely secure and leaving it useless even if it is stolen.
9. Model Retraining – Responsible for retraining the model with new data and finding new set of hyperparameters.
10. Dataset Recreation – Responsible for recreating the dataset with new data which has been given as input from the domain users
11. Hyperparameter tuning – Responsible for finding the new best set of hyperparameters using the new data.
12. Model Training – Responsible for training the new model with the new set of hyperparameters found.

**Presentation Tier**

1. User Profile Creation Wizard – The UI that presents the user to create a new profile if they are planning to use the software for their domain business, or a general user to skip the sign up if only a generalized summary is required.
2. User Input Wizard for Reviews – The UI that will request the user to input the review which needs to be summarized.
3. Summarization Feed UI – The UI that displayed the summarized text for the input review.
4. Hyperparameter tuning UI – The UI that triggers model retraining when the domain user performs an action on it.

# **6.4 System Design**

## **6.4.1 Choice of Design Paradigm**

The main reason behind the author going ahead with **SSADM (Structured Systems Analysis and Design Method)** over **OOAD (Object-Oriented Analysis and Design)** to build the protype was due to the ease of ability to extend the system features when it comes to future developments of the system. Given below are the other factors as to why the choice of SSADM was considered:

* Object Oriented approaches will not make a greater benefit since the main core project research lies towards Data Science.
* Ability to demonstrate the MVP (Minimum Viable Product) prototype implementation for the research application is more convenient.
* More time efficient when concerned with the time constraint of having to complete the documentation research along with the project implementation.

# **6.5 Design diagrams**

## **6.5.1 Data Flow diagrams**

In order to show the relationships between components and provide a clearer understanding of how data flows across the whole system, the context diagram's components have been extensively broken down in the diagram below, which was detailed in the SRS previously.

### **6.5.1.1 Level 01 Data Flow diagram**



Figure 6: Data flow diagram - level 01 (*Self-Composed*)

### **6.5.1.2 Level 02 Data Flow diagram**

The level 02 data flow diagram given below is a further breakdown of the core hyperparameter tuning and model retraining proposed in the level 01 data flow diagram.



Figure 11: Data flow diagram - level 02 (*Self-Composed*)

## **6.5.4 UI design**

Given the specifications acquired from the target audience, the author chose a web application for the simulation of the proof-of-concept application. A wireframe design was created to depict the key user interface aspects in the system and is available in [**APPENDIX C.2**](#_C.2._UI_wireframes)

## **6.5.5 System process activity diagram**

The flowchart given below represents the algorithm’s flow and the decision structures which explains the flow of the system which is initially expected requirement.



Figure 12: System process flow chart (*Self-Composed*)

# **6.6 Chapter summary**

This chapter provides an in-depth examination of the project's design, including its architectural features and explains the core flow via data flow diagrams. The chapter concludes with a preview of the user interface wireframes that will be utilized to facilitate interaction between the end-user and the system.

# **CHAPTER 07. IMPLEMENTATION**

# **7.1 Chapter overview**

This chapter will provide a thorough overview of the technologies, supporting tools, and languages utilized for the project development, as well as the fundamental implementation of the research prototype.

# **7.2 Technology selection**

## **7.2.1 Technology stack**

The technologies utilized to implement the prototype at each tier are given below.



Figure 13: Technology stack (*Self-Composed*)

In preference to macOS and Linux, **Windows** will be the operating system used for project development and documentation. This is due to a wider variety of software available, which ensures that it has more industry-standard tools than Linux and macOS, along with better compatibility and familiarity, which make things simpler to use and manage.

## **7.2.2 Data selection**

Given that the project relies heavily on data science, it is essential to use data from trustworthy sources to train the model. This ensures that the data is accurate and leads to the development of a more accurate model for general text summarization.

The goal of the project was to develop an adaptive generalized text summarization model, so a generalized dataset for text summarization was necessary to establish the base model. **TensorFlow datasets**, being a reputable source of data, offered multiple options for this dataset.

The table below shows the datasets which have been used by previous researchers, therefore this can experiment for the protype development.

Table 6: Dataset sources (*Self-Composed*)

|  |  |
| --- | --- |
| **Dataset** | **Source** |
| CNN Dailymail | TensorFlow Datasets |
| Gigaword | TensorFlow Datasets |
| Xsum | TensorFlow Datasets |

During the training process, all three of these datasets (CNN Dailymail, Gigaword, and Xsum) were utilized with various transformer architectures to determine which dataset resulted in the best evaluation metrics. Of the three datasets, Xsum performed the best, so it was selected as the final dataset for the project.

## **7.2.3 Programming Language Selection**

In this study, we employed the programming language **Python** for the implementation of our Machine Learning models and Backend APIs. Python is a widely-used language known for its readability, simplicity and versatility, making it an ideal choice for our research project. This language has a broad range of use cases including web development, data analysis, scientific computing and machine learning. Additionally, Python has a large and active community, providing ample resources and support. Furthermore, the availability of various libraries and frameworks such as NumPy, pandas, and TensorFlow, made Python a powerful tool for our data science and machine learning tasks.

**TypeScript** (it’s a superset of JavsScript) was chosen for the frontend development in order to display dynamic content and create a highly interactive and engaging user experience.

## **7.2.4 Development Framework Selection**

The author has chosen several development frameworks for the project covering all areas, the table given below describes the purpose of choosing each framework and whats it used for in the project.

Table 7: Development framework utilized (*Self-Composed*)

|  |  |
| --- | --- |
| **Framework** | **Reason for choosing** |
| React | ReactJS provides reusable components for efficient application development, and its open-source nature and strong community support enable continuous developments and learning tools, making it a handy solution for developers. |
| Ant Design | Ant Design is a popular React UI framework that offers a large selection of pre-built components, encourages consistency and usability, and enables for style customization using CSS-in-JS. It also reduces build time by using tree-shaking compatibility. Overall, it provides a complete and effective frontend development solution. |
| Flask | Flask is a Python micro web framework that is lightweight, easy to learn, and provides for flexibility in developing application structures. It is useful for developing backend APIs since it provides a straightforward approach to manage routing and request processing, as well as a built-in development server and different extensions that can be used to extend an API's capabilities. |
| Optuna | Optuna is a Python open-source framework for hyperparameter optimization that is simple to use, efficient, and has built-in parallelization support. It also offers built-in support for popular machine learning libraries, as well as automated early halting and distributed parallel optimization. It is a robust and adaptable library that can aid in the improvement of machine learning model performance. |
| PyTorch | PyTorch is a Python open-source machine learning framework that is built on Torch library and makes use of GPU capability. Because of its straightforward and easy-to-use API, vast selection of pre-built neural network layers and modules, powerful features such as dynamic computation graphs and automated differentiation, and strong community support, it's a solid choice for developing machine learning models. It is widely used in business and academia for machine learning model research and development. |

The data science core employs transformer models from Hugging Face, which have been fine-tuned with the datasets used in this research project. The purpose of retraining the model is to experiment with various hyperparameter changes.

## **7.2.5 Libraries Utilized**

Table 8: Libraries used with reasonings (*Self-Composed*)

|  |  |
| --- | --- |
| **Library** | **Reasoning for selection** |
| Firebase | Used for providing backend services for mobile and web application development. |
| Axios | Used for handling HTTP requests in JavaScript. |
| Redux | Used to control the state of JavaScript applications in a predictable manner by the use of actions, reducers, and a central store. |
| Hugging face Transformers | Hugging Face transformers library is a state-of-the-art natural language processing library that provides pre-trained transformer models and tools for fine-tuning them on specific tasks. |
| NLTK | NLTK is a library for natural language processing that provides tools for tasks such as tokenization, stemming, and part-of-speech tagging, as well as a wide range of corpora and resources for training and evaluating language models. |
| Rouge | A library for evaluating the quality of text summaries, it is used to compare an automatically generated summary or a peer summary to one or multiple reference summaries. |
| Pandas | Pandas is a library for data manipulation and analysis, it provides data structures and data analysis tools for handling and manipulating numerical tables and time series data, it is widely used for data preprocessing and data cleaning tasks in data science. |
| NumPy | NumPy is a library for scientific computing with Python, it provides support for large, multi-dimensional arrays and matrices of numerical data, as well as a large collection of mathematical functions to operate on these arrays |
| Matplotlib & Seaborn | Used for creating static, animated, and interactive visualizations in Python |
| Gramformer | Used for generating text using GPT-3 model, it's developed by Hugging Face. It provides an easy to use API that allows developers to fine-tune GPT-3 models on their own data and use them for text generation, it supports for various tasks such as text completion, text generation, and text classification. |
| Flask | Used for creating web APIs using Python to communicate with the transformer model and handling HTTP requests. |

## **7.2.6 IDE’s Utilized**

Table 9: IDE’s used along with justifications (*Self-Composed*)

|  |  |
| --- | --- |
| **IDE** | **Justification for selection** |
| VSCode | Best known for its adaptability, usefulness, and performance, it offers a wide range of capabilities, such as debugging, Git integration, syntax highlighting, and extensions to personalize the environment. |
| Google Colab | Due to its connection with Google Drive and availability of free GPUs, it’s helpful for developing machine learning models via a cloud environment. |
| Jupyter Notebook | Due to their interactive and readable format, making it ideal for local experimentation, documentation and collaboration. |

## **7.2.7 Summary of Technology Selection**

Table 10: Summary of Technology selection (*Self-Composed*)

|  |  |
| --- | --- |
| **Component** | **Tools** |
| Programming Languages | TypeScript, Python |
| Development Framework | Flask, PyTorch, Optuna |
| UI Framework | Ant Design, React |
| Libraries | NLTK, Rouge, React, Pandas, Gramformer, Matplotlib & Seaborn, Axios, Transformers (from hugging face) |
| IDE – Research | Google Colab, Jupyter Notebook |
| IDE – Product | VSCode |
| Version Control | Git, GitHub |
| Data storage | Firebase |

# **7.3 Implementation of Core Functionalities**

The project's core functionalities include the experiments of top-tier transformer architectures to determine the optimal one, applying data preprocessing steps, automating hyperparameter searching, retraining the model with new data fetched from the database and new hyperparameters, and having the model be able to summarize reviews from both domain users and general users.

## **7.3.1 Automated Hyperparameter Search & Model Training**

The author did a research on different approaches to automate the hyperparameter searching, because manual hyperparameter tuning is total waste of time. Multiple hyperparameter tuning frameworks were available, however Optuna was chosen due to its flexibility and ease of use.



Figure 14: Hyperparameter Range Initialization (*Self-Composed*)

The code snippet above illustrates how the hyperparameters are initialized with a group of values, some of which are within a range based on the setting of the min and max parameters. In order to determine the optimal parameter values from the initialized range, these parameters will be utilized during hyperparameter search training. If no range is specified, the default will start at zero.



Figure 11: Hyperparameter search using Optuna (*Self-Composed*)

The above snippet shows how the Optuna framework is integrated with the model training code to perform automated hyperparameter search. The main performance contributing parameters are considered for the hyperparameter searching this includes learning rate, weight decay, num of training epochs, warmup ratio, batch size.



Figure 12: Hyperparameter results and training arguments (*Self-Composed*)

The above snippet demonstrates how to result of the hyperparameter search is used within the training arguments for model training.



Figure 13: Model training (*Self-Composed*)

The above code snippet is the model training initiation with the optimal hyperparameters.

## **7.3.2 Model Usage General & Domain Specific Users**



Figure 14: General user review text summarization (*Self-Composed*)

The above code snippet is an API endpoint which handles text (review) summarization for the general users where they don’t need to create and account or have specialized model assigned to them instead the general model is utilized.



Figure 15: Assigning a specific model for the new domain user (*Self-Composed*)

The above code snippet describes an API for assigning a copy of the generalized model for the user id of the domain (given that the domain user signed up for the application), the reason for creating a copy is for retraining purposes with new data.



Figure 16: Domain Specific text review summarization (*Self-Composed*)

The above code snippet describes how the newly assigned domain specific model is used to generate the summary and store the input and outputs into the database along with returning the sentiment of the summary with the sentiment score. The sentiment analysis is done using a pretrained transformer directly from hugging face API.

## **7.3.3 Model Retraining**



Figure 17: Fetching related data for model retraining (*Self-Composed*)

The code snippet above describes the necessary data fetched from the database to create the new dataset for model retraining, once the new dataset is created it is passed through a function to perform hyperparameter tuning and then retrain the model. Once completed retraining, the old model will be replaced with the new model in the folder path location.

## **7.3.4 Data Preprocessing**

The raw dataset was contaminated with a lot of noise, numerous data preprocessing steps were required to clean the data before model training. The related preprocessing scripts can be found at [**APPENDIX D.1**](#_D.1._Data_Preprocessing).

# **7.4 User interface**

Screenshots of the final GUI are placed under [**APPENDIX E.7**](#_E.7._User_interface).

# **7.5 Chapter summary**

The chapter discusses the tools, technology, and languages utilized to create the research prototype. The fundamental functionality is covered, along with insights and samples of code for the implemented algorithms, moreover the testing and evaluation related code for the models is discussed.

# **CHAPTER 08. TESTING**

# **8.1 Chapter Overview**

After achieving an acceptable level of implementation, it is imperative to subject the system to rigorous testing to ascertain that its functionalities operate as intended. This chapter entails conducting comprehensive testing on both the system and the utilized model. The testing methodologies employed encompass functional, non-functional, integration, and model testing, all aimed at providing an extensive evaluation of the system's performance.

# **8.2 Testing objectives & goals**

The primary objective of testing is to verify that the system functions in the expected manner. Achieving this objective requires meeting several testing goals.:

* Confirm the model's performance is optimized to its fullest potential.
* Ensure that the implemented functionalities are in line with the "Must have" and "Should have" criteria of the MoSCoW technique.
* Identify any necessary bug fixes or improvements that need to be applied to the application.
* Ascertain if the critical non-functional requirements have been satisfied.
* Perform baseline benchmarking to establish a standard for comparing the system's future performance.

# **8.3 Testing criteria**

Before proceeding with the testing phase, a specific set of standards was established to assess the system using two different methods.

* **Functional Quality** – This involves examining the system's developmental traits and technical specifications to determine its conformity to the designated design based on functional requirements.
* **Structural Quality** – This evaluates the system's non-functional requirements while simultaneously ensuring that it satisfies the functional requirements' performance criteria.

# **8.4 Model testing & evaluation**

## **8.4.1 Model testing**

Three transformer architectures, considered to be the most prominent, were employed to train the datasets, and subsequently, conducted testing on all of them. The figures presented below display the validation accuracy and loss for each of the three models, facilitating the selection of the model that performed the best.

|  |  |
| --- | --- |
| Figure 20: Validation Accuracy by number of epochs – bart model (*Self-Composed*) | Figure 21: Validation Loss by number of epochs – bart model (*Self-Composed*) |

The figures shown above depict the outcomes of testing the BART model, where all the rouge scores exhibit an upward trend as the number of epochs increases.

These results were obtained after conducting hyperparameter tuning, which yielded a higher score compared to the benchmark scores reported in earlier research.

Similarly, there is a considerable decline in the validation loss as the number of epochs increases, indicating that the model is enhancing its capability to generate precise predictions.

|  |  |
| --- | --- |
| Figure 22: Validation Accuracy by number of epochs – t5 model (*Self-Composed*) | Figure 23: Validation Loss by number of epochs – t5 model (*Self-Composed*) |

The presented figures demonstrate the testing results of the T5 model, where all the rouge scores exhibit an exponential increase and eventually plateau as the number of epochs increases. Similarly, the validation loss also shows an exponential decrease with respect to the number of epochs. Nevertheless, the overall results indicate that the BART model performs slightly better than the T5 model.

|  |  |
| --- | --- |
| Figure 24: Validation Accuracy by number of epochs – Pegasus model (*Self-Composed*) | Figure 25: Validation Loss by number of epochs – Pegasus model (*Self-Composed*) |

The illustration above depicts the testing outcomes of the Pegasus model, indicating a poor performance despite identifying the optimal hyperparameters for the given training dataset. The rouge scores remain consistently low, and the validation loss graph depicts no variation due to the lack of improvement in accuracy.

## **8.4.2 Model evaluation**

The evaluation metrics for the models were computed based on the recommended criteria from the literature review and the author's proficiency in evaluating machine learning models. These metrics were then presented in Chapter 2, under the "Evaluation" topic.

According to previous research conducted by (Steinberger and Jezek, 2009), among the primary scoring methods of ROUGE and BLEU, ROUGE and its metric versions are regarded as the most appropriate for achieving optimal results.

Table 26: Model Evaluation Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Rouge1** | **Rouge2** | **RougeL** | **RougeLSum** |
| Bart | 80.78 | 79.42 | 80.80 | 80.83 |
| T5 | 80.75 | 79.76 | 80.77 | 80.79 |
| Pegasus | 10.07 | 1.21 | 8.34 | 8.33 |

According to the evaluation provided, it is evident that the Bart transformer model performs better than the other two models, while Pegasus has the poorest performance.

There could be multiple explanations for Pegasus' inadequate performance in this domain, even if the model has undergone hyperparameter optimization and the dataset has been suitably preprocessed. Pegasus and Bart have distinct model architectures, with Pegasus being a transformer-based model that employs an encoder-decoder framework using Seq2Seq, while Bart uses a denoising autoencoder framework. The contrasting architectures can impact how the models comprehend and portray the text data, leading to the possibility that one architecture may be more suitable for a specific dataset than the other.

# **8.5 Benchmarking**

Previously, in Chapter 2 under the “Benchmarking” topic it discussed the benchmarking results of training transformers with generalized datasets for abstractive text summarization. The table given below is a benchmarking comparison with the previous researched results and the authors results.

Table 27: Benchmarking results

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Year** | **Type of model** | **Rouge1** | **Rouge2** | **RougeL** | **Is optimized** | **Dataset** |
| 2019 | Transformer | 36.73 | 14.93 | 29.66 | No | Xsum |
| 2019 | Bart | 45.14 | 22.27 | 37.25 | No | Xsum |
| 2020 | RoBERTa | 45.42 | 22.13 | 36.92 | No | Xsum |
| 2023 | Bart | 80.78 | 79.42 | 80.80 | Yes | Xsum |

The presented table indicates the benchmarking outcomes of transformer models utilized by previous researchers between 2019 and 2020 on the same dataset (Xsum) utilized by the author. The final result displayed in the table (Year 2023) represents the author's ultimate evaluation result, which demonstrates almost a two-fold improvement in performance compared to earlier researchers who did not optimize their models.

# **8.6 Functional testing**

The system was assessed to determine if it complies with the functional requirements outlined in **Chapter 4** through the use of functional testing. A breakdown of the functional testing that was conducted can be found in **APPENDIX F.1**.

// remember to add the content to appendix

# **8.7 Module & integration testing**

The high-level architecture diagram depicted in **Chapter 6** illustrated that the system's logic was divided into modules. To ensure that each module operates as intended, they underwent testing.

// table comes here

Table 25: Module & integration testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Module** | **Input** | **Expected result** | **Actual result** | **Status** |
| Data fetcher | Triggered periodically | Fetch & update datasets. | Datasets scraped and stored into database. | Passed |

# **8.8 Non-functional testing**

The system's non-functional requirements were evaluated to assess how well they correspond with the non-functional requirements and design objectives specified in **Chapter 4** and **Chapter 5**, respectively. **APPENDIX F.2** details the specific breakdown of the non-functional testing that was executed.

# **8.9 Limitations of the testing process**

Pending…. We have to fill this part at last

# **8.10 Chapter summary**

In the beginning of this chapter, the aims of the testing process and the standards for conducting it were introduced. The evaluation of the models in use was performed by testing the main research component, and functional, non-functional, and integration testing were utilized to assess the system. Ultimately, any constraints of this methodology were identified.

# **CHAPTER 09. EVALUATION**

# **9.1 Chapter Overview**

Once the prototype design was successfully implemented and optimized through multiple testing combinations to achieve maximum performance, the system was assessed in accordance with the requirements outlined in the SRS chapter. This particular chapter is devoted to the project's evaluation, which includes self-evaluation by the author and evaluations by technical, domain, and industry experts.

# **9.2 Evaluation Methodology & Approach**

Pending….

# **9.3 Evaluation Criteria**

Pending….

Table 26: Evaluation criteria

|  |  |  |
| --- | --- | --- |
| **CR ID** | **Criterion** | **Purpose** |
| CR1 |  |  |
| CR2 |  |  |
| CR3 |  |  |
| CR4 |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# **9.4 Self-Evaluation**

Pending….

# **9.5 Selection of Evaluators**

Pending….

# **9.6 Evaluation Results & Expert Opinions**

Pending….

# **9.7 Limitations of Evaluation**

Pending….

# **9.8 Evaluation of Functional requirements**

Pending….

# **9.9 Evaluation of non-functional requirements**

Pending….

.

# **9.10 Chapter summary**

Pending….

# **CHAPTER 10. CONCLUSION**

# **10.1 Chapter Overview**

This chapter covers the preliminary conclusion of the research project, including the core functionality of its implementation for the MVP. The chapter will also review the achievements of the project's goals and objectives and the obstacles encountered. Additionally, an outline of the author's prior knowledge and modules of the program which helped to support the project will be documented along with any new knowledge and skills acquired.

# **10.2 Achievement of Research Aim & Objectives**

## **10.2.1 Achievement of Aim**

**“***The aim of this research is to design, develop and evaluate an optimal generalized transformer architecture from a range of popularly used architectures by fine-tuning via hyperparameter optimization, therefore obtaining the recommended architecture's optimum performance.***”**

The core components related to the aim of the research is successfully completed by designing, developing & evaluating a performance adaptive generalized transformer. The core functionality was researched in a way to be automated in order to meet the project requirements. The evaluations for the respective work done is attached in the implementation chapter.

## **10.2.2 Achievement of Objectives**

Appendix G – contains the achievement status related to the research objectives which were mentioned in the Chapter 01. "Completed" is the mark next to tasks that were successfully completed, while "Incomplete" is the mark next to those that weren't.

# **10.3 Utilization of knowledge from the degree**

Table 10.1: Utilization of Knowledge gained from the course

|  |  |
| --- | --- |
| **Module(s)** | **Utilized Knowledge** |
| Machine Learning | Understanding the concept underlying data collection and preprocessing, as well as how to train machine learning models, was extremely helpful in developing the models for this research project. |
| Applied AI | The in-depth understanding of how algorithms interact while building ML models provided an understanding of the theoretical principles. |
| Software Development Group Project | This module served as more of a trial run for the Final Year Project; it provided a basic understanding of how to plan, conduct, and assess the research project, providing students the confidence and knowledge necessary to carry out research in their final year. |
| Object Oriented Programming | The knowledge about creating classes and how objects are important helped to enhance the development side area of knowledge for the project. |
| Python Programming (PP1) | This project has the usage of Flask (Python Programming Language Web Framework), PP1 module helped to get introduced to working with Python. |
| Database Systems | The knowledge and the idea of how queries are used to communicate with the database from the webserver system, helped a lot in order to perform read & write operations. |
| Web Design & Development | The concepts thought from this module was used to build the UI for the prototype and the foundation idea of using HTML, CSS and JS supported a lot to move into working with advanced frameworks like React. |

# **10.4 Use of Existing Skills**

* **Full-Stack Web Development** – Throughout his internship, the author worked on a number of R&D projects at 99x, where he was able to use cutting-edge technologies for a full stack web development project.
* **Machine Learning / Deep Learning** – During the internship, the author worked on many data science-related R&D projects and also used a variety of online learning resources for self-learning and developing machine learning projects.
* **Documentation Writing** – During the internship and while working on the SDGP module report, the author gained expertise in creating project documentation.

# **10.5 Use of New Skills**

* **Text Summarization Systems –** The author has never before worked in an area involving text summarization systems. The author has investigated a number of techniques for dealing with text summarization using publicly accessible online resources, such as material from YouTube, GitHub, Google Colab, and others.
* **Data Preprocessing Techniques –** The author had to learn new various text preprocessing techniques since the project domain lies under text summarization therefore the data after preprocessing has to be meaningful.
* **Hyperparameter Optimization –** Hyperparameter tuning frameworks were explored and experimented to automated the hyperparameter search. Tutorials and tech articles were refereed in order to implement this with in the project.

# **10.6 Achievement of learning outcomes**

Pending

# **10.7 Problems and Challenges faced**

Table 10.2: Mitigations to Problems and Challenges Faced

|  |  |
| --- | --- |
| **Problem/Challenge** | **Mitigation** |
| Significant training time and computing resource limitations. | Transformer-based model training demands a lot of GPU power. To get around this problem, the author trains the models using Google Colab. |
| Limited experts for the domain | The author contacted domain experts via LinkedIn in order to conduct interviews for requirement gathering because there weren't many domain experts that could be contacted in person. |
| Due to frequent power outages, there were battery problems and internet connectivity issues. | The author continued working on the project despite the power outages by working late or in the early morning at co-working spaces. |

# **10.8 Deviations**

The initial goal of the author was to create an optimized solution for movie review summarization using transformers, but after discussions made with supervisors the research gap of the author for the technical contribution being only hyperparameter tuning of transformer felt small in magnitude, therefore the idea of creating a ***performance adaptive generalized solution*** was considered to continue the research implementation on.

After considering the possibilities of implementation this solution, the experts interviewed for the requirement gathering mentioned it to be challenging with the time frame of the project to execute however the author was able to complete the initial core functionalities for the prototype.

# **10.9 Limitations of the research**

* After the core implementation, the author attempted to implement additional performance improvements such as model hybridization, but due to the limited time available, the amount of research that needed to be done in the area of transformer hybridization was significant, which prevented the author from continuing.
* Due to the limitations of GPUs, various other transformer models weren’t explored for abstractive text summarization.

# **10.10 Future enhancements**

* Making use of transformer hybridization to future improve the performance of the text summarization models.
* Since there is a potential that user reviews entered aren't always accurate, it becomes sense to include text paraphrase models for the user reviews.
* Applying key word extraction for the sentiment classification of the review summary, to identify what key words contributed to the sentiment classification, this will help the domain users improve their service.

# **10.11 Achievement of the Contribution to the Body of Knowledge**

The author successfully contributed to the problem domain, which was the movie review summarization along with the deviations made (model generalization), the technical contributions in order to increase the performance of the system was also made and lastly the author made additional contributions to the project bringing the research project to a conclusion.

## **10.11.1 Domain Contributions**

The author as able to address the performance gap listed for movie review summarization, in order for the need of advanced approaches to increase the performance and achieve a better result.

Moreover, generalization approach considered here contributions to various other domains facing the similar problem to be address as a common.

## **10.11.2 Technical Contribution**

Using a top-tier explored transformer model, automating hyperparameter search for every domain, and use the newly exposed data to automate model retraining with the searched optimal set of hyperparameters.

Currently, there are no such approaches taken from the research done by the author, and the author believes that this approach would benefit multiple domains at the same time.

## **10.11.3 Additional Contributions**

1. Research-based Data Preprocessing scripts specifically for text summarization issue domains.
2. Sentiment Analysis on the generated review summary
3. Experimented the model training with multiple datasets, to get the best possible set of evaluation results.

# **10.12 Implementation code**

All related code and documentation material are made available in GitHub by the author at <https://github.com/nazhimkalam/gensum/tree/main/Code>

# **10.13 Concluding remarks**

The conclusion of this study finds that the author was able to design, build, and evaluate an adaptive generalized abstractive text summarization system using optimized transformers and automated hyperparameter tuning and model retraining. The purpose of this chapter was to determine if the author met the goals and objectives of the project, and to examine the role of the author's prior knowledge and academic background in supporting the research. Additionally, the author discusses the new skills they acquired during the project and the challenges and obstacles they encountered, as well as the deviations taken and the limitations of the research. The author also goes in detail about the opportunities for future improvements and also discusses about the contribution to the body of knowledge which include domain, technical and additional contributions made. The author is currently working on publishing a research paper based on their findings.

# **REFERENCES**

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>

Ackerman, D. (2021). “Liquid” machine-learning system adapts to changing conditions. *MIT News | Massachusetts Institute of Technology*. Available from <https://news.mit.edu/2021/machine-learning-adapts-0128> [Accessed 17 October 2022].

Ahmed, NK., Atiya, AF., Gayar, NE. and El-Shishiny, H. An Empirical Comparison of Machine Learning Models for Time Series Forecasting. *Econometric Reviews*. 2010;29(5-6):594–621.

Alonso-Monsalve, S. et al. (2020). Convolution on neural networks for high-frequency trend prediction of cryptocurrency exchange rates using technical indicators. *Expert Systems with Applications*, 149, 113250. Available from <https://doi.org/10.1016/j.eswa.2020.113250> [Accessed 22 October 2022].

Anumasa, S. and Srijith, P.K. (2022). Latent Time Neural Ordinary Differential Equations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36 (6), 6010–6018. Available from <https://doi.org/10.1609/aaai.v36i6.20547> [Accessed 17 October 2022].

Avdeychik, V., & Capozzi, J. (2018). SEC’s Division of Investment Management Voices Concerns Over Registered Funds Investing in Cryptocurrencies and Cryptocurrency Related Products. *Journal of Investment Compliance*, 19(2), 8–12.

Baldimtsi, F., Kiayias, A., & Samari, K. (2017). Watermarking Public-Key Cryptographic Functionalities and Implementations. In Nguyen, P. Q., & Zhou, J. (Eds.). *Information Security*, 173–191. Berlin: Springer.

Bhardwaj, S.P. et al. (2014). An Empirical Investigation of Arima and Garch Models in Agricultural Price Forecasting. *Economic Affairs*, 59 (3), 415. Available from <https://doi.org/10.5958/0976-4666.2014.00009.6> [Accessed 17 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].

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

Boom, V.D. (2021). Want to Buy Crypto? Here’s What to Look for In a Crypto Exchange. *Time, 11 June*. Available from <https://time.com/nextadvisor/investing/cryptocurrency/what-are-cryptocurrency-exchanges> [Accessed 23 October 2022].

Bouktif, S. et al. (2018). Optimal Deep Learning LSTM Model for Electric Load Forecasting using Feature Selection and Genetic Algorithm: Comparison with Machine Learning Approaches †. *Energies*, 11 (7), 1636. Available from <https://doi.org/10.3390/en11071636> [Accessed 28 December 2022].

Bouri, E., Gupta, R., Lau, C. K. M., Roubaud, D., & Wang, S. (2018). Bitcoin and global financial stress: A copula-based approach to dependence and causality in the quantiles. *The Quarterly Review of Economics and Finance*, 69, 297–307.

Buhalis, D. et al. (2019). Technological disruptions in services: lessons from tourism and hospitality. *Journal of Service Management*, 30 (4), 484–506. Available from <https://doi.org/10.1108/JOSM-12-2018-0398> [Accessed 22 October 2022].

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

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].

Cho, K. et al. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. Available from <https://doi.org/10.48550/ARXIV.1406.1078> [Accessed 9 April 2023].

Critien, J.V., Gatt, A. and Ellul, J. (2022). Bitcoin price change and trend prediction through twitter sentiment and data volume. *Financial Innovation*, 8 (1), 45. Available from <https://doi.org/10.1186/s40854-022-00352-7> [Accessed 16 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].

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].

Engle, R.F. (1982). Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation. *Econometrica*, 50 (4), 987. Available from <https://doi.org/10.2307/1912773> [Accessed 28 December 2022].

Fischer, T. and Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270 (2), 654–669. Available from <https://doi.org/10.1016/j.ejor.2017.11.054> [Accessed 14 February 2023].

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].

Fortune Business Insights (2021). Cryptocurrency Market Size, Growth & Trends | Forecast [2028]. *Fortune Business Insights*. Available from <https://www.fortunebusinessinsights.com/industry-reports/cryptocurrency-market-100149> [Accessed 23 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].

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

Gilliland, M. (2014). A naive forecast is not necessarily bad. *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].

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

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].

Hutto, C., & Gilbert, E. (2014). VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of social media Text. *Proceedings of the International AAAI Conference on Web and social media*, 8(1), 216-225

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, 22(4), 679–688

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].

IBM Cloud Team (2021). Python vs. R: What’s the Difference? *IBM*. Available from <https://www.ibm.com/cloud/blog/python-vs-r> [Accessed 18 October 2022].

InterviewBit (2021). Flask Vs Django: Which Python Framework to Choose? *InterviewBit*. Available from <https://www.interviewbit.com/blog/flask-vs-django/> [Accessed 12 December 2022].

Kerr, J. (2018). How Can Legislators Protect Sport from the Integrity Threat Posed by Cryptocurrencies? *The International Sports Law Journal*, 18(1), 79–97.

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].

Kfir, I. (2020). Cryptocurrencies, national security, crime and terrorism. *Comparative Strategy*, 39 (2), 113–127. Available from <https://doi.org/10.1080/01495933.2020.1718983> [Accessed 22 October 2022].

Kim, M. et al. (2019). A Hybrid Neural Network Model for Power Demand Forecasting. *Energies*, 12 (5), 931. Available from <https://doi.org/10.3390/en12050931> [Accessed 16 October 2022].

Kim, Y.B. et al. (2016). Predicting Fluctuations in Cryptocurrency Transactions Based on User Comments and Replies. *PLOS ONE*, 11 (8), e0161197. Available from <https://doi.org/10.1371/journal.pone.0161197> [Accessed 16 October 2022].

Kuan, L. et al. (2017). Short-term electricity load forecasting method based on multilayered self-normalizing GRU network. *2017 IEEE Conference on Energy Internet and Energy System Integration* (EI2). November 2017. Beijing: IEEE, 1–5. Available from <https://doi.org/10.1109/EI2.2017.8245330> [Accessed 17 October 2022].

Kurama, V. (2022). PyTorch vs. TensorFlow: 2022 Deep Learning Comparison | Built In. *Built In*. Available from <https://builtin.com/data-science/pytorch-vs-tensorflow> [Accessed 12 December 2022].

Lapicque, L. 1907. Recherches quantitatives sur l’excitation electrique des nerfs traitee comme une polarization. *Journal de Physiologie et de Pathologie Generalej* 9: 620–635.

Lara-Benítez, P., Carranza-García, M. and Riquelme, J.C. (2021). An Experimental Review on Deep Learning Architectures for Time Series Forecasting. *International Journal of Neural Systems*, 31 (03), 2130001. Available from <https://doi.org/10.1142/S0129065721300011> [Accessed 16 October 2022].

Li, A.W. and Bastos, G.S. (2020). Stock Market Forecasting Using Deep Learning and Technical Analysis: A Systematic Review. *IEEE Access*, 8, 185232–185242. Available from <https://doi.org/10.1109/ACCESS.2020.3030226> [Accessed 16 October 2022].

Li, S. et al. (2019). Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting. Available from <https://doi.org/10.48550/ARXIV.1907.00235> [Accessed 17 October 2022].

Li, X. et al. (2020). Scalable Gradients for Stochastic Differential Equations. Available from <http://arxiv.org/abs/2001.01328> [Accessed 18 January 2023].

Lim, B. and Zohren, S. (2020). Time Series Forecasting With Deep Learning: A Survey. Available from <https://doi.org/10.1098/rsta.2020.0209> [Accessed 21 October 2022].

Lim, B. et al. (2019). Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting. Available from <https://doi.org/10.48550/ARXIV.1912.09363> [Accessed 17 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].

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].

Miller, P. (2016). Chapter 1—The Cryptocurrency Enigma. In Sammons, J. (Ed.), *Digital Forensics*, 1–25. Syngress. <https://doi.org/10.1016/B978-0-12-804526-8.00001-0>

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].

Mudassir, M. et al. (2020). Time-series forecasting of Bitcoin prices using high-dimensional features: a machine learning approach. *Neural Computing and Applications*. Available from <https://doi.org/10.1007/s00521-020-05129-6> [Accessed 3 December 2022].

Nica, O., Piotrowska, K., & Schenk-Hoppé, K. R. (2017). Cryptocurrencies: Economic Benefits and Risks. SSRN Scholarly Paper ID 3059856.

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].

Pan, C. et al. (2019). Very Short-Term Solar Generation Forecasting Based on LSTM with Temporal Attention Mechanism. *2019 IEEE 5th International Conference on Computer and Communications (ICCC)*. December 2019. Chengdu, China: IEEE, 267–271. Available from <https://doi.org/10.1109/ICCC47050.2019.9064298> [Accessed 14 February 2023].

Pant, D.R. et al. (2018). Recurrent Neural Network Based Bitcoin Price Prediction by Twitter Sentiment Analysis. 2018 IEEE 3rd International Conference on Computing, *Communication and Security (ICCCS)*. October 2018. Kathmandu: IEEE, 128–132. Available from <https://doi.org/10.1109/CCCS.2018.8586824> [Accessed 16 October 2022].

Patadiya, J. (2021). Angular vs React | Angular vs Vue | React vs Vue - Know the Difference. *Radixweb*. Available from <https://radixweb.com/blog/angular-vs-react-vs-vue> [Accessed 12 December 2022].

Peluchetti, S. and Favaro, S. (2019). Infinitely deep neural networks as diffusion processes. Available from <https://doi.org/10.48550/ARXIV.1905.11065> [Accessed 2 December 2022].

Picasso, A. et al. (2019). Technical analysis and sentiment embeddings for market trend prediction. *Expert Systems with Applications*, 135, 60–70. Available from <https://doi.org/10.1016/j.eswa.2019.06.014> [Accessed 16 October 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].

Pournader, M. et al. (2020). Blockchain applications in supply chains, transport and logistics: a systematic review of the literature. *International Journal of Production Research*, 58 (7), 2063–2081. Available from <https://doi.org/10.1080/00207543.2019.1650976> [Accessed 22 October 2022].

Press, W.H. (ed.). (2007). *Numerical recipes: the art of scientific computing*, 3rd ed. Cambridge, UK; New York: Cambridge University Press.

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].

Rejeb, A., Rejeb, K. and G. Keogh, J. (2021). Cryptocurrencies in Modern Finance: A Literature Review. *ETIKONOMI*, 20 (1), 93–118. Available from <https://doi.org/10.15408/etk.v20i1.16911> [Accessed 22 October 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].

Rosenthal, S. et al. (2014). SemEval-2014 Task 9: Sentiment Analysis in Twitter. Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014). 2014. *Dublin, Ireland: Association for Computational Linguistics*, 73–80. Available from <https://doi.org/10.3115/v1/S14-2009> [Accessed 16 October 2022].

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].

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].

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

Sagheer, A. and Kotb, M. (2019). Time series forecasting of petroleum production using deep LSTM recurrent networks. *Neurocomputing*, 323, 203–213. Available from <https://doi.org/10.1016/j.neucom.2018.09.082> [Accessed 17 October 2022].

Sarkodie, S.A., Ahmed, M.Y. and Owusu, P.A. (2022). COVID-19 pandemic improves market signals of cryptocurrencies–evidence from Bitcoin, Bitcoin Cash, Ethereum, and Litecoin. *Finance Research Letters*, 44, 102049. Available from <https://doi.org/10.1016/j.frl.2021.102049> [Accessed 16 October 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.

Scharding, T. (2019). National Currency, World Currency, Cryptocurrency: A Fichtean Approach to the Ethics of Bitcoin. *Business and Society Review*, 124(2), 219–238.

Serafini, G. et al. (2020). Sentiment-Driven Price Prediction of the Bitcoin based on Statistical and Deep Learning Approaches. *2020 International Joint Conference on Neural Networks (IJCNN)*. July 2020. Glasgow, United Kingdom: IEEE, 1–8. Available from <https://doi.org/10.1109/IJCNN48605.2020.9206704> [Accessed 16 October 2022].

Shen, D., Urquhart, A. and Wang, P. (2019). Does twitter predict Bitcoin? *Economics Letters*, 174, 118–122. Available from <https://doi.org/10.1016/j.econlet.2018.11.007> [Accessed 16 October 2022].

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].

Siami-Namini, S., Tavakoli, N. and Siami Namin, A. (2018). A Comparison of ARIMA and LSTM in Forecasting Time Series. *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. December 2018. Orlando, FL: IEEE, 1394–1401. Available from <https://doi.org/10.1109/ICMLA.2018.00227> [Accessed 17 October 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].

Taylor, S.J. and Letham, B. (2017). Forecasting at scale. *PeerJ Preprints*. Available from <https://doi.org/10.7287/peerj.preprints.3190v2> [Accessed 17 October 2022].

Tzen, B. and Raginsky, M. (2019). Neural Stochastic Differential Equations: Deep Latent Gaussian Models in the Diffusion Limit. Available from <http://arxiv.org/abs/1905.09883> [Accessed 2 December 2022].

Ugurlu, U., Oksuz, I. and Tas, O. (2018). Electricity Price Forecasting Using Recurrent Neural Networks. *Energies*, 11 (5), 1255. Available from <https://doi.org/10.3390/en11051255> [Accessed 14 February 2023].

Valencia, F., Gómez-Espinosa, A. and Valdés-Aguirre, B. (2019). Price Movement Prediction of Cryptocurrencies Using Sentiment Analysis and Machine Learning. *Entropy*, 21 (6), 589. Available from <https://doi.org/10.3390/e21060589> [Accessed 16 October 2022].

Valipour, M. (2015). Long-term runoff study using SARIMA and ARIMA models in the United States: Runoff forecasting using SARIMA. *Meteorological Applications*, 22 (3), 592–598. Available from <https://doi.org/10.1002/met.1491> [Accessed 16 October 2022].

Wang, Y. et al. (2019). Making sense of blockchain technology: How will it transform supply chains? *International Journal of Production Economics*, 211, 221–236. Available from <https://doi.org/10.1016/j.ijpe.2019.02.002> [Accessed 22 October 2022].

Wang, Y., Liao, W. and Chang, Y. (2018). Gated Recurrent Unit Network-Based Short-Term Photovoltaic Forecasting. *Energies*, 11 (8), 2163. Available from <https://doi.org/10.3390/en11082163> [Accessed 28 December 2022].

Wilson, C. (2019). Cryptocurrencies: The Future of Finance? In: Yu, F.-L.T. and Kwan, D.S. (eds.). *Contemporary Issues in International Political Economy*. Singapore: Springer Singapore, 359–394. Available from <https://doi.org/10.1007/978-981-13-6462-4_16> [Accessed 22 October 2022].

Wolf, T. et al. (2020). Transformers: State-of-the-Art Natural Language Processing. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. October 2020. Online: Association for Computational Linguistics, 38–45. Available from <https://doi.org/10.18653/v1/2020.emnlp-demos.6> [Accessed 19 October 2022].

Yenidogan, I. et al. (2018). Bitcoin Forecasting Using ARIMA and PROPHET. *2018 3rd International Conference on Computer Science and Engineering (UBMK)*. September 2018. Sarajevo: IEEE, 621–624. Available from <https://doi.org/10.1109/UBMK.2018.8566476> [Accessed 16 October 2022].

Zhang, R. et al. (2022). Comparison of ARIMA and LSTM for prediction of hemorrhagic fever at different time scales in China. *PLOS ONE*, 17 (1), e0262009. Available from <https://doi.org/10.1371/journal.pone.0262009> [Accessed 17 October 2022].

# **APPENDIX A – INTRODUCTION**

# **A.1. Prototype feature diagram**

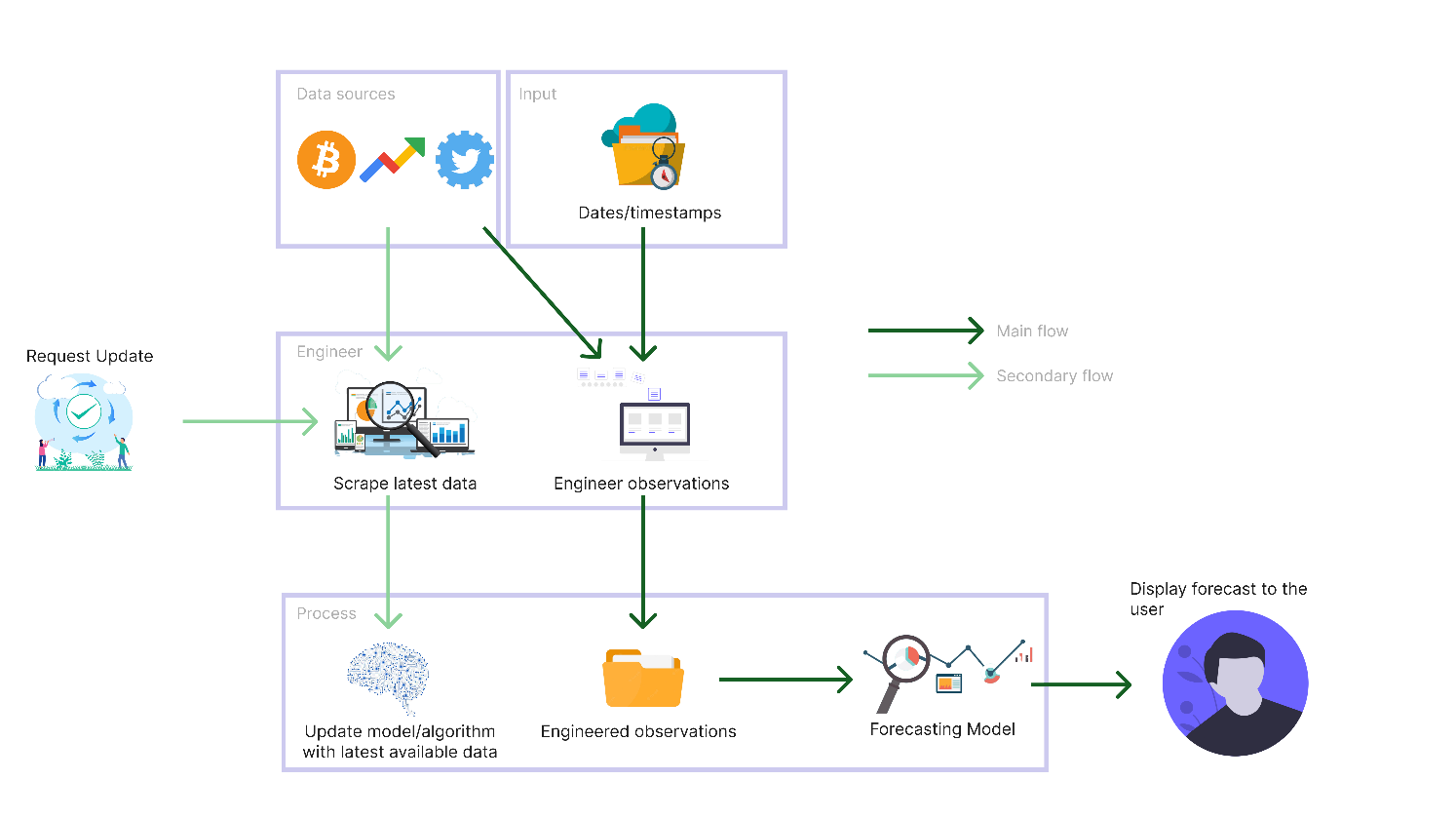
The diagram below depicts the prototype feature diagram proposed in the proposal document.

Figure 23: Prototype feature diagram (*Self-Composed*)

# **A.2. Project scope**

**In scope**

* Implementing a novel LTC architecture capable of being used as currently existing solutions and the corresponding creation of a system.
* Periodic updates of the model with the latest available data.
* Evaluate and compare the implemented system against existing solutions.
* 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.

**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.

**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.

# **APPENDIX B – LITERATURE REVIEW**

# **B.1. Analysis of forecasting algorithms**

Table 32: Analysis of forecasting algorithms

|  |  |  |  |
| --- | --- | --- | --- |
| **Ref.** | **Brief** | **Improvements/Contribution** | **Limitations/Future work** |
| **Statistical-based forecasting algorithms** | | | |
| [Box et al., 2015](#arimaref) | **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 and cannot capture data volatility. |
| [Engle, 1982](#engleref) | **GARCH**. A modeling technique that specializes in predicting volatility in data. | Captures volatility in datasets and boasts significant performance improvements in the family of statistical forecasting algorithms. | Needs to improve interpretability and adaptability. |
| [Taylor and Letham, 2017](#taylorref) | **Prophet**. A modular regression model with interpretable parameters. These parameters can be adjusted according to the problem by domain experts, similar to ARIMA. | Solves forecasting at scale, where scale refers to three types. 1) A large number of people forecasting. 2) A large variety of problems. 3) A large number of forecasts being created. | It uses simple and weak assumptions and produces much poorer performance than ARIMA. And it does not model relationships between the past and future. |
| **DL-based forecasting algorithms** | | | |
| [Hochreiter and Schmidhuber, 1997](#hochreiterref) | **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. They are also prone to overfitting. |
| [Cho et al., 2014](#choref) | **GRU**. Similar architecture to that of LSTMs but combine the “forget” and “input” gates to create two gates, “reset” and “update”, instead of the three found in LSTMs. | Solve the vanishing gradient problem in RNNs as LSTMs, but also consume less memory and run faster. | Suitable for problems with smaller datasets and tend to be less accurate for datasets with larger sequences. |
| [Oreshkin et al., 2020](#orekshinref) | **N-BEATS**. An architecture that solves the univariate time series point forecasting problem. It carries some benefits, some of which are being understandable, easily applicable to multiple other fields, and being fast to train. | Outperformed the M4 competition winner of the previous year and improved the statistical benchmark forecast. | Tailored specifically for univariate TS analysis, therefore, would perform poorly on multivariate analysis. Additionally, Meta-learning is speculated to be a reason for the performance and must be investigated. |
| [Lim et al., 2019](#limtftref) | **TFT**. An attention-based architecture that solves multi-horizon forecasting with interpretability of the used inputs. | Demonstrate significant performance improvements over set benchmarks for a variety of datasets. | Training and inference times are expensive and require moderately extensive resources. Hardware optimizations can reduce these. |
| [Hasani et al., 2020](#hasani2020ref) | **LTC**. A novel formulation of the NODE architecture. Boasts superior expressivity that is capable of adapting to unforeseen changes. | Surpassed traditional DL and statistical models and overcame the underwhelming performance of other neural ODE architectures. | It cannot model uncertainty and is computationally intensive. |

# **B.2. Studies associated with these algorithms**

Table 33: Few studies associated with these algorithms

|  |  |  |  |
| --- | --- | --- | --- |
| **Ref.** | **Technology** | **Outperforms** | **Findings** |
| **Statistical-based forecasting algorithms** | | | |
| [Zhang et al., 2022](#zhangref) | ARIMA | LSTM | ARIMA outperformed the LSTM model for monthly and weekly forecasts, while LSTM performed better for daily forecasts. Additionally, they mentioned that there is no clear superior. |
| [Yenidogan et al., 2018](#yenidoganref) | Prophet | ARIMA | Prophet is strong to outliers and missing data. It is also optimized for business forecasts with trends and seasonality within and nonlinear data growths, which ARIMA cannot handle. |
| **DL-based forecasting algorithms** | | | |
| [Kuan et al., 2017](#kuanref) | GRU | LSTM, traditional GRU | Scaled exponential linear units were proposed to deal with vanishing gradients. These architectures significantly outperformed LSTM and traditional GRU models. |
| [Ugurlu, Oksuz and Tas, 2018](#ugurluref) | GRU | MLP, LSTM | Performed much better than LSTM and trained faster as it is simply a simpler form of LSTM. |
| [Wang, Liao and Chang, 2018](#wang2018ref) | GRU | LSTM, ARIMA | K-Means clustering and Pearson coefficient were used to cluster groups and extract the most important features, respectively, which were then used to train the model. |
| [Sagheer and Kotb, 2019](#sagheerref) | LSTM | ARIMA, GRU | Genetic algorithms were used to create an optimized LSTM architecture. |
| [Bouktif et al., 2018](#bouktifref) | LSTM | MLP, Linear regression | The most optimal time lags and number of layers for an LSTM was found using genetic algorithms. |
| [Fischer and Krauss, 2018](#fischerref) | LSTM | MLP, Logistic regression | Peeked into the internals of the LSTM to find common stock patterns in noisy data. |
| [Pan et al., 2019](#panref) | LSTM | MLP, traditional LSTM | Utilized a novel attention-based LSTM architecture to improve the performance of traditional LSTMs. |
| [Oreshkin et al., 2020](#orekshinref) | N-BEATS | Competition winner | Surpassed the previous winner of the M4 competition with a significant difference in performance. Concluded that hybrid models are only **sometimes** the best-performing. |
| [Hasani et al., 2020](#hasani2020ref) | LTC | LSTM, GRU, CT-RNN, CT-GRU | A more stable implementation of the neural ODE can be built by using a linear system of ODEs to declare the network’s flow. |

# **APPENDIX C – SRS**

# **C.1. Requirement Elicitation Methodologies**

Table 34: Stakeholder groups (*Self-Composed*)

|  |  |  |  |
| --- | --- | --- | --- |
| **Group** | **Stakeholders** | **Reason** | **Instrument** |
| G1 | Domain experts (NLP Experts, AI Researchers, Data Scientists) | In order to respond to research questions and discover anything the author may have overlooked, gather any insights and information especially in the study area. | Interview |
| G2 | Domain and General Users | Gather requirements which will help develop features expected in the application. | Survey & LR |
| G3 | Competitors | Analyze any existing systems related to the research and understand how the project can be enhanced | Self-Evaluations & Brain Storming |
| G4 | Developers | Cross checking if the project is feasible to be continued with. | Prototyping |

# **C.2. Interview analysis**

Table 35: Interview participant details

|  |  |  |
| --- | --- | --- |
| **ID** | **Affiliation** | **Expertise related to the research** |
| P1 | Google Brain visiting researcher and Associate Professor at University of Toronto. | Neural ODEs and SDEs. |
| P2 | Research scientist at Deepmind. | Neural ODEs and SDEs. |
| P3 | Research scientist at Meta AI. | Probabilistic DL and differential equations. |
| P4 | PhD candidate at University of Nottingham. | XAI |
| P5 | Chief Product Officer at Niftron. | Blockchain and cryptocurrencies. |

Table 36: Interview thematic analysis themes, conclusions & evidence

|  |  |  |
| --- | --- | --- |
| **Theme** | **Conclusion** | **Evidence** |
| **Research component** | | |
| Research Problem & Gap | The interviewees validated the research gap and the defined problem. They were also happy that the author had been conducting this research, as few papers were published in this domain. | “Yes, there are many TS forecasting algorithms; however, many are obsolete.”  “Yes, the chosen field of architectures can be considered an advancement.”  “As per my knowledge, I have not seen a system using the basic LTC architecture itself, so this new architecture will be novel.” |
| Requirements | The interviewees were concerned that ODEs and SDEs could be expensive to compute and hence could take some time, which can be an issue given that the forecasts must be produced quickly. Therefore, the author must optimize the model as much as possible to avoid user-unfriendliness. | “They are expensive to compute.”  “It can be resource-intensive.” |
| Advice | The author had initially planned on only creating an implementation of the LTC architecture proposed by Hasani et al. ([2020](#hasani2020ref)). However, the author could further improve the architecture by using SDEs instead (the base LTC uses ODEs), which could manifest into a novel algorithm, which is the author’s current aim as it carries more significance and a potentially more outstanding contribution. | “I think latent ODEs are obsolete.”  “You should look into latent SDEs instead.”  “Latent SDEs are more flexible, you could try applying LTC architectures to those more flexible models instead.” |
| Other suggestions | What was concluded here was that XAI is primarily present for image classification, and there needs to be more literature on the TS domain. However, XAI integration into TS modelling could be confusing and complicated due to the temporal component. Additionally, XAI for SDEs needs to be researched, which the author could look into if time permits. | “Yea, in the domain of TS I have not seen many explainable AI research conducted.”  “Explainable AI is flourishing in image classification but I have not seen it in TS.”  “Integrating explainable AI might not be straightforward as other domains.” |
| **Problem domain** | | |
| Robustness | The interview was an additional validation for the data collected in the survey. Most suggestions were to use as many extra features as possible to make the model robust. Therefore, the author will ensure that they utilize the mentioned exogenous features. | “It is best if you try to include as many features as possible.”  “It is not practical to forecast with only historical prices.” |

# **C.3. Survey analysis**

Table 37: Survey thematic analysis codes, themes & conclusions

|  |  |  |  |
| --- | --- | --- | --- |
| **Code** | | **Theme** | |
| Exogenous factors | | Robustness | |
| Explainability, Insights | | Reliability | |
| Simplicity, Convenience | | User-friendly | |
| Tuning | | Editability | |
| On-demand | | Future consideration | |
| **Theme** | **Conclusion** | | **Evidence** |
| Robustness | Participants believed that prediction needed more than just including historical prices and that social media Trends and other factors (ex: sentiment) are required to make the system as robust and performant as possible. | | “Use previous trends in the past.”  “Consider all possible external factors.” |
| Reliability | Almost all respondents requested that the system provide an Explainability component so that the insights obtained can be reliable as the inference becomes as transparent as possible. | | “Insights about the forecast will be beneficial.”  “Provide as much Explainability to make the prediction as credible as possible.”  “The rate of success of the prediction would be useful.” |
| User-friendly | A couple of participants requested that the system provide some cryptocurrency news to make it convenient and make the inference procedure as straightforward as possible, so there is no hindrance. | | “Show some news about the current cryptocurrency world in the platform, so it’s convenient for the users.”  “Make the steps from choosing a date to forecasting as simple as possible.” |
| Editability | An ML-knowledgeable participant mentioned that it would be an ideal scenario if the system could tune the hyperparameters of the model in use, which could be an excellent enhancement to the system as the model anyways retrains periodically. | | “Coming from machine learning point of view, I think it’ll be a good idea if there’s a functionality to change the hyperparameters used.” |
| Future considerations | A couple of participants mentioned some additional features the author believes they will not be able to cover, given the time allotted. | | “Predict the market for any given time duration.”  “Ability to identify a pump and dump scenario compared to an actual increase in the price of stock/crypto.” |

# **C.4. Use case descriptions**

Table 38: Use case description UC:03; UC:04; UC:05

|  |  |
| --- | --- |
| Use case | Manage exogenous features |
| Id | UC:03; UC:04, UC:05 |
| Description | Manage and process new data without the need for manual interaction. |
| Actor | None |
| Supporting actor (if any) | None |
| Stakeholders (if any) | None |
| Pre-conditions | The latest available data must be scraped and available. |
| Main flow | 1. A cron job triggered fetches the latest historical prices, tweets, Twitter volume, trends, and block reward size data. 2. Twitter volume, Google trends, and block reward size are scaled and cleaned. 3. Tweets undergo sentiment analysis to determine current speculation. 4. The sentiment is further weighted based on the tweeter’s importance. 5. Features are combined with historical closing prices to create an enriched dataset and retrain the model. |
| Alternative flows | None |
| Exceptional flows | 1. The script could not fetch recent data – retry a few days later or alert Admin for manual overhaul. |
| Post-conditions | A new enriched dataset with the features is generated. |

Table 39: Use case description UC:07

|  |  |
| --- | --- |
| Use case | Update model hyperparameters |
| Id | UC:07 |
| Description | Manually change the hyperparameters used by the model. |
| Actor | Admin |
| Supporting actor (if any) | None |
| Stakeholders (if any) | None |
| Pre-conditions | All the data must be scraped and preprocessed (as the model would ideally need to be retrained upon hyperparameter tuning). |
| Main flow | 1. Admin authorizes themselves. 2. Admin can change the hyperparameters in use to a set of predefined values. 3. The system ensures data available is up-to-date (must be in this case, as the script will run periodically automatically). If not:    1. Obtains the latest available data.    2. Performs sentiment analysis and self-retrains. 4. The system retrains itself with the data and new hyperparameters. |
| Alternative flows | None |
| Exceptional flows | None |
| Post-conditions | The model is updated with the chosen hyperparameters. |

# **C.5. Functional requirements**

Table 40: ‘MoSCoW’ technique of requirement prioritization

|  |  |
| --- | --- |
| **Priority level** | **Description** |
| M (Must have) | The author must implement requirements with this priority for the project to succeed. |
| S (Should have) | Requirements that would be of value but are not necessary. |
| C (could have) | Features that are optional and have no significant impact. It is desirable to implement them if time permits. |
| W (Will not have) | Requirements that will not be a part of the implementation at this point. |

# **APPENDIX D – DESIGN**

# **D.1. LTS algorithm intuition**

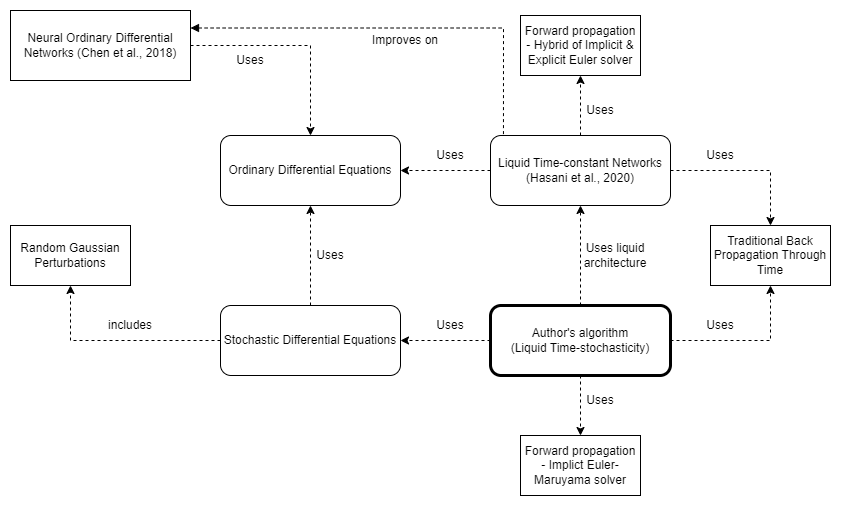


Figure 24: Algorithm intuition (*Self-Composed*)

**What exactly an SDE solves compared to an ODE?**

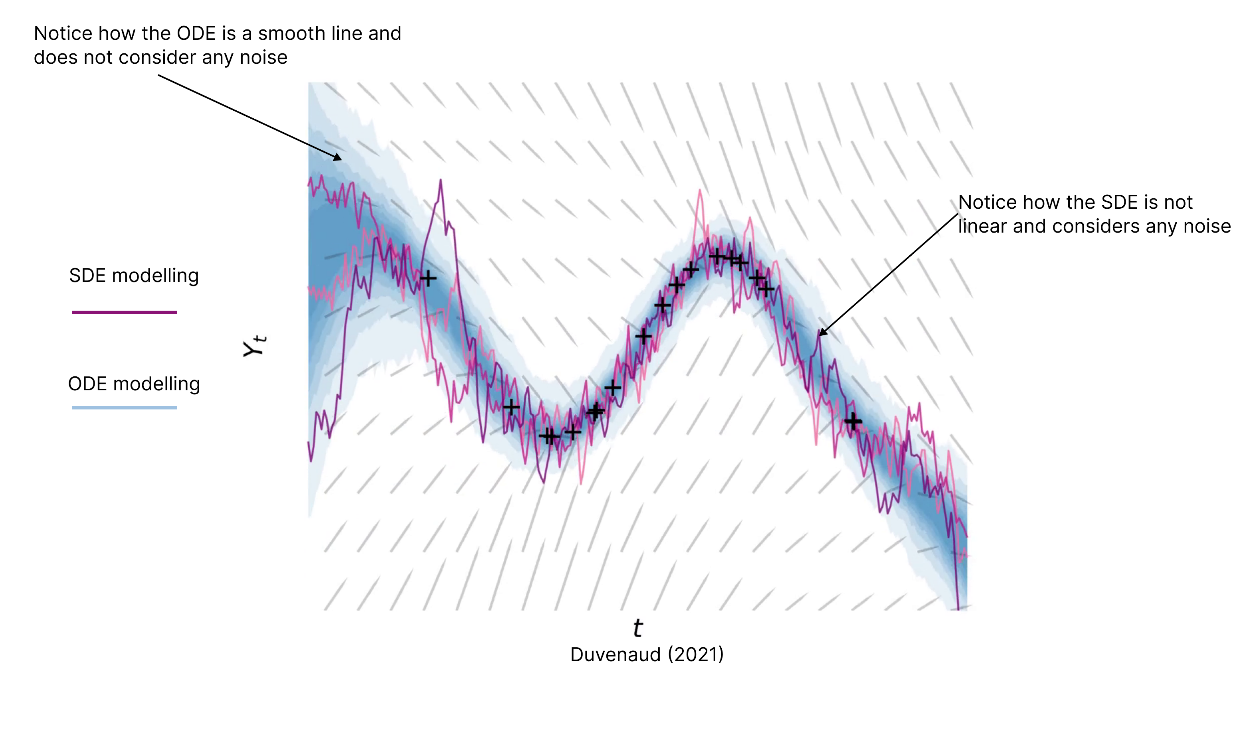


Figure 25: Understanding what an SDE solves

Considering this, SDEs would more accurately model domains that have high volatility/noise.

**Existing LTC architecture**

|  |  |
| --- | --- |
|  | *Time-constant* |
|  | *Hidden state* |
|  | *Input* |
| *t* | *Time* |
|  | *Neural network* |
|  | *Parameters* |

The above formulation was proposed byHasani et al. ([2020](#hasani2020ref)), where a system of linear ODEs is used to declare the flow of the hidden state; the ODEs are of the following form.

*Where S(t) represents the following nonlinearity*

The equation manifests by plugging the above equation into the system of linear ODEs.

**Formulation**

***Step 01 – transitioning from an ODE to an SDE***

In simple terms, an SDE is an ODE with additional noise added at each step, which the model can use to model uncertainty.

The above ODE can be used to calculate the ‘expected’ slope, whereas the ‘realized’ slope differs from the ‘expected’ due to random noise, also called random Gaussian perturbations or Gaussian white noise. With that in consideration, the following can be derived:

However, noise can be of varying intensities (some could be high, some could be low). Considering this varying intensity, the SDE can be further expressed as follows:

As implied, the missing factor in the existing architecture that consists of ODEs is the absent stochastic transition dynamics (i.e., a noise for each timestep – which is vital to model the tiny unobserved interactions). The above equation considers the small unobserved interactions and uncertainties that could occur; this is further important in the context of TS data, as the initial state of data is unlikely to be certain.

***Step 02 – adding neural networks into SDE dynamics***

Based on the findings of Duvenaud ([2021](#duvenaudref)), the noise mentioned in the previous step can be considered as Brownian motion: a generalized form of the Gaussian noise. Researchers can produce the following by plugging Brownian motion into the equation determined in the previous step.

A neural network can be integrated into the above equation to solve the system, resulting in the following equation:

***Step 03 – Integrating the above equation into the LTC architecture***

Moving back to the main problem at hand, the author can now construct a new formula by using the equation determined in the previous step.

The above equation is a linear system of ODEs initially proposed by Lapicque ([1907](#lapicqueref)) - the author could add the uncertainty noise to the equation to produce the following:

The above equation now defines a stochastic process instead of deterministic evolution. Therefore, researchers can model any tiny unobserved interactions.

Finally, the following could be derived by applying this to the LTC formula:

# **D.2. Tweet sentiment weighting algorithm intuition**

The proposed formula is a linear combination of the metrics considered by the author.

Initially, the metrics are specified and defined as follows:

Alpha, beta, gamma and delta are different weighting factors applied to each metric. This is determined by observing how much of an impact each metric makes. Applying these weighting factors will produce the following:

The author wanted to provide more weightage to the tweeter themselves rather than the specific tweet, as the impact of a tweet is more correlated with the tweeter rather than its likes and retweets. As such, the author proposes the arbitrary values of **0.5**, **0.3**, **0.1** and **0.1** for alpha, beta, gamma and delta respectively.

To avoid a specific metric from dominating the score (if the number of followers is in the millions and number of retweets is in the thousands, the number of followers would dominate the score), a logarithm can be applied. Applying a logarithm can normalize and balance all the metrics into a similar scale to avoid any bias, as such accurately reflect each metric’s true impact on the score. Furthermore, the author will add the value “1” to each metric prior to obtaining the logarithmic value. Performing this will prevent mathematical errors that would arise for the case of a metric having the value of 0. The terms can now be written as follows:

As the formula is a linear combination of these metrics, they are summed up to provide a tweet sum and an influencer sum.

The obtained sums do not fall within any range and must be normalized to ensure that the final score will not exceed 1. Normalization can be applied as follows:

Finally, this influencer score can be applied to the vanilla compound score to obtain a weighted compound score.

# **D.3. LTS algorithm complexity analysis**

Table 41: Complexities of BPTT and adjoint sensitivity

**Note:** *L* = number of steps

|  |  |  |
| --- | --- | --- |
|  | **BPTT** | **Adjoint sensitivity** |
| Time | O(L) | **O(LlogL)** |
| Memory | O(L) | **O(1)** |
| Forward accuracy | High | High |
| Backward accuracy | **High** | Low |

# **D.4. UI wireframes**

|  |  |
| --- | --- |
| Figure 26: UI wireframes – Home (*Self-Composed*) | Figure 27: UI wireframes – News (*Self-Composed*) |
| Figure 28: UI wireframes – Cryptocurrencies (*Self-Composed*) | Figure 29: UI wireframes – Cryptocurrency (*Self-Composed*) |

|  |  |
| --- | --- |
| Figure 30: UI wireframes – Admin login (*Self-Composed*) | Figure 31: UI wireframes – Admin metrics (*Self-Composed*) |
| Figure 32: UI wireframes – Forecast (*Self-Composed*) | |

# **APPENDIX E – IMPLEMENTATION**

# **E.1. Selection of programming language**

The below table summarizes the analysis of the language chosen for the data science component, where each option was given a score within H – High, M – Medium, and L – Low.

Table 42: Selection of data science language

|  |  |  |  |
| --- | --- | --- | --- |
| **Data science**  Two of the most popular languages used widely for data science were analyzed to implement the core data science components. | | | |
| **Aspect** | **Relevance** | **Python** | **R** |
| Availability of libraries. | A language supporting multiple libraries is paramount, as the author would require numerous techniques to gather the necessary data and streamline the model and algorithm development. | H | M |
| Author familiarity and ease of implementation. | Implementing the algorithm, the mathematical intricacies, and the respective model should be as simple as possible. It is an additional benefit if the author has hands-on experience with the chosen language. | H | M |
| Learning curve | The difficulty of the chosen language must not be a hindrance, as the goal is to utilize the tool to implement a system rather than spending time learning the language. | L | M |
| Community and documentation. | Community support and well-written documentation are important, as the author will not have time to debug trivial issues. | H | M |
| **Conclusion**  Based on the analysis, the author decided to use **Python**, as it was more relevant. | | | |

# **E.2. Selection of Deep Learning (DL) framework**

Table 43: Selection of DL framework

|  |  |
| --- | --- |
| **Framework** | **Description** |
| TensorFlow | Used for production-level applications, has detailed documentation and community support, and handles large datasets. It also provides better visualization options, making it easy to debug and monitor training, which is vital as a novel algorithm is being built, and no comparison is present. |
| PyTorch | It is more lightweight and developer-friendly, as it provides a higher-level development. Therefore, it has a much smaller learning curve, easier to get started, and feels more intuitive as it is simpler to build models. |
| **Conclusion**  The author opted to use **TensorFlow**. Although it is more complicated, the higher-level API: Keras, is now officially a part of TensorFlow. Therefore, model development has become much more straightforward. Additionally, building the algorithm requires more low-level details ([Kurama, 2022](#pytorchvstensorflowref)). | |

# **E.3. Selection of User Interface (UI) framework**

Table 44: Selection of UI framework

|  |  |
| --- | --- |
| **Framework** | **Description** |
| Angular | Suitable for large-scale applications with dedicated submodules for particular functionalities. However, it can be less performant in comparison and unnecessarily heavy. |
| Vue | A tiny framework that takes little to no time to startup and is much more intuitive as the code is simple. Additionally, based on simulations, it has been identified to perform better than Angular and React. However, it has much fewer resources. |
| Svelte | The most lightweight and genuinely reactive. Much more performant than the rest; however, it has a small community of developers and is relatively new. |
| React | Customizable and promotes code reusability via functions as components. It carries a large community and is open-source while being SEO-friendly. Additionally, the React developer tools is very handy. |
| **Conclusion**  Based on the analysis, the author chose **React** as the GUI built will be simple, and there is no requirement for large-scale applications, as it is not the primary focus ([Patadiya, 2021](#angularvsreactvsvuevssvelteref)). | |

# **E.4. Selection of Application Programming Interface (API) framework**

Table 45: Selection of web framework

|  |  |
| --- | --- |
| **Framework** | **Description** |
| Flask | A very lightweight framework that provides only the simplest of functionalities. However, it is the preferred choice for ML API development because it is light. |
| Django | Suitable for more larger scaled applications that provide a vast range of functionalities, it is stricter and less flexible. Therefore, is much more demanding and heavier. |
| **Conclusion**  The author chose **Flask** as it provides only the necessities in exposing an ML model and since the luxury features provided by Django (ex: authentication) were not required ([InterviewBit, 2021](#flaskvsdjangoref)). | |

# **E.5. Fetch data**

**Fetch historical prices**



Figure 33: Fetch historical prices (*Self-Composed*)

The above script describes a couple of functions that can be used to fetch the latest BTC historical prices data and store it into a MongoDB collection, so that it can be later read from by the model. A third-party API was used to fetch the data, as existing APIs are all discontinued.

**Fetch Twitter volume, Google Trends & block reward size**

|  |  |
| --- | --- |
| Figure 34: Fetch Twitter volume (*Self-Composed*) | Figure 35: Fetch block reward size (*Self-Composed*) |
| Figure 36: Fetch google trends (*Self-Composed*) | |

The above scripts fetch the Twitter volume and block reward from a website that publicly exposes this data. Therefore, a simple website scraping tool can be used without authentication or authorization.

**Fetch tweet data**

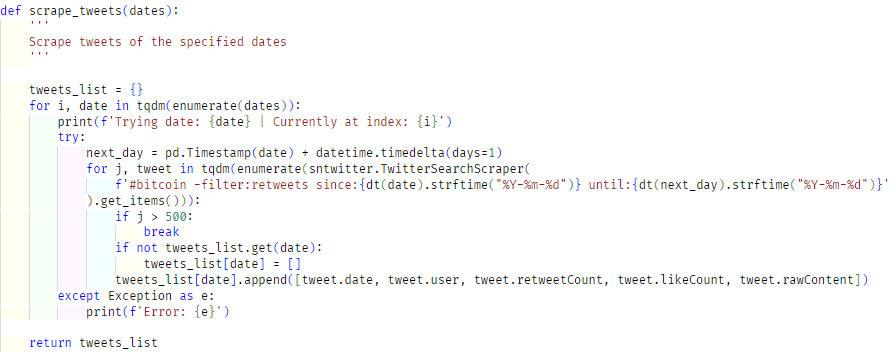


Figure 37: Scrape tweets (*Self-Composed*)

Obtaining the tweet data required a more tedious process as the Twitter API had been updated only to provide tweets for the past week; however, third-party libraries offer this functionality. Tweets fetched were limited to 500 for a single day due to time, performance, and storage constraints, and as the application is not the core contribution. Initially, tweets were fetched up to a specific time point; in the future, the above script could be run to scrape tweets of particular dates that are described to be from the days currently existing in the collection up to the day at which the script is run. There is a further limitation as only ‘#bitcoin’ is searched.



Figure 38: Clean tweets (*Self-Composed*)

As this research is currently limited to only English, the tweets are filtered, and non-English tweets are removed.

# **E.6. Preprocessing**

**Tweet sentiment analysis**

The main step of preprocessing is to perform sentiment analysis on the obtained tweet data. In this research, the VADER sentiment analyzer is used as determined in previous chapters.

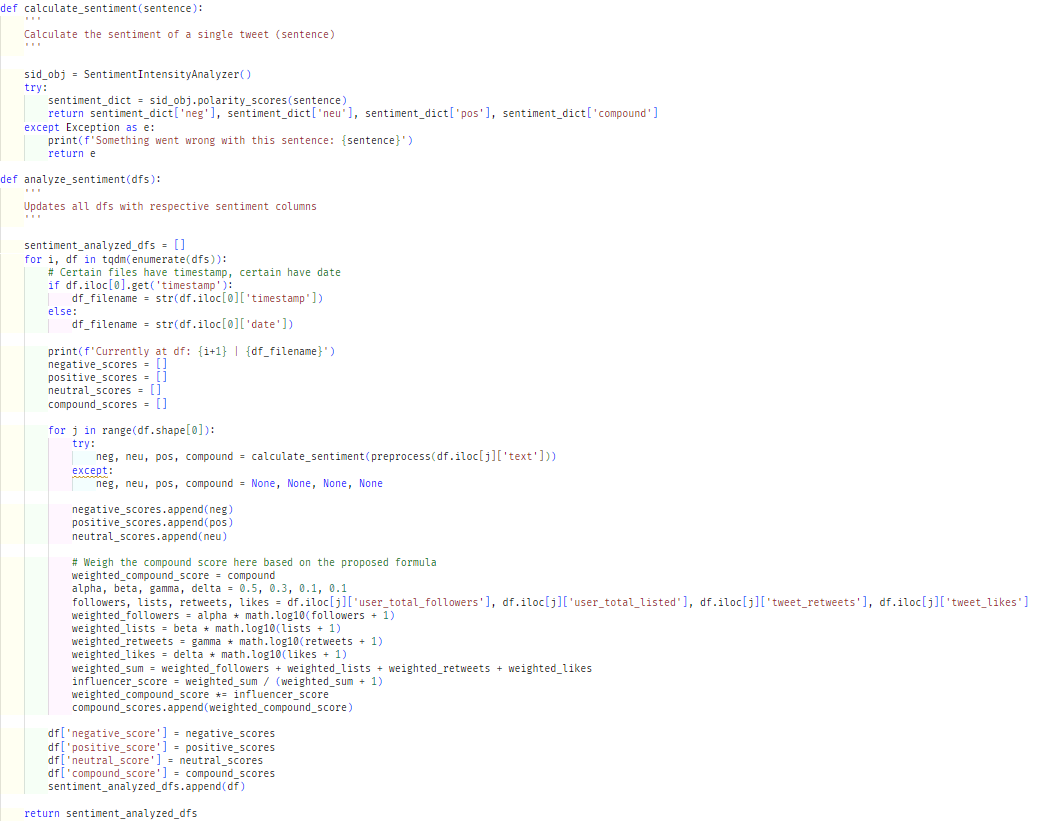


Figure 39: Analyze sentiments (*Self-Composed*)

The above script is used to perform sentiment analysis on the tweets and concatenates the negative, positive, neutral, and compound scores into the existing tweet dataset. The compound score is weighted beforehand by utilizing the proposed sentiment weighting formula in [**Chapter 6**](#_6.5.2.2_Tweet_sentiment)(as only the compound score is used on forth, there is no requirement to weight the negative, positive and neutral scores). They can then be condensed down to create an average score for a single day.

**Tweet dataset condensation**

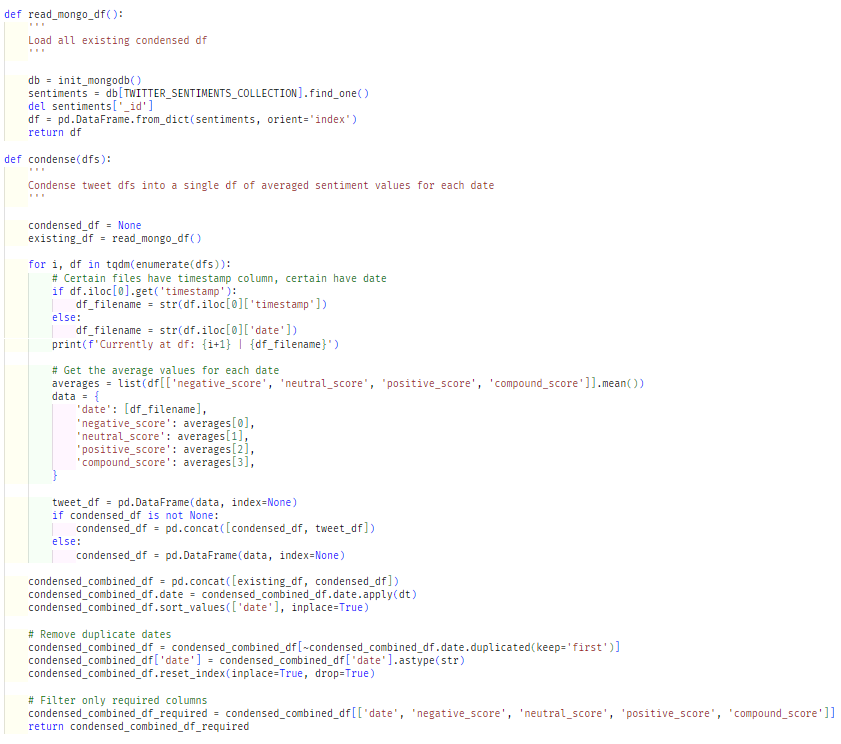


Figure 40: Combine and condense tweets (*Self-Composed*)

As the other data being used directly creates a single data frame with a row for each date, the condensation process is unnecessary. However, as the tweet data fetched consists of a separate dataframe file for each date, this data must be compressed to the same format as other datasets.

The above script condenses the tweet dataset into a single data frame by averaging the sentiment scores for each day.

**Final dataset creation**

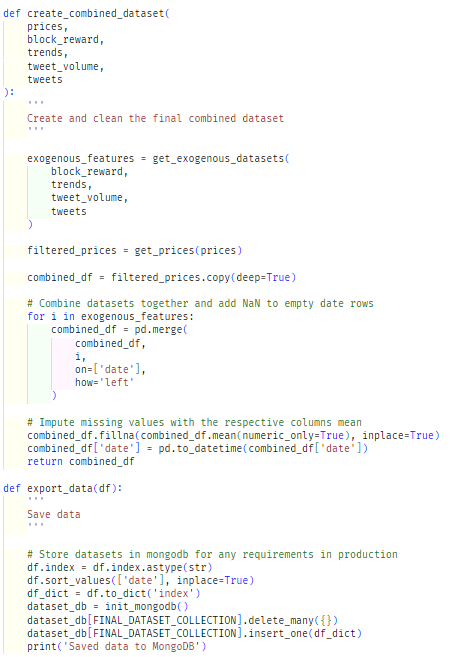


Figure 41: Combine all datasets (*Self-Composed*)

The above script is used to create the final dataset that the model uses. It fetches all the datasets and combines them into a single data frame. Initially, a helper function removes unneeded columns from the data, which were decided upon conducting correlation tests initially. The mean of their respective columns imputes missing values of each feature of specific dates, and the combined dataset is stored in a MongoDB collection so the model can finally utilize it.

# **E.7. User interface**

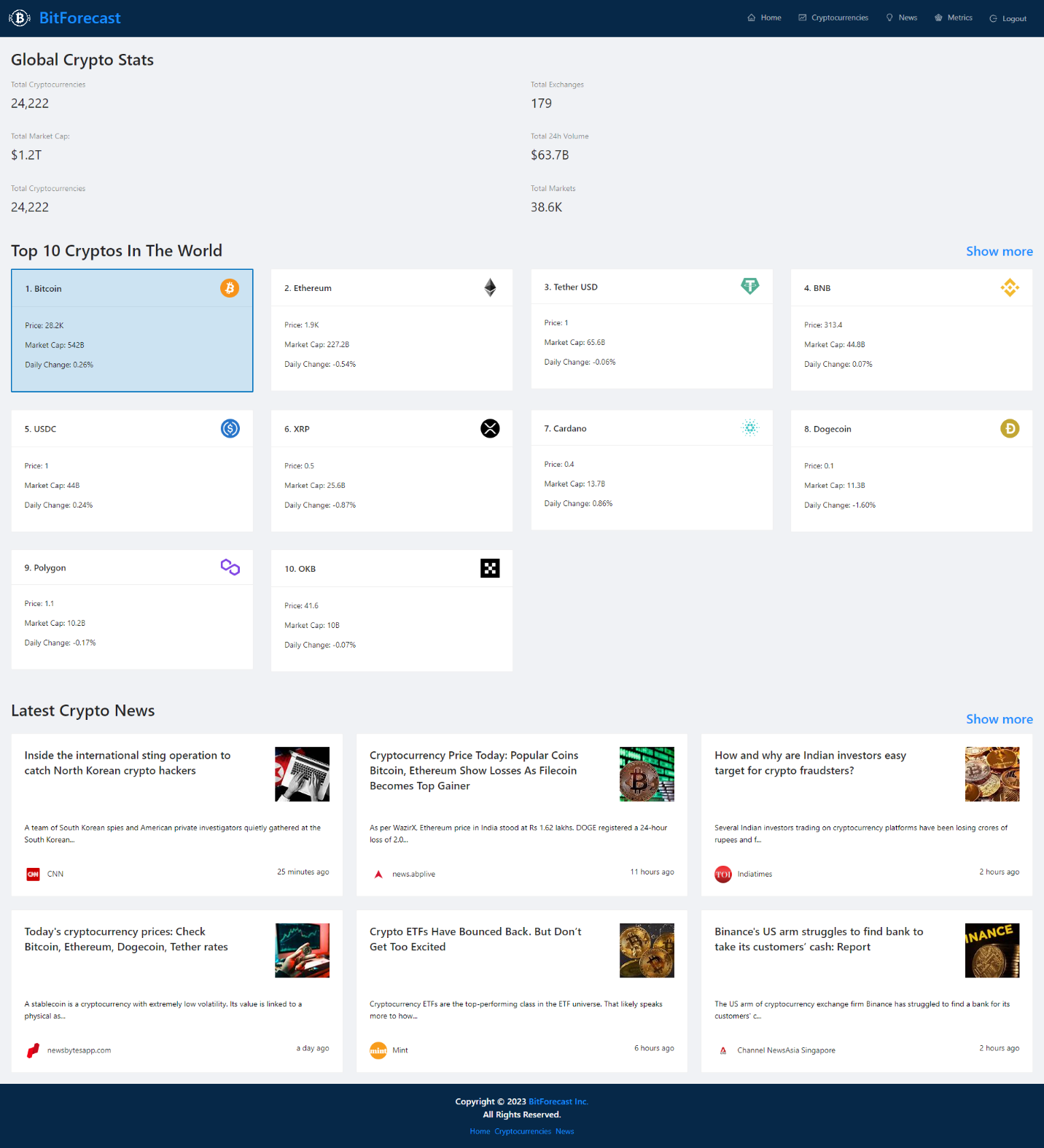


Figure 42: GUI - Home (*Self-Composed*)

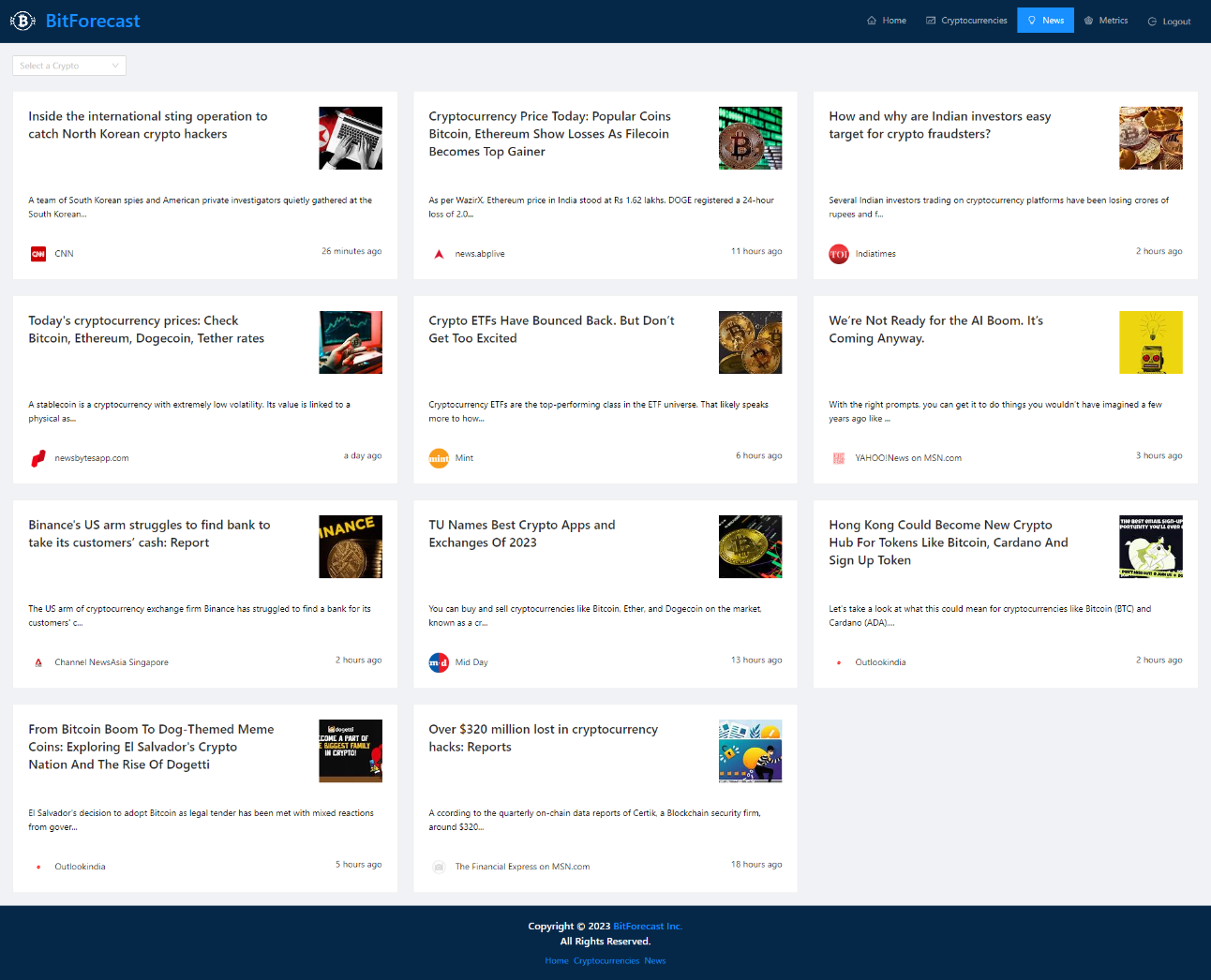


Figure 43: GUI - News (*Self-Composed*)

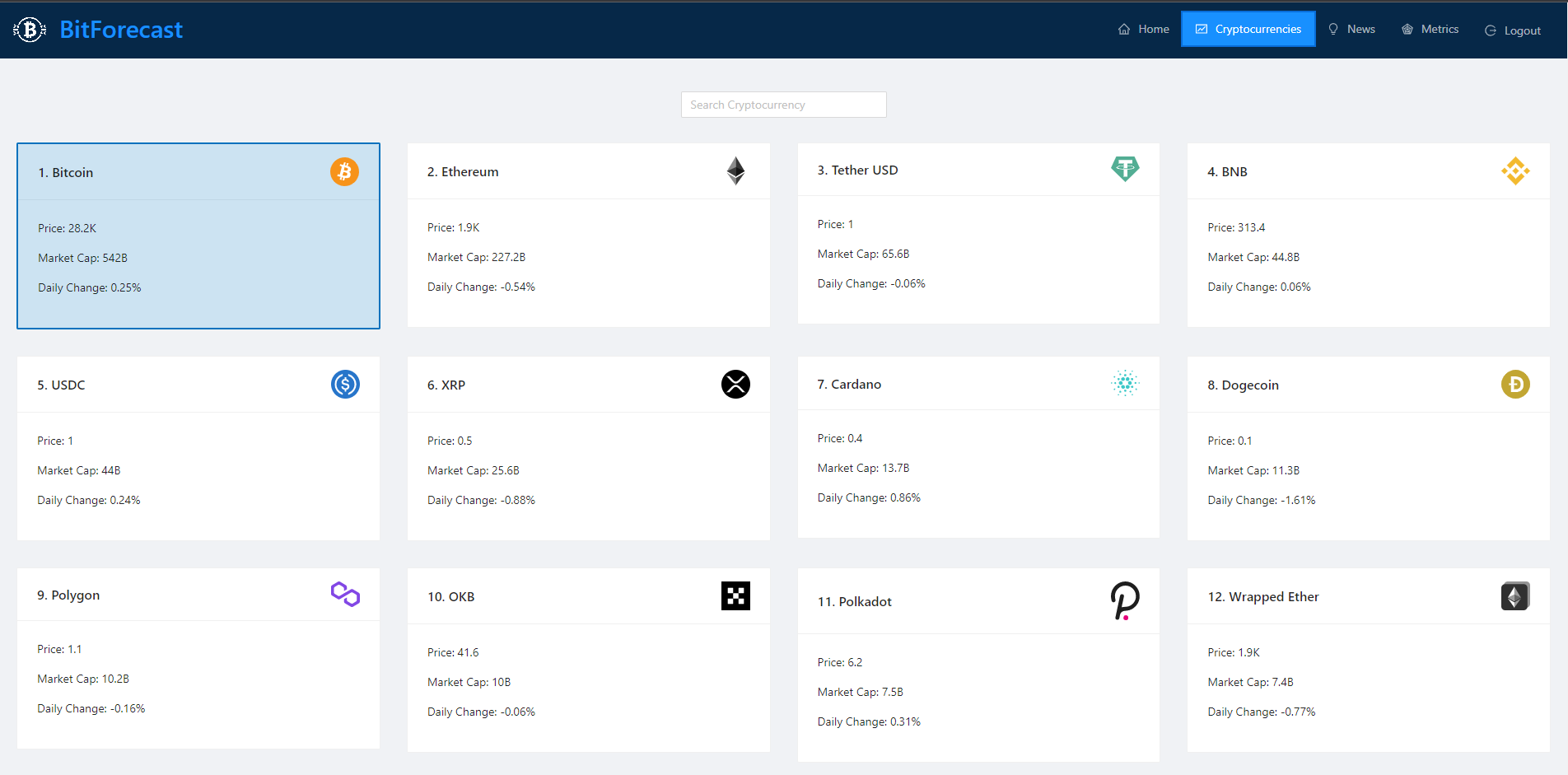


Figure 44: GUI - Cryptocurrencies (*Self-Composed*)



Figure 45: GUI - Cryptocurrency (*Self-Composed*)

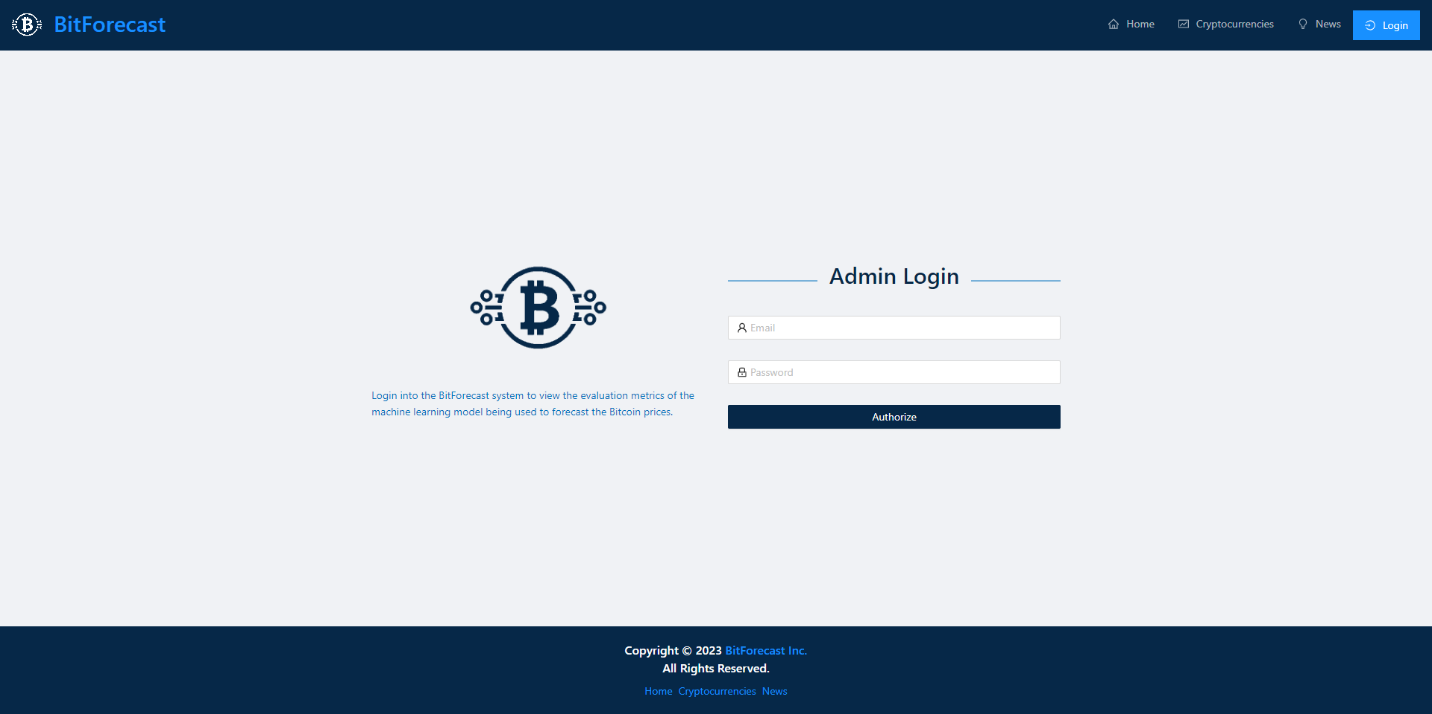


Figure 46: GUI - Admin login (*Self-Composed*)

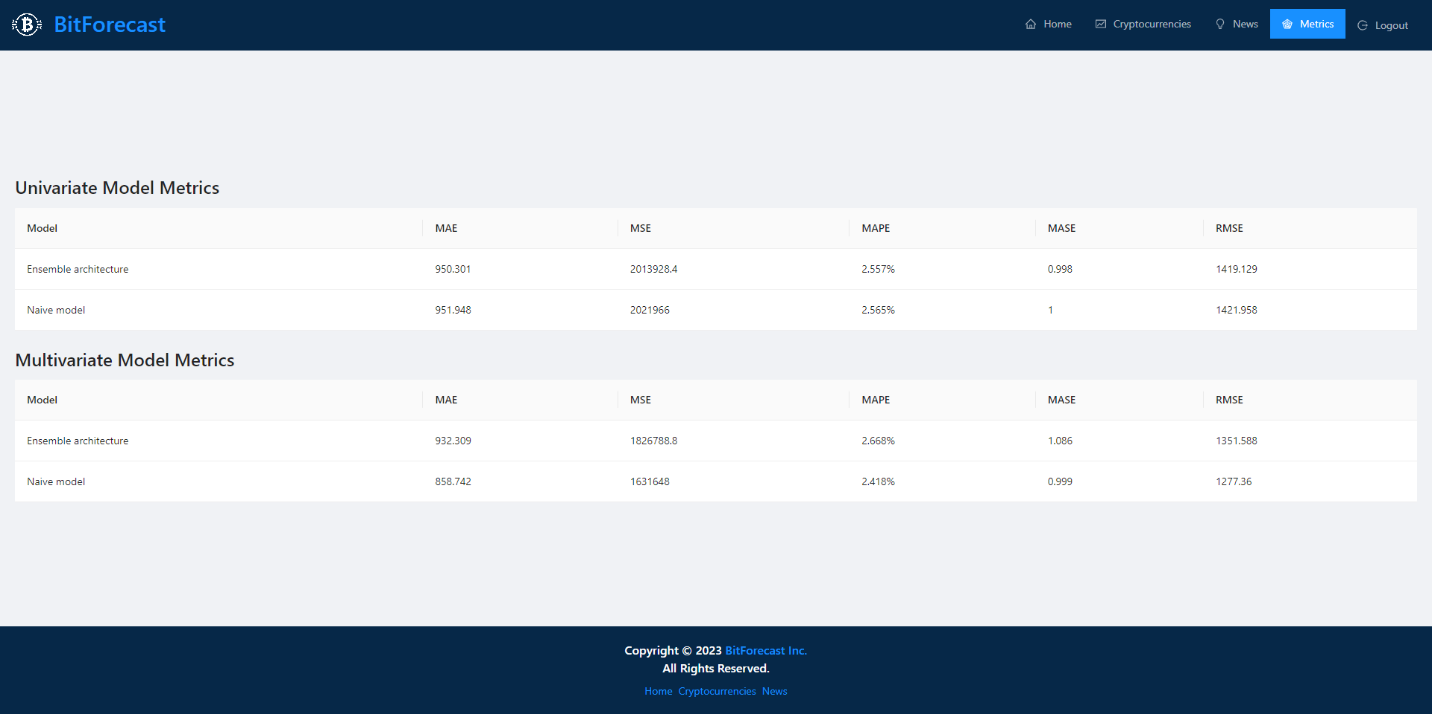


Figure 47: GUI - Admin metrics (*Self-Composed*)



Figure 48: GUI - Forecast (*Self-Composed*)

# **APPENDIX F – TESTING**

# **F.1. Functional testing**

Table 46: Functional testing

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test case** | **ID** | **Action** | **Expected result** | **Actual result** | **Status** |
| **Research level** | | | | | |
| 1 | [FR1](#fr1) | - | The LTS follows recommended standards so that it can be scalable and built upon. | The architecture was built as a Keras layer so behind-the-scenes techniques that need to happen is handled by Keras. | Passed |
| 2 | [FR2](#fr2) | - | The LTS can be used as existing layers such as Conv1D and LSTM. | Building the LTS as a Keras layer provided this functionality. | Passed |
| **System level** | | | | | |
| 3 | [FR3](#fr3), [FR4](#fr4), [FR5](#fr5) | Users choose the future dates. | The user can view the prices of the chosen dates. | The deployed endpoint is triggered and the model’s response of the chosen dates are returned to the user. | Passed |
| 4 | [FR3](#fr3), [FR8](#fr8), [FR9](#fr9) | The user can view the price ranges of the chosen dates. | The deployed endpoint is triggered and the models’ responses of the chosen dates are returned to the user. | Passed |
| 5 | [FR6](#fr6), [FR7](#fr7) | Cron script is triggered periodically. | The latest data available is stored in the database. | The exogenous features are scraped/extracted, processed, condensed, combined, and saved into the database. | Passed |
| 6 | [FR10](#fr10) | Upon receiving the responses, the user views the updated graph. | The graph is updated with the predictions and plotted alongside the past prices. | The GUI is updated with more datapoints that are the future predictions. | Passed |
| 7 | [FR11](#fr11) | Sentiments extracted from scraped data. | Sentiments are weighted based on influencer score by the proposed formula. | The sentiments undergo a weighting stage where, based on certain metrics, the score is changed. | Passed |
| 8 | [FR14](#fr14) | Admins log into the system. | Technical information about the models is shown. | The evaluation metrics of the two models are displayed. | Passed |

# **F.2. Non-functional testing**

The author applied performance, GUI and maintainability testing, and a few test-cases to determine if the system meets the non-functional requirements and the design goals.

**Performance testing**

The author had deployed the API and model; therefore, there is no requirement of having a high GPU and CPU power. Docker, GitHub Actions and Heroku with basic Dynos were utilized for deployment purposes, which is capable of serving requests for small-scale applications. However, for large-scale purposes, it is recommended that the Dynos are scaled up, as the application would not be able to handle multiple requests concurrently. It is also worth mentioning that as the system is developed using TensorFlow, initial load times can take some time.

**GUI testing**

The requirement gathering phase determined that the need for developing a simple and effective GUI was important. The GUI was tested by Google Lighthouse to determine its performance and accessibility, the diagram below illustrates the obtained results.

|  |  |
| --- | --- |
| Figure 49: Lighthouse home page (*Self-Composed*) | Figure 50: Lighthouse login page (*Self-Composed*) |
| Figure 51: Lighthouse cryptocurrencies page (*Self-Composed*) | Figure 52: Lighthouse cryptocurrency page (*Self-Composed*) |
| Figure 53: Lighthouse news page (*Self-Composed*) | Figure 54: Lighthouse metrics page (*Self-Composed*) |

The results vary from page-to-page. This is likely since most pages utilize third party APIs to render information, hence demonstrating a subpar performance value.

**Maintainability testing**

Maintainability is important so that future research on the system and especially the developed algorithm can be conducted seamlessly. CodeFactor and CodeQL were used to ensure that the repositories are maintained and documented well and that there are no vulnerabilities.

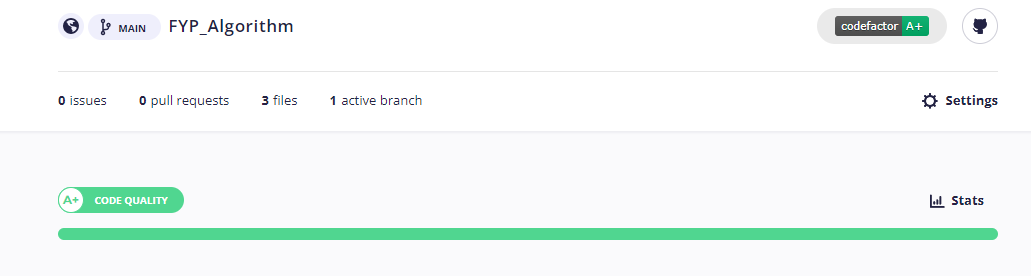


Figure 55: CodeFactor - Algorithm repository (*Self-Composed*)

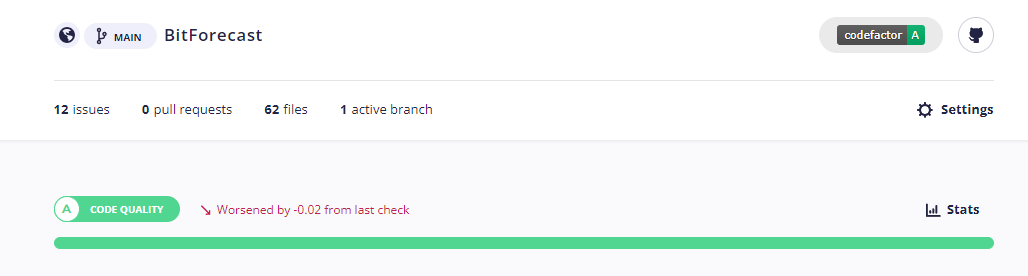


Figure 56: CodeFactor - Application repository (*Self-Composed*)

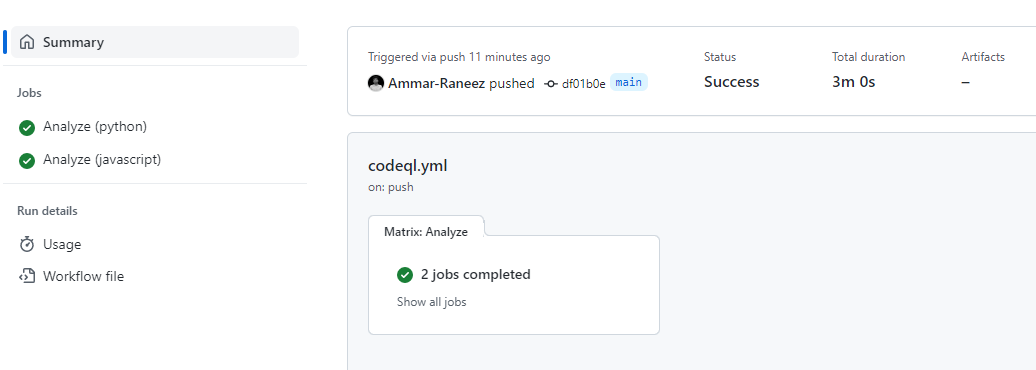


Figure 57: CodeQL - Algorithm repository (*Self-Composed*)

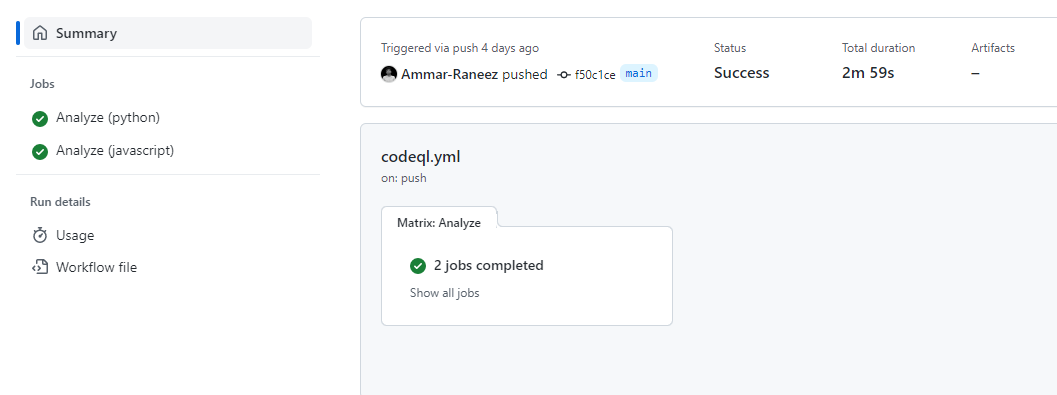
****

Figure 58: CodeQL - Application repository (*Self-Composed*)

**Test cases**

Table 47: Non-functional testing

|  |  |  |
| --- | --- | --- |
| **Non-functional requirements** | | |
| **Test case** | **ID** | **Result** |
| 1 | [NFR1](#nfr1) | The system gives responses within 1-2 minutes. |
| 2 | [NFR2](#nfr2) | Only the data for specific dates are scraped and updated whenever necessary. |
| 3 | [NFR3](#nfr3) | The developed GUI is simple, easy to follow, and attractive. Additionally, no technical information is displayed to the users, except for admins who log in. |
| 4 | [NFR4](#nfr4) | Docstrings were added for each method and comments wherever necessary. |
| 5 | [NFR5](#nfr5) | The responses are plotted within the graph itself to show a growth/decline. |
| 6 | [NFR6](#nfr6) | The system was deployed to Heroku by Docker and GitHub Actions, but will not scale as traffic increases: the utilized Dynos are the most basic version. |
| 7 | [NFR7](#nfr7) | The model is stored in AWS S3 for backup and requests, the data is stored in MongoDB, and admin authentication is handled by Firebase. Therefore, security is integrated to some extent. |
| 8 | [NFR8](#nfr8) | The application is responsive across a wide range of screen sizes. |
| **Design goals** | | |
| 9 | [DG1](#dg1) | The data are stored in MongoDB and fetched whenever necessary. For the case where the available data is not up-to-date, the script fetches data only for the missing dates and updates the database. |
| 10 | [DG2](#dg2) | The system is built to be as user-friendly as possible with zero information shown to users on what happens behind the scenes. |
| 11 | [DG3](#dg3) | The forecast is plotted alongside the existing price chart using a different color to differentiate between them. Two more lines are plotted to demonstrate the uncertainty estimations that display the range of prices. |
| 12 | [DG4](#dg4) | Docstrings were added for each method and comments wherever necessary. Analysis using CodeFactor produced a grade of A+ for the algorithm repository, which is the maximum grade possible. |

# **APPENDIX G – EVALUATION**

# **G.1. Expert evaluators**

Table 48: Selected expert evaluator details

|  |  |  |
| --- | --- | --- |
| **ID** | **Affiliation** | **Expertise related to the research** |
| **Research domain** | | |
| EV1 | Google Brain visiting researcher and Associate Professor at University of Toronto. | Neural ODEs and SDEs. |
| EV2 | Research scientist at Deepmind. | Neural ODEs and SDEs. |
| EV3 | Research scientist at Meta AI. | Probabilistic DL and differential equations. |
| EV4 | PhD candidate at University of Nottingham. | ML & DL. |
| **Problem domain** | | |
| EV5 | Prefer not to say | Blockchain and cryptocurrencies. |
| EV6 | Prefer not to say | Cryptocurrencies and crypto exchanges. |

# **G.2. Evaluation of functional requirements**

Table 49: Evaluation of the implementation of functional requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Description** | **Priority** | **Use Case** | **Evaluation** |
| **Research level** | | | |  |
| [FR1](#fr1) | A robust and scalable implementation of the novel algorithm must follow recommended standards. | M | **-** | Implemented |
| [FR2](#fr2) | The developed algorithm must be able to be used as existing layers and algorithms (ex: LSTM, CNN). | M | - | Implemented |
| **System level** | | | |  |
| [FR3](#fr3) | Users must be able to choose a future date. | M | UC:01 | Implemented |
| [FR4](#fr4) | Users must be able to view the point prediction price. | M | UC:03 | Implemented |
| [FR5](#fr5) | The system must generate the point prediction price based on the user’s choice of date. | M | UC:02 | Implemented |
| [FR6](#fr6) | The script must obtain the latest data available periodically. | M | UC:04 | Implemented |
| [FR7](#fr7) | The script must extract trends and sentiments from obtained data. | M | UC:05 | Implemented |
| [FR8](#fr8) | Users should be able to view a range of prices along with the single-point price. | S | UC:03 | Implemented |
| [FR9](#fr9) | The system should generate higher and lower bound uncertainty estimations. | S | UC:02 | Implemented |
| [FR10](#fr10) | The GUI should plot the forecast with the current prices in a single graph to show the growth/decline. | S | UC:03 | Implemented |
| [FR11](#fr11) | The script could weight sentiment based on any influential personnel’s tweet. | C | UC:06 | Implemented |
| [FR12](#fr12) | The system could display some insights to the user, such as a highly influential tweet that made it predict the price. | C | UC:03 | Not-considered |
| [FR13](#fr13) | Admins could authenticate and update the model with different parameters. | C | N/A | Not-considered |
| [FR14](#fr14) | Admins could get additional information about a prediction, such as the evaluation metric and accuracy. | C | N/A | Implemented |
| [FR15](#fr15) | The system will not produce forecasts for other cryptocurrencies. | W | N/A | Not-considered |
| [FR16](#fr16) | The system will not produce real-time forecasts (ex: hourly). | W | N/A | Not-considered |
| Functional requirement completion percentage = | | | | |

# **G.3. Evaluation of non-functional requirements**

Table 50: Evaluation of the implementation of non-functional requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Requirement** | **Description** | **Priority** | **Evaluation** |
| [NFR1](#nfr1) | Performance | The system must take little time to generate a forecast, given that a couple of extra features are in use. | Important | Implemented |
| [NFR2](#nfr2) | Performance | The system must not unnecessarily keep updating its data. | Important | Implemented |
| [NFR3](#nfr3) | Usability | The user interface must be simple and effective and provide user-friendly errors if any occur. | Important | Implemented |
| [NFR4](#nfr4) | Maintainability | The author must document the codebase well in case of future reference, mainly the algorithm development repository. | Important | Implemented |
| [NFR5](#nfr5) | Quality | The output must be of good quality so that it provides vital insights. | Desirable | Implemented |
| [NFR6](#nfr6) | Scalability | The system must be deployed to a cloud with no scaling issues and good resources for efficient and optimal performance, especially as there could be multiple concurrent active user requests. | Desirable | Implemented (~50%) |
| [NFR7](#nfr7) | Security | The system must be resilient to attackers, specifically to prevent data manipulation. | Desirable | Implemented |
| [NFR8](#nfr8) | Compatibility | To ensure compatibility, the developer must test the system on most browsers and mobile phones. | Desirable | Implemented |
| [NFR9](#nfr9) | Availability | In critical failures, the primary operator must be available and solve issues as soon as possible. | Desirable | Not- implemented |
| Non-functional requirement completion percentage = | | | | |

Table 51: Evaluation of the achievement of design goals

|  |  |  |
| --- | --- | --- |
| **ID** | **Goal** | **Evaluation** |
| [DG1](#dg1) | Performance | Achieved |
| [DG2](#dg2) | Usability | Achieved |
| [DG3](#dg3) | Quality | Achieved |
| [DG4](#dg4) | Maintainability | Achieved |
| Design goals achievement percentage | | |

# **APPENDIX H – CONCLUSION**

# **H.1. Status of research objectives**

Table 52: Status of research objectives

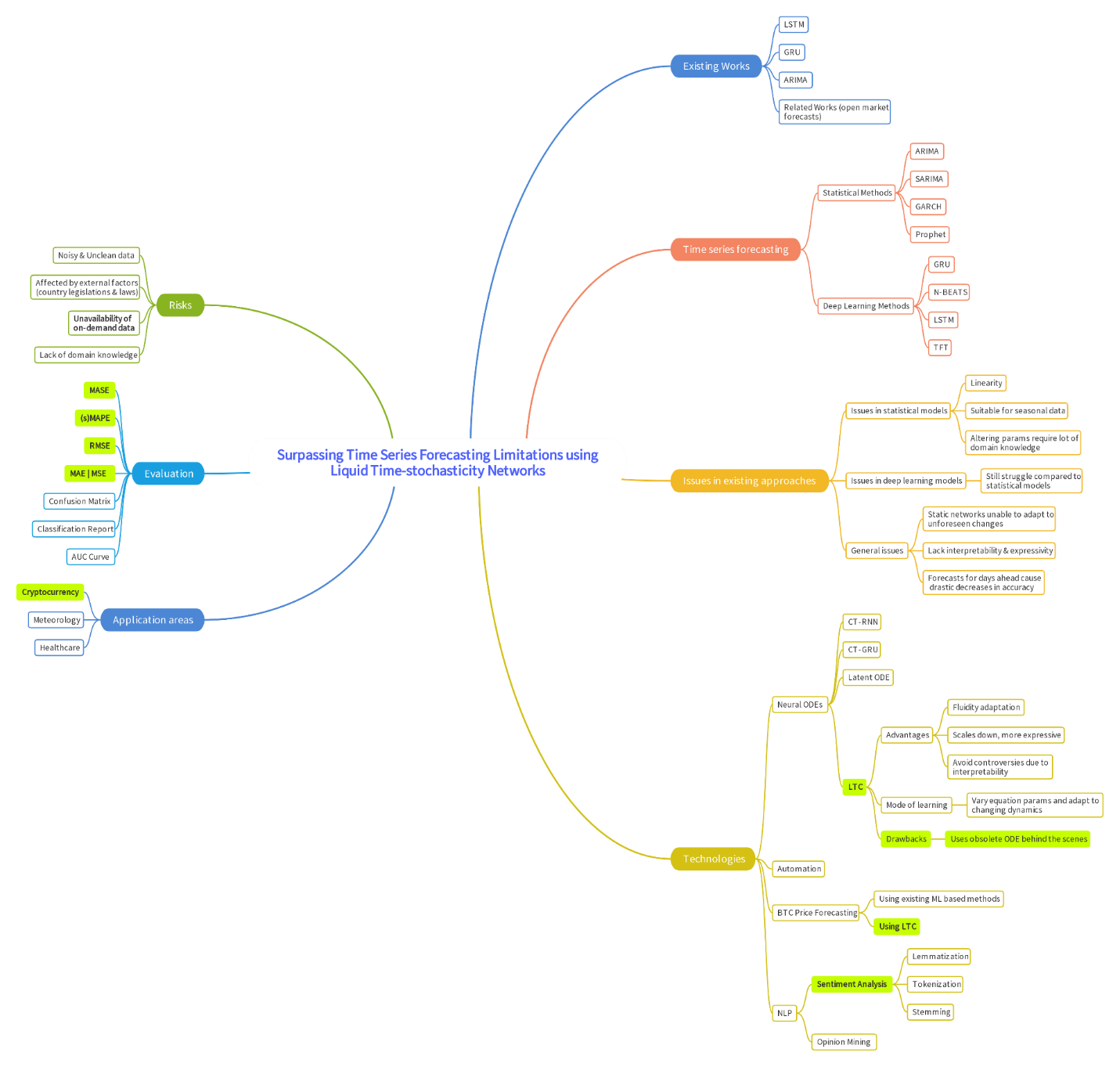
|  |  |  |
| --- | --- | --- |
| **Objective** | **Description** | **Status** |
| Problem Identification | Understand and document the identified problem and provide reasoning on what makes it novel.  **R**O1: Conduct research on a domain of interest and identify a comprehensive enough issue that requires solving.  **R**O2: Delve deeper into the identified problem to obtain a general understanding on how to approach solving the problem.  **R**O3: Split the problem down into manageable subsections so it is easier to digest and to solve one section at a time.  **R**O4: Design a respective schedule, associated deliverables, and the Gantt chart. | Completed |
| Literature Review | Collate relevant information by reading, understanding, and evaluating previous work.  **R**O5: Conduct preliminary studies and investigations on existing TS forecasting systems and algorithms.  **R**O6: Analyze the requirement for specialized TS algorithms.  **R**O7: Conduct research on neural ODEs, LTCs & SDEs.  **R**O8: Obtain deep insights into the architecture behind the LTC.  **R**O9: Research and obtain insights on factors affecting the price of BTC.  **R**O10: Research on existing BTC forecasting and related open market systems.  **R**O11: Research on necessary ML techniques and evaluation approaches. | Completed |
| Requirement Elicitation | Collect and analyze project requirements using appropriate tools and techniques.  **R**O12: Analyze stakeholders and understand their viewpoints and concerns.  **R**O13: Gather the requirements and architectures of LTCs and SDEs.  **R**O14: Collate the most up-to-date details of BTC and obtain insights on the perspectives of the end users.  **R**O15: Design necessary diagrams to justify the product’s specification. | Completed |
| Design | Design the architecture and a corresponding system capable of effectively solving the identified problems.  **R**O16: Design necessary diagrams required to understand the algorithm  RO17: Design diagrams required to understand the supplementary system being developed.  **R**O18: Design the novel algorithm and analyze its complexities. | Completed |
| Implementation | Implement a system that is capable of addressing the research gaps.  **R**O19: Design, evaluate and pick necessary technologies best-suited for the implementation.  **R**O20: Develop an efficient LTC implementation.  **R**O21: Build on the LTC to implement the LTS.  **R**O22: Integrate the algorithm developed into a TS forecasting application.  **R**O23: Integrate the intelligent system into a client application to display forecasts.  **R**O24: Design and implement an automated flow to update the built network with the latest data.  **R**O25: Design and implement a pipeline for easy deployments.  **R**O26: Consider any legal, social, ethical & professional issues upon implementation. | Completed |
| Evaluation | Effectively test the algorithm implemented, the system, and the respective data science model using recommended techniques.  **R**O27: Evaluate the developed algorithm and the respective model against the evaluation metrics researched in the literature review.  **R**O28: Create a test plan & test cases and perform unit, performance, and integration testing. | Completed |
| Documentation | Document the progression of the research project and inform about any challenges faced.  **R**O29: Create a coherent report of new skills obtained, evaluations, contributions etc., and ensure that all the above-stated objectives are met. | Completed |

# **H.2. Achievement of learning outcomes**

Table 53: Achievement of learning outcomes

|  |  |
| --- | --- |
| **Description** | **Learning outcome(s)** |
| The project was broken down into two main subproblems: algorithm implementation and application development. They were then further broken down into digestible units to understand and implement one at a time by applying appropriate techniques recommended by analyzing literature and requirements. | LO1 |
| The units identified were placed into a project plan as milestones to achieve within a set timeframe and successfully complete the project within the given timescale. | LO2 |
| Project requirements were collected and analyzed from two parties: academic researchers and end users, to obtain insights into developing the LTS algorithm and prioritizing features that must be implemented in the BTC forecasting application. | LO3 |
| Literature was read and critiqued upon, both recent and some over a century old, to understand in detail certain mathematical concepts. | LO4 |
| Having obtained requirements, insights and knowledge. The author worked on implementing the two subproblems one unit at a time, whilst learning any new skills required. The author met with the supervisor regularly to make sure that they are on the right track by producing milestone deliverables. Any SLEP issues were also considered and documented. | LO5, LO6, LO7 |
| The research was documented and each individual chapter was presented with the supervisor as milestones. They were then updated based on feedback from the supervisor and the module leader. Before completion of this dissertation, two document artefacts were submitted: the project proposal and PSDP. Additionally, papers were presented in conferences to justify the authors solution. | LO8 |

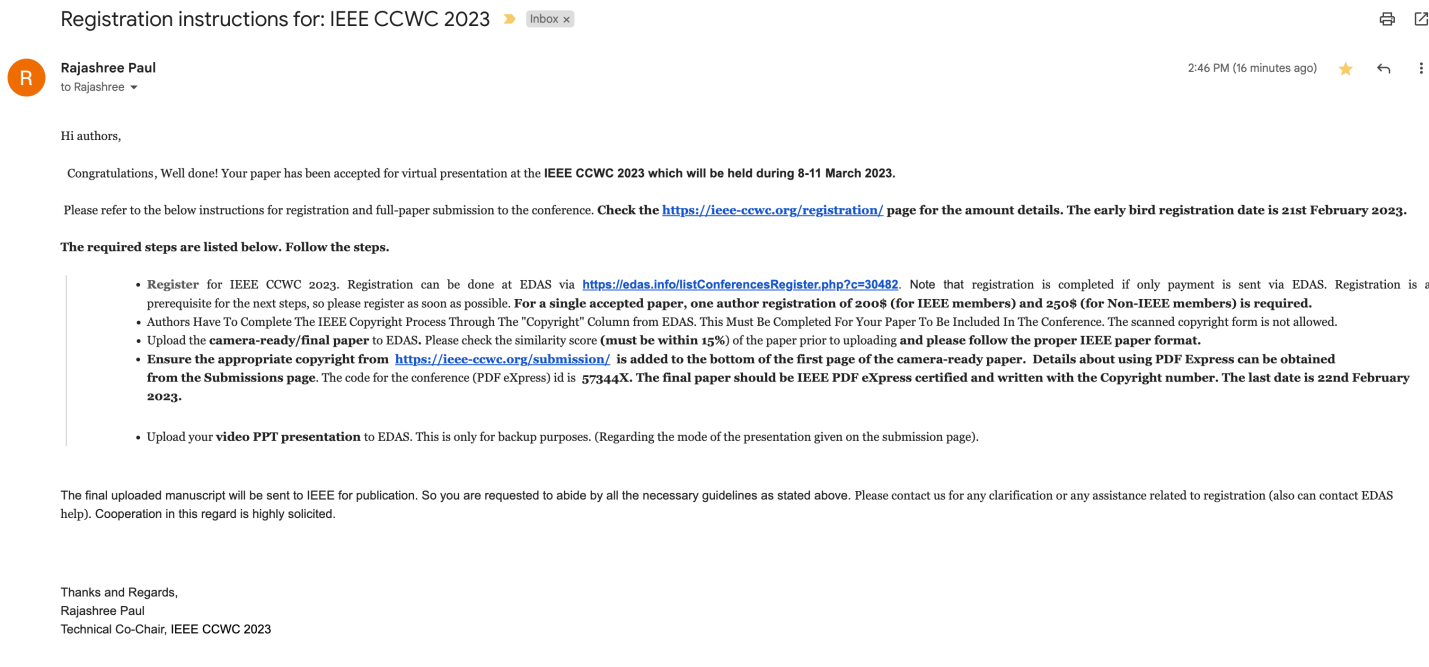
# **APPENDIX I – CONCEPT MAP**



*A clearer version can be found* [*Here*](https://drive.google.com/file/d/1GwNYLEG649gszfBqYi9VXud5MPE70aPZ/view?usp=sharing)

# **APPENDIX J – EXTENDED REVIEW PAPER**

# **J.1. Acceptance notification**



# **J.2. Extended review paper**



