

Informatics Institute of Technology

In Collaboration With

The University of Westminster, UK



*The University of Westminster, Coat of Arms*

## GenSum

### **Adaptive 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:** May 1, 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 with respect to any domain. **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. An Adaptive Approach for Generalized Text Summarization Using Optimized Transformers**

Conference: SmartNets 2023 – Big Data Analytics and Machine Learning

Status: Submitted, In-Review

Venue: Asia, Colombo

Date: April 25, 2023

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

Conference: SmartNets 2023 - Big Data Analytics and Machine Learning

Status: Submitted, In-Review

Venue: Asia, Colombo

Date: April 08, 2023

**APPENDIX L** shows the proof of Research & Review paper submission made at SmartNets 2023 Conference.

## 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 begin by thanking my supervisor, Mr. Torin Wirasingha, for his valuable contribution in refining my project idea and expanding its scope. His insights and guidance have been crucial in shaping my work. Additionally, I express my deepest gratitude to Mr. Guhanathan Poravi, my esteemed module leader, for consistently reviewing my reports and providing invaluable feedback. Thanks to his guidance, I now have the confidence to conduct research at any level. 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 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 opportunities, unwavering support, and 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 motivation have fueled my progress over the span of four years. Lastly, I extend my deepest gratitude to my parents for their support and encouragement in pursuing my dreams, and to my beloved family, friends, and significant other, whose encouragement has been a strength during challenging times.

## CONTENTS

<b>DECLARATION</b> .....	i
<b>ABSTRACT</b> .....	ii
<b>PUBLICATIONS</b> .....	iii
<b>ACKNOWLEDGEMENT</b> .....	iv
<b>LIST OF TABLES</b> .....	xii
<b>LIST OF FIGURES</b> .....	xiv
<b>CHAPTER 01. INTRODUCTION</b> .....	1
<b>1.1 Chapter Overview</b> .....	1
<b>1.2 Problem Domain</b> .....	1
<b>1.2.1 Movie User Reviews</b> .....	1
<b>1.2.2 Text Summarization</b> .....	2
<b>1.3 Problem Definition</b> .....	3
<b>1.3.1 Problem statement</b> .....	3
<b>1.4 Research Motivation</b> .....	3
<b>1.5 Research Questions</b> .....	4
<b>1.6 Research Aim &amp; Objectives</b> .....	4
<b>1.6.1 Research Aim</b> .....	4
<b>1.6.2 Research Objectives</b> .....	5
<b>1.7 Novelty of the Research</b> .....	8
<b>1.7.1 Problem Novelty</b> .....	8
<b>1.7.2 Solution Novelty</b> .....	8
<b>1.8 Research Gap</b> .....	8
<b>1.9 Contribution to the body of knowledge</b> .....	9
<b>1.9.1 Research Domain Contribution</b> .....	9
<b>1.9.2 Problem Domain Contribution</b> .....	10
<b>1.10 Research Challenge</b> .....	10
<b>1.11 Chapter summary</b> .....	10
<b>CHAPTER 02. LITERATURE REVIEW</b> .....	11
<b>2.1 Chapter Overview</b> .....	11
<b>2.2 Concept Map</b> .....	11
<b>2.3 Problem Domain</b> .....	11

<b>2.3.1 User Reviews.....</b>	11
<b>2.4 Existing work.....</b>	16
<b>2.4.1 Text Summarization Systems.....</b>	16
<b>2.4.1.1 Abstractive Approach.....</b>	16
<b>2.4.1.2 Extractive Approach.....</b>	17
<b>2.4.2 Algorithmic approaches for Text Summarization .....</b>	17
<b>2.4.3 Usage of Transformers .....</b>	18
<b>2.5 Technological review .....</b>	19
<b>2.5.1 Proposed architecture for the Generalized Text Summarization System. ....</b>	19
<b>2.5.2 Machine Learning Text Summarization Techniques .....</b>	20
<b>2.5.3 Deep Learning Text Summarization Techniques.....</b>	21
<b>2.5.4 Available Datasets for generalized text summarization .....</b>	21
<b>2.5.5 Preprocessing Techniques used in Text Summarization.....</b>	22
<b>2.6 Evaluation Techniques .....</b>	22
<b>2.6.1 Evaluation Approaches.....</b>	22
<b>2.6.2 Benchmarking .....</b>	24
<b>2.7 Chapter summary .....</b>	25
<b>CHAPTER 03. METHODOLOGY .....</b>	26
<b>3.1 Chapter overview .....</b>	26
<b>3.2 Research methodology .....</b>	26
<b>3.3 Development methodology .....</b>	27
<b>3.3.1 Life cycle model.....</b>	27
<b>3.3.2 Requirement elicitation methodology .....</b>	27
<b>3.3.3 Design methodology .....</b>	27
<b>3.3.4 Software development methodology.....</b>	27
<b>3.4 Project management methodology .....</b>	27
<b>3.4.1 Schedule .....</b>	27
<b>3.4.1.1 Gantt chart .....</b>	27
<b>3.4.1.2 Deliverables .....</b>	28
<b>3.5 Resources .....</b>	29
<b>3.5.1 Software requirements .....</b>	29
<b>3.5.2 Hardware requirements .....</b>	29
<b>3.5.3 Technical skills .....</b>	29

<b>3.5.4 Data requirements .....</b>	30
<b>3.6 Risks &amp; mitigation .....</b>	30
<b>3.7 Chapter summary .....</b>	30
<b>CHAPTER 04. SOFTWARE REQUIREMENTS SPECIFICATION.....</b>	31
<b>4.1 Chapter overview .....</b>	31
<b>4.2 Rich picture .....</b>	31
<b>4.3 Stakeholder analysis .....</b>	32
<b>4.3.1 Stakeholder onion model .....</b>	32
<b>4.3.2 Stakeholder viewpoints.....</b>	33
<b>4.4 Selection of Requirement Elicitation Methodologies .....</b>	34
<b>4.5 Discussion of Findings .....</b>	35
<b>4.5.1 Literature review .....</b>	35
<b>4.5.2 Brainstorming .....</b>	36
<b>4.5.3 Interviews.....</b>	36
<b>4.5.4 Survey.....</b>	38
<b>4.5.5 Self Evaluation .....</b>	42
<b>4.5.6 Prototyping .....</b>	43
<b>4.5.6 Summary of findings.....</b>	44
<b>4.7 Use case Diagram .....</b>	46
<b>4.8 Use case Descriptions.....</b>	46
<b>4.9 Requirements.....</b>	48
<b>4.9.1 Functional Requirements .....</b>	48
<b>4.10 Chapter summary .....</b>	50
<b>CHAPTER 05. SOCIAL, LEGAL, ETHICAL &amp; PROFESSIONAL ISSUES .....</b>	51
<b>5.1 Chapter overview .....</b>	51
<b>5.2 SLEP issues and mitigation.....</b>	51
<b>5.3 Chapter summary .....</b>	51
<b>CHAPTER 06. DESIGN.....</b>	52
<b>6.1 Chapter overview .....</b>	52
<b>6.2 Design goals .....</b>	52
<b>6.3 High Level Design .....</b>	52
<b>6.3.1 Architecture Diagram.....</b>	52

<b>6.3.2 Discussion of tiers of the architecture .....</b>	53
<b>6.4 System Design.....</b>	54
<b>6.4.1 Choice of Design Paradigm .....</b>	54
<b>6.5 Design diagrams .....</b>	55
<b>6.5.1 Data Flow diagrams.....</b>	55
<b>6.5.1.1 Level 01 Data Flow diagram .....</b>	55
<b>6.5.1.2 Level 02 Data Flow diagram .....</b>	56
<b>6.5.4 UI design .....</b>	56
<b>6.5.5 System process activity diagram.....</b>	57
<b>6.6 Chapter summary .....</b>	57
<b>CHAPTER 07. IMPLEMENTATION.....</b>	58
<b>7.1 Chapter overview .....</b>	58
<b>7.2 Technology selection .....</b>	58
<b>7.2.1 Technology stack.....</b>	58
<b>7.2.2 Data selection.....</b>	59
<b>7.2.3 Programming Language Selection .....</b>	59
<b>7.2.4 Development Framework Selection.....</b>	59
<b>7.2.5 Libraries Utilized .....</b>	60
<b>7.2.6 IDE's Utilized .....</b>	61
<b>7.2.7 Summary of Technology Selection .....</b>	61
<b>7.3 Implementation of Core Functionalities .....</b>	62
<b>7.3.1 Automated Hyperparameter Search &amp; Model customization .....</b>	62
<b>7.3.2 Model Usage General &amp; Domain Specific Users .....</b>	65
<b>7.3.3 Model Retraining .....</b>	68
<b>7.3.4 Data Preprocessing .....</b>	69
<b>7.4 User interface .....</b>	69
<b>7.5 Chapter summary .....</b>	69
<b>8.4 Model testing &amp; evaluation .....</b>	70
<b>8.4.1 Model testing .....</b>	71
<b>8.4.2 Model Evaluation .....</b>	72
<b>8.5 Benchmarking .....</b>	74
<b>8.6 Functional testing.....</b>	74

<b>8.7 Module &amp; integration testing .....</b>	74
<b>8.8 Non-functional testing .....</b>	76
<b>8.9 Limitations of the testing process .....</b>	76
<b>8.10 Chapter summary .....</b>	76
<b>CHAPTER 09. EVALUATION .....</b>	77
<b>9.1 Chapter Overview.....</b>	77
<b>9.2 Evaluation Methodology &amp; Approach .....</b>	77
<b>9.3 Evaluation Criteria .....</b>	77
<b>9.4 Self-Evaluation .....</b>	78
<b>9.5 Selection of Evaluators .....</b>	79
<b>9.6 Evaluation Results &amp; Expert Opinions.....</b>	80
<b>9.7 Limitations of Evaluation.....</b>	82
<b>9.8 Evaluation of Functional requirements .....</b>	82
<b>9.9 Evaluation of non-functional requirements.....</b>	82
<b>9.10 Chapter summary .....</b>	82
<b>CHAPTER 10. CONCLUSION .....</b>	83
<b>10.1 Chapter Overview .....</b>	83
<b>10.2 Achievement of Research Aim &amp; Objectives.....</b>	83
<b>10.2.1 Achievement of Aim.....</b>	83
<b>10.2.2 Achievement of Objectives .....</b>	83
<b>10.3 Utilization of knowledge from the degree .....</b>	83
<b>10.4 Use of Existing Skills.....</b>	84
<b>10.5 Use of New Skills .....</b>	85
<b>10.6 Achievement of learning outcomes .....</b>	85
<b>10.7 Problems and Challenges faced .....</b>	85
<b>10.8 Deviations.....</b>	86
<b>10.9 Limitations of the research.....</b>	86
<b>10.10 Future enhancements.....</b>	86
<b>10.11 Achievement of the Contribution to the Body of Knowledge .....</b>	87
<b>10.11.1 Domain Contributions .....</b>	87
<b>10.11.2 Technical Contribution .....</b>	87
<b>10.11.3 Additional Contributions .....</b>	87

<b>10.12 Implementation code .....</b>	87
<b>10.13 Concluding remarks .....</b>	88
<b>REFERENCES.....</b>	I
<b>APPENDIX A – INTRODUCTION .....</b>	VII
<b>A.1. Prototype feature diagram .....</b>	VII
<b>A.2. Project scope.....</b>	VII
<b>APPENDIX B – LITERATURE REVIEW .....</b>	IX
<b>B.1. Related work in abstractive text summarization.....</b>	IX
<b>APPENDIX C – SRS.....</b>	I
<b>C.1. Requirement Elicitation Methodologies.....</b>	I
<b>C.2. Interview analysis.....</b>	I
<b>C.3. Use case descriptions.....</b>	III
<b>APPENDIX D – DESIGN.....</b>	X
<b>D.1. Design goals .....</b>	X
<b>D.2. UI wireframes.....</b>	XI
<b>APPENDIX E – IMPLEMENTATION .....</b>	XIV
<b>E.1. Dataset Resources.....</b>	XIV
<b>E.2. Selection of Programming Language .....</b>	XVIII
<b>E.3. Selection of Deep Learning (DL) framework.....</b>	XIX
<b>E.4. Selection of User Interface (UI) framework.....</b>	XX
<b>E.5. Selection of Application Programming Interface (API) framework .....</b>	XX
<b>E.6. Data Preprocessing.....</b>	XXI
<b>E.7. User interface.....</b>	XXVI
<b>APPENDIX F – TESTING.....</b>	XXXIII
<b>F.1. Functional Testing .....</b>	XXXIII
<b>F.2. Non-functional testing .....</b>	XXXV
<b>APPENDIX G – EVALUATION.....</b>	XL
<b>G.1. Expert evaluators .....</b>	XL
<b>G.2. Evaluation of functional requirements .....</b>	XL
<b>G.3. Evaluation of non-functional requirements.....</b>	XLII
<b>APPENDIX H – CONCLUSION.....</b>	XLIV
<b>H.1. Status of Research Objectives .....</b>	XLIV

<b>H.2. Achievement of Learning Outcomes .....</b>	<b>XLVI</b>
<b>H.3. Project Plan .....</b>	<b>XLVIII</b>
<b>APPENDIX I – CONCEPT MAP .....</b>	<b>L</b>
<b>APPENDIX J – REVIEW PAPER .....</b>	<b>LI</b>
<b>APPENDIX K – RESEARCH PAPER .....</b>	<b>LIX</b>
<b>APPENDIX L – PROOF OF SUBMISSION.....</b>	<b>LXV</b>
.....	

## LIST OF TABLES

Table 1: Research Objectives (Self-Composed) .....	5
Table 2: Comparison of Text Summarization Techniques (Self-Composed) .....	13
Table 3: Evaluation metrics for abstractive text summarization (Self-Composed) .....	23
Table 4: Comparison table for abstractive text summarization using transformers (Etemad, Abidi and Chhabra, 2021).....	25
Table 5: Research Methodology (Self-Composed).....	26
Table 6: Deliverables & Dates (Self-Composed).....	28
Table 7: Risk Management Plan (Self-Composed).....	30
Table 8: Stakeholder Viewpoints & Requirements (Self-Composed) .....	33
Table 9: Requirement Elicitation Methodologies (Self-Composed).....	34
Table 10: Literature Review Findings (Self-Composed) .....	35
Table 11: Observations Findings (Self-Composed).....	36
Table 12: Survey Analysis (Self-Composed) .....	38
Table 13: Survey Thematic Analysis (Self-Composed) .....	41
Table 14 Competitor Analysis (Self-Composed).....	42
Table 15: Prototyping Findings (Self-Composed).....	43
Table 16: Summary of Findings (Self-Composed) .....	44
Table 17: Use Case Description Uc:07 (Self-Composed).....	46
Table 18: Use Case Description Uc:03 (Self-Composed).....	47
Table 19: Functional Requirements (Self-Composed) .....	48
Table 20: Non-Functional Requirements (Self-Composed) .....	50
Table 21: SLEP Issues & Mitigation .....	51
Table 22: Dataset Sources (Self-Composed) .....	59
Table 23: Development Framework Utilized (Self-Composed) .....	60
Table 24: Libraries Used with Reasonings (Self-Composed).....	60
Table 25: Ide's Used Along with Justifications (Self-Composed) .....	61
Table 26: Summary of Technology Selection (Self-Composed) .....	61
Table 27: Model Evaluation Results.....	72
Table 28: Benchmarking results .....	74
Table 29: Module & integration testing.....	75
Table 30: Evaluation criteria.....	77
Table 31: Self-Evaluation of The Author .....	78
Table 32: Categorization of selected evaluators .....	79
Table 33: Thematic analysis of expert feedback.....	80
Table 34: Utilization of Knowledge Gained from The Course .....	83
Table 35: Mitigations to Problems and Challenges Faced .....	85
Table 36: Related Work in Abstractive Text Summarization .....	IX
Table 37: Stakeholder groups ( <i>Self-Composed</i> ).....	I
Table 38: Interview Thematic Analysis (Self-Composed).....	I
Table 39: Interview participant details.....	II
Table 40: Interview evidence.....	III
Table 41: Usecase mappings ( <i>Self-Composed</i> ).....	III
Table 42: Use case description UC:01 ( <i>Self-Composed</i> ) .....	IV

Table 43: Use case description UC:02 ( <i>Self-Composed</i> ) .....	IV
Table 44: Use case description UC:10 ( <i>Self-Composed</i> ) .....	V
Table 45: Use case description UC:04 ( <i>Self-Composed</i> ) .....	VI
Table 46: Use case description UC:05 ( <i>Self-Composed</i> ) .....	VI
Table 47: Use case description UC:06 ( <i>Self-Composed</i> ) .....	VII
Table 48: Use case description UC:08 ( <i>Self-Composed</i> ) .....	VIII
Table 49: Use case description UC:09 ( <i>Self-Composed</i> ) .....	VIII
Table 50: ‘MoSCoW’ technique of requirement prioritization.....	IX
Table 51: Design Goals of The Proposed System ( <i>Self-Composed</i> ) .....	X
Table 52: Selection of Data Science Language.....	XVIII
Table 53: Selection of DL framework .....	XIX
Table 54: Selection of UI framework .....	XX
Table 55: Selection of web framework .....	XX
Table 56: Functional Testing .....	XXXIII
Table 57: Non-functional testing .....	XXXIX
Table 58: Details of the expert evaluator(s) selected. ....	XL
Table 59: Evaluation of the implementation of functional requirements.....	XL
Table 60: Evaluation of the implementation of non-functional requirements .....	XLII
Table 61: Evaluation of the achievement of design goals.....	XLIII
Table 62: Status of research objectives.....	XLIV
Table 63: Achievement of learning outcomes .....	XLVI

## LIST OF FIGURES

Figure 1: Transformer Architecture Downloads Rate (Wolf et al., 2020) .....	15
Figure 2: Proposed Generalized Abstractive Summarization System Process Flow (Self-Composed) – <i>(view high qual version)</i> .....	20
Figure 3: Rich Picture Diagram (Self-Composed) – <i>(view high qual version)</i> .....	31
Figure 4: Stakeholder Onion Model (Self-Composed) – <i>(view high qual version)</i> .....	32
Figure 5: Context Diagram (Self-Composed).....	45
Figure 6: Use Case Diagram (Self-Composed).....	46
Figure 7: Three-Tiered Architecture (Self-Composed).....	52
Figure 8: Data Flow Diagram - Level 01 (Self-Composed).....	55
Figure 9: Data Flow Diagram - Level 02 (Self-Composed).....	56
Figure 10: System Process Flow Chart (Self-Composed) – <i>(view high qual version)</i> .....	57
Figure 11: Technology Stack (Self-Composed).....	58
Figure 12: Model customization parameter initialization .....	62
Figure 13: Training hyper-parameter initialization.....	63
Figure 14: Model decoder customization .....	63
Figure 15: Model training hyperparameter tuning.....	64
Figure 16: Hyperparameter Results and Training Arguments (Self-Composed).....	64
Figure 17: Model Training (Self-Composed) .....	65
Figure 18: General User Review Text Summarization (Self-Composed).....	65
Figure 19: Assigning A Specific Model for The New Domain User (Self-Composed).....	66
Figure 20: Domain Specific Text Review Summarization (Self-Composed) .....	67
Figure 21: Fetching Related Data for Model Retraining (Self-Composed).....	68
Figure 22: Validation Accuracy by number of epochs – bart model ( <i>Self-Composed</i> ) .....	71
Figure 23: Validation Loss by number of epochs – bart model ( <i>Self-Composed</i> ) .....	71
Figure 24: Validation Accuracy by number of epochs – t5 model ( <i>Self-Composed</i> ) .....	71
Figure 25: Validation Loss by number of epochs – t5 model ( <i>Self-Composed</i> ) .....	71
Figure 26: Validation Accuracy by number of epochs – Pegasus model ( <i>Self-Composed</i> ) .....	72
Figure 27: Validation Loss by number of epochs – Pegasus model ( <i>Self-Composed</i> ) .....	72
Figure 28: Hotel Domain (Cinnamon Type) Evaluation.....	73
Figure 29: Movie Domain (Scope Cinema Type) Evaluation .....	73
Figure 30: Prototype Feature Diagram (Self-Composed).....	VII
Figure 31: Homepage Wireframe (Self-Composed).....	XI
Figure 32: Review History Page (Self-Composed).....	XII
Figure 33: User Profile Page (Self-Composed) .....	XIII
Figure 34: CNN Dailymail Dataset ( <i>view</i> ) .....	XIV
Figure 35: Gigaword Dataset ( <i>view</i> ).....	XV
Figure 36: Xsum Dataset ( <i>view</i> ) .....	XVI
Figure 37: Amazon Movie Reviews Dataset ( <i>view</i> ). ....	XVII
Figure 38: Hotel Reviews Dataset ( <i>view</i> ) .....	XVIII
Figure 39: Preprocessing: Remove Markdown (Self-Composed) .....	XXI
Figure 40: Preprocessing – Remove Hyperlinks (Self-Composed) .....	XXI
Figure 41: Preprocessing: Remove Html Tags (Self-Composed) .....	XXII

Figure 42: Preprocessing: Char Words Extension (Self-Composed) .....	XXII
Figure 43: Preprocessing: Handling Common Contractions (Self-Composed) .....	XXIII
Figure 44: Preprocessing: Removing Special Characters (Self-Composed) .....	XXIV
Figure 45: Preprocessing: Resolving Spelling Mistakes (Self-Composed).....	XXIV
Figure 46: Preprocessing: Removing Duplicates (Self-Composed) .....	XXV
Figure 47: Preprocessing: Restoring Missing Punctuations (Self-Composed) .....	XXV
Figure 48: Preprocessing: Grammarly Correction (Self-Composed).....	XXV
Figure 49: GUI – Homepage (After Signup) (Self-Composed).....	XXVII
Figure 50: GUI – Review History Page (Self-Composed) .....	XXX
Figure 51: GUI – User Profile (Self-Composed).....	XXXI
Figure 52: GUI – Model Retraining Modal (Self-Composed) .....	XXXI
Figure 53: GUI – Homepage (Before Signup) (Self-Composed) .....	XXXII
Figure 54: System Resource Manager Graph (Self-Composed).....	XXXVI
Figure 55: Lighthouse Landing Page .....	XXXVII
Figure 56: Lighthouse Records Page .....	XXXVII
Figure 57: Lighthouse User Profile Page.....	XXXVII
Figure 58: CodeFactor - Gensum Repository .....	XXXVIII
Figure 59: CodeQL - Gensum Repository .....	XXXVIII
Figure 60: Gantt Chart: Initial Plan (Self-Composed).....	XLVIII
Figure 61: Gantt Chart: Final Plan (Self-Composed) .....	XLIX
Figure 62: Research & Review Paper Submission .....	LXV

## LIST OF ABBREVIATIONS

<b>AI</b>	Artificial Intelligence.
<b>DL</b>	Deep Learning
<b>GUI</b>	Graphical User Interface
<b>ML</b>	Machine Learning
<b>NLP</b>	Natural Language Processing
<b>ROUGE</b>	Recall-Oriented Understudy for Gisting Evaluation.
<b>BLEU</b>	BiLingual Evaluation Understudy.
<b>T5</b>	Text to Transfer Transformer.
<b>BART</b>	Bidirectional Auto-Regressive Transformers.
<b>BERT</b>	Bidirectional Encoder Representations from Transformers.
<b>PEGASUS</b>	Pre-training with Extracted Gap-sentences for Abstractive Summarization Sequence-to-sequence
<b>ILP</b>	Inductive logic programming.
<b>LSTM</b>	Long Short-Term Memory.
<b>RNN</b>	Recurrent Neural Network.
<b>CNN</b>	Convolutional Neural Network.
<b>SEQ2SEQ</b>	Sequence to Sequence
<b>RoBERTa</b>	Robustly Optimized BERT Pre-training Approach
<b>GPT-3</b>	Third Generation Generative Pre-Trained Transformer
<b>REST</b>	Representational State Transfer
<b>GPU</b>	Graphical Processing Unit
<b>API</b>	Application Programming Interface

# CHAPTER 01. INTRODUCTION

---

---

## 1.1 Chapter Overview

---

In this research project, the author experiments and optimizes a series of top-tier pretrained transformer designs by automating the process of model architecture customization and search hyperparameter optimization in an effort to improve the performance of abstractive text summarization for movie reviews while developing a generalized solution that can adapt and be used in other domains.

This chapter presents a clear definition of the problem, research gap, research challenge, and the research strategy that will guide the author's work over the upcoming months. In addition, the chapter provides an overview of the necessary evidence to substantiate the problem, as well as a review of relevant prior research.

## 1.2 Problem Domain

---

### 1.2.1 Movie User Reviews

In recent times, an increasing number of websites such as Amazon and the Internet Movie Database (IMDB), which is a platform dedicated to movie reviews, permit their users to share their feedback on subjects that intrigue them. This trend has been concurrent with the emergence of Web 2.0, a framework that prioritizes user engagement and participation (Khan, Gul, Zareei, et al., 2020).

According to Khan, Gul, Uddin, et al. (2020), online movie reviews are becoming an increasingly important source of information for users, particularly given the vast amount of data available on the web. Despite this, the sheer volume of movie reviews posted by online users on a daily basis can make it difficult for individuals to manually summarize reviews and assess their interest in a particular film. Consequently, one of the most challenging problems in natural language processing involves the mining and summarization of movie reviews.

Alsaquer and Sasi (2017) state that text summarization is a useful tool for assisting users and business decision-makers in compiling and analyzing a large number of online reviews. Nowadays, the vast majority of people read reviews of films before selecting or watching them on platforms such as Netflix or Amazon Prime.

However, users may encounter conflicting reviews that are either positive or negative, and reading all the reviews in detail can be time-consuming. As a result, summarizing reviews can make it faster and easier for users to make decisions. Additionally, this can help streaming services like Netflix to rapidly identify the viewing habits or preferences of their users, as noted by Dashtipour et al. (2021).

### **1.2.2 Text Summarization**

Mahajan et al. (2021) suggest that in the present day, there is a vast amount of textual content accessible to us, such as news articles and reviews. Text summarization plays a critical role in facilitating the process of extracting crucial information from such pieces of text by minimizing the volume of text to be perused.

Etemad, Abidi and Chhabra (2021) explain that there are two primary methods for text summarization: extractive summarization and abstractive summarization. Extractive summarization involves selecting the most important sentences from the context or article without making any alterations to them. Conversely, abstractive summarization aims to generate its own sentences and create a summary; this approach is more effective than extractive summarization as it allows for the creation of more meaningful phrases within the context, rather than relying on selected sentences from the context without modification.

### **1.2.3 Transformers**

The Transformer architecture in Natural Language Processing (NLP) is a new and innovative model designed to handle sequence-to-sequence tasks and effectively deal with long-range dependencies. Its performance has exceeded that of other neural models, such as Convolutional Neural Nets (CNN) and Recurrent Neural Nets (RNN), and has emerged as the leading architecture in NLP. (Wolf et al., 2020).

According to Etemad, Abidi, and Chhabra (2021), Transformers adopt a self-attention mechanism to focus on specific parts of the input sequence, followed by the encoder and decoder architecture.

## 1.3 Problem Definition

---

In the field of movie review summarization, current research has not explored optimizing the state-of-the-art deep learning methods, including *transformers*, to address this issue. Previous studies have relied on conventional machine and deep learning algorithms like naïve Bayes and RNN. However, leveraging advanced deep learning approaches may improve the quality and accuracy of text summarization.

Deep learning models require more time for training, but they offer higher accuracy as they can automate both feature extraction and classification simultaneously. On the other hand, machine learning algorithms necessitate feature selection initially. Consequently, the implementation of deep learning techniques can help enhance the precision of text summarization and enable users to make informed decisions (Etemad, Abidi, & Chhabra, 2021).

### 1.3.1 Problem statement

To date, there has been no investigation into optimizing and utilizing the state-of-the-art deep learning techniques, such as transformers, to generate abstractive summaries from movie reviews, which can enhance text summarization. This approach intends to be all-encompassing and available to any industry. (Khan, Gul, Zareei, et al., 2020).

## 1.4 Research Motivation

---

The problem of improving the quality of abstractive text summarization using advanced deep learning techniques is not limited to movie reviews alone but can also be applied to various other domains. Hence, a **generalized solution** was initially proposed to address this issue. (Kouris, Alexandridis, & Stafylopatis, 2019).

In previous studies, scholars such as (Etemad, Abidi and Chhabra, 2021) have identified syntactic and semantic challenges as the main obstacles to effective text summarization. In their research, they explored several deep learning techniques and found that transformer-based models, particularly the T5 model, outperformed other models in various natural language processing tasks. This finding encourages further investigation into the optimization of transformer models to improve the quality of text summarization and overcome the challenges associated with summarizing movie reviews.

## 1.5 Research Questions

---

**RQ1:** What are the top tier transformer architectures widely used and known for NLP problems related to text summarization?

**RQ2:** How can a pretrained transformer architecture be fine-tuned to get the optimal hyperparameters and automate model customization?

**RQ3:** What kind of evaluations should we perform after fine-tuning to filter out the best transformer architecture?

**RQ4:** 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 automating the processing of model customization and hyperparameter tuning for optimization, therefore obtaining the recommended architecture's optimum performance*

The aim of this research project is to develop a functional system capable of conducting abstractive text summarization from user input in various domains, including movies, hotels, and ecommerce. The main emphasis will be placed on the quality of the text summary and performance optimization. The research will investigate various techniques, including data preparation, data analysis, hyperparameter tuning, model customization and model evaluation, to achieve the best results.

The study aims to construct the essential components, collect and analyze necessary information, and evaluate the performance of the system. The system will be designed to be accessible for private or public use on both hosted servers and local browsers. The source code of the data science models will be publicly available in a repository to facilitate further research and practical use. A research paper will be published as the outcome of this investigation.

## 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
Problem Identification	<p>Comprehend and document the identified issue.</p> <p><b>RO1:</b> Perform research in a domain of interest and identify a sufficiently comprehensive problem that needs to be addressed.</p> <p><b>RO2:</b> Thoroughly explore and analyze potential solutions for addressing the problem.</p> <p><b>RO3:</b> Explore methods for developing an adaptive and generalized approach.</p> <p><b>RO4:</b> Create a schedule, determine associated deliverables, and develop a Gantt chart for the project.</p>	LO1, LO2	RQ1, RQ4
Literature Review	<p>Complete a thorough critical review of earlier related work.</p> <p><b>RO1:</b> Make a preliminary investigation on existing abstractive text summarization using deep learning approaches.</p> <p><b>RO2:</b> Make a preliminary investigation on why transformers architecture was the chosen deep learning choice for this research.</p> <p><b>RO3:</b> Analyze the top tier transformer architectures widely used.</p> <p><b>RO4:</b> Analyzing how the models can be fine-tuned via hyperparameter optimization &amp; perform model customization.</p>	LO1, LO4, LO8	<span style="background-color: #00AEEF; border: 1px solid black; padding: 2px;">RQ1,</span> <span style="background-color: #00AEEF; border: 1px solid black; padding: 2px;">RQ2,</span> <span style="background-color: #00AEEF; border: 1px solid black; padding: 2px;">RQ3,</span> <span style="background-color: #00AEEF; border: 1px solid black; padding: 2px;">RQ4</span>

	<p><b>RO5:</b> Analyzing the different approaches used for model evaluation.</p> <p><b>RO6:</b> Analyze how the model can be generalized for every other domain.</p>		
Methodology Selection and SLEP Framework	<p>This defines the outline structure for the requirement analysis and the design process followed by the social legal ethical and professional issues.</p> <p><b>RO1:</b> Analyzing the Research Methodology approaches.</p> <p><b>RO2:</b> Analyzing the Development Methodology approaches.</p> <p><b>RO3:</b> Analyzing the Project Management Methodology approaches.</p> <p><b>RO4:</b> Analyzing the Solution Methodology approaches.</p> <p><b>RO5:</b> Analyzing the Social, Legal Ethical and Professional Issues which could develop during the phase of the project.</p>	LO2, LO6	<b>RQ4,</b> <b>RQ2,</b> <b>RQ1</b>
Requirement Elicitation	<p>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.</p> <p><b>RO1:</b> Gathering information related to the expected metadata required for the dataset to contain for the model training.</p> <p><b>RO2:</b> Gathering the requirements of transformer architectures for fine-tuning and understand the end to end user expectations.</p> <p><b>RO3:</b> Getting insights from domain experts to build a suitable system.</p> <p><b>RO4:</b> Gathering the requirements for handling generalization.</p>	LO1, LO3, LO5	<b>RQ4,</b> <b>RQ2,</b> <b>RQ1</b>

Design	<p>Considering the following when developing the suggested system:</p> <p><b>RO1:</b> Design a component to preprocess the dataset for the respective model inputs.</p> <p><b>RO2:</b> Design a component to store the top tier transformer models with their respective metadata, to use throughout.</p> <p><b>RO3:</b> Design a hyperparameter tuning component that can improve accuracy of the transformer model &amp; customize model architecture.</p> <p><b>RO4:</b> Design high-level architecture for the system.</p>	LO1, LO5	<b>RQ2</b>
Implementation	<p>Setting up a mechanism capable of addressing the gaps that were intended to be covered.</p> <p><b>RO1:</b> To develop data preprocessing component.</p> <p><b>RO2:</b> To develop a component that handles and stores the top tier transformer architectures for fine-tuning.</p> <p><b>RO3:</b> To develop the automated model customization &amp; hyperparameter search component that handles all the top tier architectures assigned.</p> <p><b>RO4:</b> To develop a component for the model evaluations for the measured hyperparameters</p>	LO1, LO5, LO7	<b>RQ2,</b> <b>RQ3</b>
Evaluation	<p>Testing and evaluating the developed system (including the data science models with the suitable metrics)</p> <p><b>RO1:</b> Performing unit test, integration and performance testing along with a test plan created.</p>	LO1, LO5	<b>RQ3</b>

	<b>RO2:</b> Evaluating all the transformer architectures used for fine-tune experimentations, using recommended scores such as (ROUGE or BLEU SCORE).		
Documentation	Keeping track of and documenting the study project's ongoing progress and any challenges encountered.	LO6, LO8	-
Publication	<p>Ensure that the documentation, reports, and papers are well-structured and include a critical analysis of the research.</p> <p><b>RO1:</b> To publish a review paper on the related work done.</p> <p><b>RO2:</b> To publish a research paper on the related work done.</p> <p><b>RO3:</b> To publish the code implementation repository as public to be access by future research investigations, along with the models and datasets</p>	LO4, LO8	-

## 1.7 Novelty of the Research

---

### 1.7.1 Problem Novelty

The novelty of the problem being addressed in this research lies in the absence of efforts to enhance the performance of transformers for improved text summarization results, as pointed out by Khan, Gul, Zareei et al. (2020).

### 1.7.2 Solution Novelty

The novelty of this proposed solution lies in its automated approach to model customization & hyperparameter tuning along with the incorporation of a retraining mechanism with newly available data to further enhance the performance of the optimal transformer model for any domain, thus achieving generalization. The approach proposed in this study is considered novel as it has not been previously explored in existing research (Etemad, Abidi, & Chhabra, 2021).

## 1.8 Research Gap

---

The literature suggests a requirement for advanced deep learning techniques to improve the performance of text summarization for movie reviews, as identified from previous research (Khan,

Gul, Zareei, et al., 2020). Traditional machine learning methods have been found to be less effective in comparison to advanced deep learning techniques in the domain of movie review summarization.

This project focuses on empirical gap in the movie domain, as well as theoretical and performance gaps in the area of transformer optimization. Transformers have emerged as a vital component in the field of deep learning for addressing problems related to natural language processing. The optimization of the model architecture & hyperparameters on various transformer architectures could aid in enhancing the quality of abstractive text summarization. Additionally, developing a generalized model adaptable to the particular domains' requirements can further enhance performance.

## 1.9 Contribution to the body of knowledge

---

The contributions of this project can be distinguished into two categories, namely theoretical contributions and domain-specific contributions.

The following is a summarization of the authors contribution:

- **Abstractive Text Summarization:** Automated (model architecture & 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

According to previous research conducted by Zhang, Xu, and Wang (2019), there are numerous deep learning techniques available for handling abstractive text summarization. However, transformers appear to outperform most other deep learning approaches, as identified by previous research. Despite this, there has been little research into optimizing transformers for even better performance.

The primary objective of this study is to develop an adaptive generalized solution by optimizing the transformer architecture through fine-tuning and automated model customization & hyperparameter optimization of top-tier existing architectures in the domain of abstractive text

summarization. According to Liu and Wang (2021), the proposed approach can enhance the performance of the recommended architecture.

### **1.9.2 Problem Domain Contribution**

Mahajan et al. (2021) have noted that neural networks form the foundation of deep learning algorithms, allowing them to handle intricate unstructured data better than traditional machine learning algorithms. However, previous research has failed to investigate the potential of utilizing advanced deep learning approaches for summarizing movie reviews.

The solution proposed for movie review summarization involves using optimizing transformers. This approach has shown promising results in the field of natural language processing. The solution will be adaptable and applicable to other text summarization tasks beyond the movie review domain.

### **1.10 Research Challenge**

---

The primary aim of this study is to attain a generalized and optimal transformer architecture for abstractive text summarization. In the field of NLP, transformers have become the most preferred option for tackling NLP problems, as they have replaced RNN models. However, since transformers were introduced relatively recently in 2017 by a team at Google Brain, there is still a shortage of research on optimizing them specifically for the task of abstractive text summarization. (Wolf et al., 2020).

Hence, developing and identifying the optimal transformer architecture and parameters for text summarization while ensuring generalization poses a significant challenge, especially considering the limited resources available for research in this area. Furthermore, selecting appropriate datasets for movie review summarization and generalization is a difficult task since it requires significant effort in data preprocessing & experimenting.

### **1.11 Chapter summary**

---

In this chapter, the author describes their research endeavor, highlights its novelty and significance, and acknowledges the potential challenges that may arise during the process. Moreover, the chapter outlines the essential objectives that must be achieved for the research to be considered successful and links them to the required learning outcomes for the degree.

## CHAPTER 02. LITERATURE REVIEW

---

### 2.1 Chapter Overview

---

In this chapter, the author critiques prior studies that have utilized abstractive text summarization techniques, specifically deep learning approaches such as transformers, for summarizing movie reviews. Additionally, the author proposes an adaptable and generalizable solution capable of handling a variety of domains beyond just movies. Finally, through domain generalization, the author fine-tunes the transformer model and adapts it to new data and customize the model & hyperparameters, ultimately identifying the optimal design with the most favorable results.

### 2.2 Concept Map

---

The project scope of the literature review is presented in a concept map, where the main study areas are illustrated as nodes. The concept map serves as a visual aid to guarantee the inclusion of all pertinent literature and can be found in **APPENDIX I**.

### 2.3 Problem Domain

---

The advancement of technology and the widespread adoption of the internet have simplified the process of selling products or services to customers. By leveraging customer feedback, sellers can make informed decisions on how to enhance sales and achieve higher levels of customer satisfaction (Boorugu, Ramesh, & Madhavi, 2019). However, in the context of the film industry, customers face difficulties in rapidly assessing whether a movie aligns with their preferences by relying solely on reviews, which can be lengthy and time-consuming to read (Khan et al., 2020).

#### 2.3.1 User Reviews

A user/customer review is commonly defined as written feedback provided by a customer who has utilized a product or service. These reviews are often used by consumers to inform their purchasing decisions. However, the unstructured nature of review data can make it challenging for consumers to compare and comprehend longer reviews (Lackermair, Kailer, & Kanmaz, 2013).

In industries such as tourism and hospitality, user and customer reviews hold great significance as they serve as a major driver for a country's economic growth and development.

Tourists from various parts of the world share their experiences and provide feedback online in various formats, such as blogs and reviews (Mukherjee et al., 2020).

### **2.3.2 Corporate Advantage**

Maintaining customer loyalty is crucial for businesses since acquiring a new customer typically requires at least five times the time and resources compared to retaining an existing one. Customer satisfaction plays a pivotal role in the survival of corporate industries, and analyzing customer feedback and reviews is an effective method to gain insights into their expectations and rectify any issues (Pizam & Ellis, 1999).

Streaming services such as Netflix and Amazon Prime can leverage movie summaries to better comprehend user viewing patterns and interests. In the same vein, industries related to movies should enable customers to quickly assess a summary and make prompt decisions on whether or not to watch a movie (Khan et al., 2020).

### **2.3.3 Text Summarization**

In today's digital age, there is an enormous amount of textual data available on the internet, making it challenging to extract relevant information from a large number of documents. Text summarization aims to provide a condensed and meaningful version of lengthy textual content (Shi et al., 2020).

This technique has various applications in internet-based fields, such as search engines that use it for querying and e-commerce sites that utilize sentiment analysis to assess customer satisfaction with products (Etemad, Abidi and Chhabra, 2021).

In the movie industry, text summarization can simplify customer reviews of movies, which are often lengthy and time-consuming to read. This simplification can enable users to make better decisions when deciding whether or not to watch a particular movie (Khan et al., 2020).

### **2.3.4 Abstractive and Extractive Techniques**

In general, text summarization can be divided into two categories: abstractive text summarization and extractive text summarization. However, it is feasible to create a hybrid model for text summarization (Alsaqer and Sasi, 2017). The abstractive technique of text summarization intends to generate its own sentences and utilize them to create a consistent summary. Consequently, the summary's substance may differ from the original text but still communicate the same concept

(Mahajan et al., 2021). Furthermore, it is widely recognized that a solid abstractive summary involves the primary information from the input and is linguistically coherent (Zhang et al., 2020).

The technique of extractive text summarization involves identifying essential phrases or groups of phrases from the original input content and merging them to create a brief but informative text summary. The selection of sentences to include in the summary is based on the statistical and linguistic features of those sentences. In contrast, abstractive text summarization aims to generate sentences independently and utilize them to produce a coherent summary that may differ from the original context while conveying the same message. A hybrid text summarization system combines different methods to create a unified system. Combining extractive and abstractive summarization strategies can lead to a hybrid system that employs encoder-decoders, as demonstrated in prior research (Kirmani et al., 2019; Abolghasemi, Dadkhah and Tohidi, 2022).

Table 2: Comparison of Text Summarization Techniques (Self-Composed)

<b>Abstractive</b>	<b>Extractive</b>
According to Mahajan et al. (2021), the ability of an AI system to paraphrase content in a manner similar to humans involves the generation of a new context. In other words, the AI system can create its own context, just as humans do.	The text summarization approach that does not create its own context but selects the most suitable phrases from the original document is the one identified by Gupta and Lehal (2010).
There exists a plethora of datasets that can be used for experimentation in this field.	It has the ability to visualize the scores of individual sentences and explore gradient-based techniques for computing the contribution of each input token to score prediction (Pai, 2014).
There is a possibility of generating misinformation or producing a meaning that differs from the original text.	There is a potential for errors in the combined sentences formed from the extracted sentences.

### 2.3.5 NLP with Deep Learning

Natural Language Processing (NLP) is a computer-based method that enables effective and intelligent analysis, comprehension, and understanding of human language, distinguishing it from other methods that only consider the interactions between human language and computers. In recent times, deep learning techniques have been increasingly used in AI research due to their superior performance in handling complex and high computing learning tasks compared to traditional machine learning approaches (Lopez and Kalita, 2017; Mahajan et al., 2021).

Although machine learning is widely used in NLP today, it mostly involves optimizing the weights of features and representations that are created by humans. In contrast, deep learning is aimed at exploring how computers can leverage data to create features and representations suitable for complex interpretation tasks (Socher, Bengio, and Manning, 2012).

### 2.3.6 Transformers

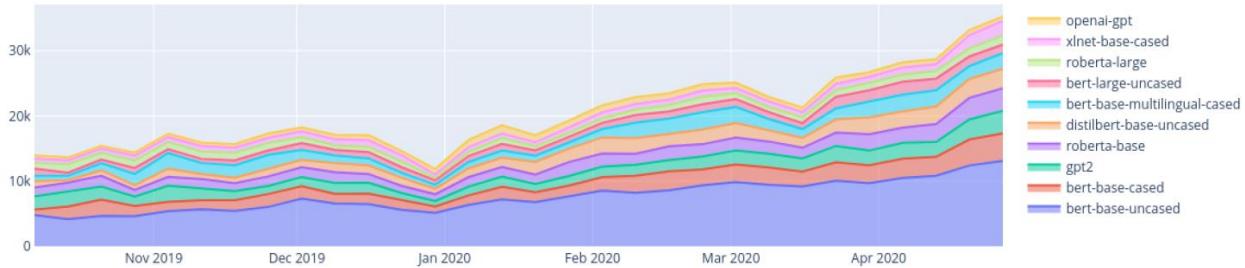
In the field of natural language processing (NLP), the open-source library Transformers provides modern transformer architectures that are thoroughly developed and integrated through a common API. Pretraining has made these architectures efficient for various tasks, enabling the construction of higher-capacity models.

These transformer models have been designed to be accessible for practitioners, extensible for researchers, and fast and reliable for industrial applications (Wolf et al., 2020).

Recent studies have demonstrated the high competence of modern pre-trained language models based on transformers in identifying both syntactic and semantic cues, including noun modifiers, possessive pronouns, prepositions, co-referents, entities, and relations (Brasoveanu and Andonie, 2020).

Hugging Face Hub provides a range of transformer designs, such as BERT, GPT2, T5, PEGASUS, and others. As shown in the figure below, the daily average of unique downloads for pretrained transformer model architectures has been steadily increasing between October 2019 and May 2020 (Wolf et al., 2020).

Figure 1: Transformer Architecture Downloads Rate (Wolf et al., 2020)



(Etemad, Abidi and Chhabra, 2021) conducted research that involved comparing various approaches taken by researchers to perform abstractive text summarization. These techniques included the use of transformers, as well as other neural network approaches such as CNN and LSTM RNN networks. The comparison table presented in the research only focuses on the approaches using transformers for abstractive text summarization.

### 2.3.7 Model customization & hyperparameter tuning

Hyperparameter tuning is the process of finding the most suitable set of parameter values for training an algorithm to create a model relevant to a given dataset (Liu and Wang, 2021). The tuning of hyperparameters is crucial to improving the performance of the algorithm by assessing the improvement in performance that can be achieved by altering each of the hyperparameters from its initial value to a target value determined by the tuning strategy (Joy and Selvan, 2022).

The selection of certain hyperparameters such as learning rate, weight decay, number of epochs, batch size, and warmup ratio can significantly improve model performance during the training phase. Meanwhile, other parameters such as decoder attention heads, decoder layerdrop, and decoder layers control the model's architecture. It is important to prioritize the most critical hyperparameters. Automated framework tools like Optuna can make hyperparameter optimization easier by simplifying the application of techniques such as Grid Search, Random Search, TPE, and CMA-ES algorithms. (Joy and Selvan, 2022).

### 2.3.8 Generalization

In various fields, generalization has become an important technique for addressing problems that are related to the same issue. The ability of a model to generalize to new data that has not been observed before and comes from the same distribution as the model's original data is known as generalization (Neyshabur et al., 2017).

Generalization is a valuable approach that enables the development of foundational knowledge and allows for further refinement and specialization as new, unseen domain data becomes available. As a result, a generalized solution has the potential to adapt to unseen domain data, making it a viable solution for resolving a common problem across multiple domains (Zhou et al., 2021).

### **2.3.9 Data Expansion**

The effectiveness of a machine learning or deep learning model is influenced by multiple factors, such as the quantity and quality of the data utilized for model training. Various techniques are available to enhance the amount and diversity of data, including data augmentation, which involves creating new data points from existing ones. Additionally, obtaining new data points from user interactions during model usage can also expand the available dataset for retraining (Shorten and Khoshgoftaar, 2019). In cases where a generalized model needs to be adapted to a specific domain, retraining with new data from that domain can be utilized.

## **2.4 Existing work**

---

The domain of movie reviews has been the subject of various studies in abstractive text summarization, primarily relying on conventional machine learning techniques. Nonetheless, these methods have encountered certain constraints, necessitating the use of more recent deep learning methods to enhance the system's efficiency.

### **2.4.1 Text Summarization Systems**

#### **2.4.1.1 Abstractive Approach**

The research conducted by Boorugu, Ramesh, and Madhavi (2019) is focused on ecommerce, but it is still relevant to text summarization for customer reviews on products. The goal is to aid potential customers in making better purchasing decisions by reducing the time-consuming process of going through all the reviews. The researchers chose to use an abstractive approach to summarization, which they deemed to be a more appropriate choice for their purposes.

The research conducted by Gupta et al. (2021) is a comprehensive comparison study of various pretrained transformer architectures including BART, BERT, T5, and PEGASUS for abstractive text summarization. The study employs an abstractive approach and uses various types

of datasets to explore each model and evaluate their performance. The authors conclude that T5 is the best performing transformer architecture based on their evaluation results.

In their study, Mahajan and colleagues (2021) employ an abstractive approach to text summarization that utilizes deep learning techniques, specifically RNNs, to ensure proper grammar and avoid repetition of words in the generated summary. Similarly, Etemad, Abidi, and Chhabra (2021) conduct an experimental study to evaluate various deep learning approaches for abstractive text summarization. The goal of their study is to identify the best approach to the problem, as determined by benchmarking evaluations.

#### **2.4.1.2 Extractive Approach**

In the past, several studies have been conducted in the field of text summarization, which include both extractive and abstractive text summarization techniques. Khan et al. (2020) conducted research on the topic of summarizing movie reviews, which is also the focus of this project. The study involved the development of an automatic approach to summarize lengthy movie reviews and enable users to quickly identify the positive and negative aspects of the movie based on the processed review. The approach taken by this author was extractive, and sentence score ranking played a crucial role in generating the summary.

In the study conducted by Mukherjee et al. (2020), an extractive approach was employed to develop a solution for creating personalized aspect-based opinion summaries from a vast dataset of online tourist reviews. The author also introduced personalization features to the summary generation process by incorporating user interests. However, the use of abstractive summarization is deemed to be more effective but also challenging when customizing user interests, as it involves creating sentences using unique words instead of relying on sentence ranking techniques.

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

As indicated in the Problem Domain section of this literature review, deep learning methods are generally preferred over traditional machine learning methods due to their ability to handle highly computational tasks. The author has encountered several deep learning techniques, as well as machine learning techniques, for addressing the challenge of abstractive text summarization.

The prioritization of deep learning approaches over traditional machine learning approaches for highly computational tasks is emphasized in the Problem Domain section of the literature review, as outlined in (Khan et al., 2020). The author discusses several deep learning

techniques used alongside traditional machine learning techniques for abstractive text summarization. However, the study's focus on using only standard machine learning approaches, such as the Naive Bayes method and an undirected weighted graph-based ranking algorithm for review classification, limits the use of sophisticated deep learning algorithms that could improve performance.

In contrast, Boorugu, Ramesh, and Madhavi (2019) employed a seq2seq model for text summarization, utilizing the attention mechanism for improved accuracy and the Concept net Number batch word embedding model, which is superior to Glove. The authors also used a 1D convolutional layer, max pooling layer, LSTM layer, and fully connected layer to improve performance. However, the use of generic deep learning algorithms in this study introduced a constraint that prevented performance from being improved using more recent deep learning strategies for NLP-related problems, such as transformers.

As previously mentioned, the research by Mukherjee et al. (2020) utilizes an extractive method for text summarization based on integer linear programming (ILP [Unsupervised method]) to select an informative subset of opinions based on identified aspects. The study employs ROUGE-based criteria to assess and compare the summaries, resulting in competitive outcomes. However, due to the dataset's limitations, extractive summaries may not be particularly insightful, and utilizing an abstractive technique could produce better results.

Lastly, the study by Mahajan et al. (2021) focuses on utilizing an encoder-decoder model with an attention layer to produce text summaries with proper syntax and no repeated words. The authors developed an encoder-decoder model with gated recurrent units and trained it to provide an abstract summary of a piece of writing. Although deep learning was employed, its application in production required real-time training to update the model with the latest content over time.

### **2.4.3 Usage of Transformers**

In their research, Gupta et al. (2021) conducted a comparison study of various pretrained transformer models, including Pipeline BART, modified BART, T5, and PEGASUS, for text summarization using ROUGE scores as evaluation measures. Although the author employed transformer designs, the hyperparameters used were default and may benefit from further tuning for improved performance. The study focuses on developing more robust models that can be

expanded to produce summaries of varying lengths and applied for multi-document summarization.

Etemad, Abidi, and Chhabra (2021) conducted an experimental study on deep learning methods, such as RNN, CNN, and Transformers, in the broad domain of text summarization to determine which method performs best. The study considers model evaluation metrics such as BLEU and ROUGE. Despite the use of sophisticated deep learning algorithms, the authors did not undertake hyperparameter tuning to improve the method and obtain a better outcome.

## 2.5 Technological review

---

In current times, text summarization systems have numerous applications, particularly in research papers. Summaries enable users to easily grasp the context and key concepts in various contexts such as research papers and customer reviews. Text summarization tools are frequently utilized by researchers to create abstracts of their findings. Text summarization techniques can be either extractive or abstractive. While extractive text summarization involves selecting important segments of text from the original content, abstractive text summarization creates a new context that is more logical or human-like in language and can aid in problem-solving (Barna and Heickal, 2022). Text summarizers can be particularly useful in identifying the key elements of reviews by providing a summary of user reviews, which can be lengthy and descriptive at times.

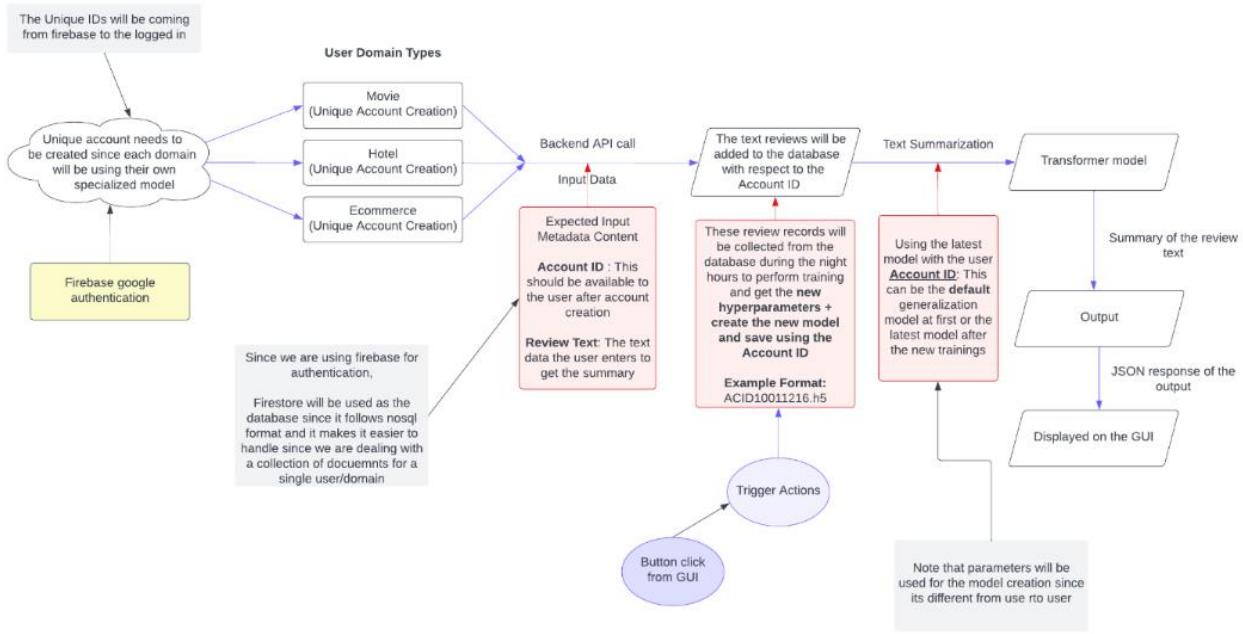
Although traditional machine learning and deep learning techniques have been widely used for text summarization in the movie review domain, advanced deep learning approaches such as Transformers have not been explored. While traditional approaches have performed well, there are limitations to pushing the boundaries with new approaches. Therefore, Transformer optimization is being considered through repeated hyperparameter tuning with exposure to new data to make the approach applicable to any domain.

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

The presented figure depicts a methodology utilized to create a universal text summarization system. The process entails enhancing the system's efficiency by retraining the model with pertinent domain-specific data and identifying a new set of hyperparameters (for model customization and model training) that are appropriate for the updated data. The model is

subsequently retrained utilizing the optimal transformer architecture chosen, making it more tailored to the domain and consequently improving the system's performance.

Figure 2: Proposed Generalized Abstractive Summarization System Process Flow (Self-Composed) – (*view high qual version*)



## 2.5.2 Machine Learning Text Summarization Techniques

According to (Boorugu, Ramesh and Madhavi, 2019), a previous study used a hybrid classifier approach with SVM and Naïve Bayes machine learning algorithms combined with fuzzy logic. They found that increasing the number of classifiers can improve accuracy. They also used supervised ML algorithms, such as KNN, for product feature identification by combining appropriate words.

On the other hand, (Khan et al., 2020) proposed a system for summarizing customer reviews in the movies domain. Their methodology included preprocessing, feature extraction, review classification, and review summarization. They employed the Naive Bayes classification method, which is known to achieve high accuracy, to categorize reviews as negative or positive using supervised ML classification. They used an extractive summarization approach, which involved creating a graph from classified reviews, ranking the graph nodes, and selecting the top-ranked sentences for summary generation.

Initially, machine learning approaches were widely used for text summarization, but new technologies and techniques like RNN, CNN, etc., have emerged over time to achieve better performance.

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

Etemad, Abidi, and Chhabra (2021) conducted numerous studies on deep learning approaches to abstractive text summarization, including the use of CNN, LSTM-CNN, Convolutional Seq2Seq, Sequence to Sequence RNN, Convolutional Sequence to Sequence, Transformers, T5, BART, BERT, and more. These models were trained on general datasets such as Gigaword, DUC 2002, DUC 2004, CNN Daily Mail, DUC, Xsum, Newsroom, and others to determine which approach performed the best. Ultimately, the T5 Transformer proved to outperform the other techniques for abstractive text summarization.

Shi and colleagues (2020) performed an extensive examination of recent advancements in seq2seq models for abstractive text summarization, reviewing multiple different models for abstractive summarization.

Among these models, transformers emerged as the most advanced deep learning approach for text summarization, as they are encoder-decoder models equipped with an attention layer that enables them to produce superior results compared to traditional simple RNN architectures (Mahajan et al., 2021).

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

The author will examine two datasets during the course of the project. One of these datasets is the Amazon movie reviews dataset from Stanford University Education, which includes over 8 million review data records spanning more than 10 years (McAuley and Leskovec, 2013).

This dataset will be utilized to test the abstractive text summarization solution in the domain of movies. After successfully creating the solution for this domain, the author plans to extend the solution's scope by employing the Gigaword dataset from TensorFlow datasets, which has previously been used to generate generalized content for text summarization (Kouris, Alexandridis, and Stafylopatis, 2019).

### 2.5.5 Preprocessing Techniques used in Text Summarization

Effective text preprocessing is crucial for handling text-based data. A range of text preprocessing techniques have been employed in prior studies to facilitate text summarization. One key step in natural language processing (NLP) applications, including information retrieval, machine translation, semantic role labeling, and summarization, is sentence segmentation. This process involves identifying boundaries within a document that demarcate its text into sentences, usually at punctuation marks like periods, exclamation points, and question marks. Additionally, tokenization and stop word removal are performed. Tokenization is accomplished via a tokenizer program that splits sentences into distinct words by dividing them at white spaces like blank spaces, tabs, and strong punctuation marks. Stop word removal is used to eliminate frequently occurring words in the document, such as "I," "an," and "a," which are of little significance and can be removed from the text (Khan et al., 2020).

Other scholars have integrated various techniques, including noise removal to eliminate extraneous text from input documents, such as headers and footers, and named entity recognition (NER) to identify words in the input text as names of people, places, and objects, among other things (Barna and Heickal, 2022).

Datasets may also contain unneeded, null, or duplicated records that are completely useless. These null-value records or rows are eliminated, and unnecessary HTML tags and URL links are removed from the text as part of the text preprocessing. Contraction mapping is critical and will handle short word forms such as "aren't" and expand them to longer versions such as "are not." Converting the entire text content to a single case, preferably lowercase, simplifies further character filtration (Mahajan et al., 2021).

## 2.6 Evaluation Techniques

---

### 2.6.1 Evaluation Approaches

The assessment of a machine learning model's performance, as well as its strengths and limitations, is achieved through model evaluation which involves various evaluation measures. It is essential to evaluate models in the initial stages of research to determine their effectiveness.

The following table presents the measures and metrics that can be used to evaluate the text summarization system quantitatively.

Table 3: Evaluation metrics for abstractive text summarization (Self-Composed)

Measure	Description	Objective Orientation
<b>Metric: 01</b>		
ROUGE	ROUGE, which stands for Recall-Oriented Understudy for Gisting Evaluation, is a measure used to compare an automatically generated summary or translation against a set of reference summaries, typically created by humans. This measure evaluates the recall of the automatically generated summary by comparing the frequency of terms from the human-created summaries with those from the computer-generated summary (Lin, 2004).	Positively oriented. Higher, the better
<b>Equations</b>		
<b>Rouge-1</b>		
Refers to the system summary and reference summary's overlap of <u>unigrams</u> (one-word sequence).		
$ROUGE - 1 = \frac{(\sum_{\text{Reference Summary}} \sum_{\text{unigram}} \text{Count}_{\text{match}}(\text{unigram}))}{(\sum_{\text{Reference Summary}} \sum_{\text{unigram}} \text{Count}(\text{unigram}))}$		
<b>Rouge-2</b>		
Refers to the <u>bigram</u> (two-word-sequence) overlap between the system and the reference summaries		
$ROUGE - 2 = \frac{\sum_{S \in \{\text{RefSummaries}\}} \sum_{\text{bigrams } i \in S} \min(\text{count}(i, X), \text{count}(i, S))}{\sum_{S \in \{\text{RefSummaries}\}} \sum_{\text{bigrams } i \in S} \text{count}(i, S)}$		
<b>Rouge-L</b>		
Measures the longest matching word sequence		
$ROUGE - L_{(\text{candidate}, \text{references})} = \max_k \{ ROUGE - L_{\text{single}}(\text{candidate}, \text{references}_k) \}$		

<b>Metric: 02</b>		
<b>Equations</b>		
BLEU	BLEU, or Bilingual Evaluation Understudy, is a metric used to evaluate the quality of machine-generated text by comparing it to a reference text that is expected to be generated. According to Steinberger and Jezek (2009), BLEU measures precision, which is the extent to which words in the generated summaries appear in the human-generated summaries.	Positively oriented. Higher, the better

ROUGE, which stands for Recall-Oriented Understudy for Gisting Evaluation, has several variations such as ROUGE-1, ROUGE-N, ROUGE-L, and ROUGE-S, each with its unique way of evaluating the performance of a machine-generated text against reference summaries created by humans.

For instance, ROUGE-L evaluates the longest common sequence while ROUGE-S and ROUGE-SU consider skip sequences. According to Etemad, Abidi, and Chhabra (2021), the most common ROUGE versions used for evaluation in text summarization are ROUGE-1, ROUGE-2, and ROUGE-L, with the scores ranging from 0 to 1, and higher scores indicating better performance. Comparing ROUGE and BLEU, Steinberger and Jezek (2009) suggest that ROUGE performs better for text summarization, while Lin (2004) introduced the ROUGE package for summarization evaluation, which demonstrated better performance through thorough evaluations of automated measures using three years' worth of DUC data.

## 2.6.2 Benchmarking

The following table displays the benchmarking outcomes of training transformers with generalized datasets for abstractive text summarization. The author plans to adopt these same measures for evaluating the prototype in order to make it comparable.

Table 4: Comparison table for abstractive text summarization using transformers (Etemad, Abidi and Chhabra, 2021)

<b>Researcher</b>	<b>Year</b>	<b>Type of model</b>	<b>Rouge 1</b>	<b>Rouge 2</b>	<b>Rouge L</b>	<b>Dataset</b>
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	17.87	36.17	CNN Daily Mail
			36.73	14.93	29.66	Xsum
			40.87	28.59	37.62	Newsroom
Kaiqiang Song et al.	2019	Transformer	40.89	19.11	37.60	Gigaword, Newsroom
			45.93	24.14	42.51	
Mike Lewis et al.	2019	BART	44.16	21.28	40.90	CNN Daily Mail
			45.14	22.27	37.25	Xsum
Itsumi Saito et al.	2020	RoBERTa Base	45.80	22.53	42.48	CNN Daily Mail
			45.42	22.13	36.92	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

At the start of this chapter, a concept map was created to breakdown the problem, technology domains, prior work, and assessment strategies. Each of these four areas was further subdivided into subtopics and explored in the context of previous research and ideas presented in the literature. A comprehensive analysis of the literature was conducted, comparing and contrasting the strengths and weaknesses of previous research, potential future work identified in the literature, and innovative approaches that the author of this study suggests as unexplored possibilities.

## CHAPTER 03. METHODOLOGY

---

### 3.1 Chapter overview

This chapter outlines the selected methodologies, reasons for their selection, project prerequisites, hazard minimization strategies, project timeline, task distribution plan, and expected outcomes to ensure efficient research procedures.

### 3.2 Research methodology

When evaluating project quality, key factors like cost, time, and scope must be managed effectively using appropriate methodologies. The Saunders Research Onion model was chosen and implemented for this project, along with other methodologies that were considered and presented in a table., as described by Saunders, Lewis, and Thornhill (2007).

Table 5: Research Methodology (Self-Composed)

Research Philosophy	To determine the most effective technique for achieving the research aim, the author will experiment with various methods using a combined approach. As a result, the author has opted for the <b>pragmatic approach</b> over positivism, realism, and interpretivism approaches.
Research Approach	The researcher will use various methods to determine the most effective approach to achieve the research objective. They chose the <b>deductive approach</b> since the goal is to apply existing model architectures for optimal results. The researcher also opted for <b>qualitative data analysis</b> methods.
Research Strategy	To answer the research questions, multiple data collection methods were chosen. Survey, Archival Research, and Ethnography were selected to complement each other and provide adequate data for the research. Although <b>Survey</b> is the main method, <b>Archival Research and Ethnography</b> were included to address the qualitative aspect of the research, which may impact the quantitative results.
Research Choice	The methodology chosen for a study determines whether it will focus on qualitative or quantitative aspects. Although this study prioritizes quantitative findings, a <b>multi-method</b> approach was considered because it's important to

	determine the quality of data used in the development process, which can affect the quantitative results.
Time Horizons	A <b>cross-sectional approach</b> will be used to collect data for the requirement engineering and evaluation phases. This approach involves collecting data only once and not repeatedly over time.
Techniques and procedures	This section discusses the data collection and analysis methods to be used. The sources of data collection will include internet news, discussions, reports, surveys, publications, and organizational records.

### 3.3 Development methodology

---

#### 3.3.1 Life cycle model

The project will use the **Agile** Software Development Life Cycle as its research development methodology because the project relies on an iterative development method.

#### 3.3.2 Requirement elicitation methodology

Approaches such as surveys, reviewing previous research, experimenting with transformer architectures, and brainstorming will be used to gather insights for the project.

#### 3.3.3 Design methodology

The system's expandability and future growth were considered, leading the author to choose **SSADM as the Design Methodology** for the project, as it supports an incremental methodology.

#### 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 (initial and final phase) and it can be found at **APPENDIX.H.3**

### 3.4.1.2 Deliverables

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

Table 6: Deliverables & Dates (Self-Composed)

<b>Deliverable</b>	<b>Date</b>
<b>Literature Review</b> A comprehensive evaluation of existing research and proposed solutions.	27 <sup>th</sup> October 2022
<b>Project Proposal Document + Ethics Forms</b> Initial proposal of the project.	9 <sup>th</sup> November 2022
<b>Software Requirement Specification</b> Documentation outlining requirements, including data collection methods, designed as the ultimate prototype.	24 <sup>th</sup> November 2022
<b>Proof of Concept with Implementation Presentation</b> Presenting implementation and proof of concept.	23 <sup>rd</sup> December 2022
<b>Project Specifications Design &amp; Prototype</b> Functional prototype with documented design approach.	16 <sup>th</sup> February 2023
<b>Test &amp; Evaluation Report</b> Documented Evaluation Report conducted on the Prototype.	23 <sup>rd</sup> March 2023
<b>Draft Project Reports</b> A draft thesis submission, in order to get supervisors feedback	10 <sup>th</sup> April 2023
<b>Final Thesis</b> Final report detailing the research and project decisions	27 <sup>th</sup> April 2023
<b>Review Research Paper</b> Review paper on existing systems for abstractive text summarization.	8 <sup>th</sup> April 2023
<b>Final Research Paper</b> Research paper on transformer optimization experimentation.	25 <sup>th</sup> April 2023
<b>Public project repository</b> A publicly accessible project repository to setup and test the development	27 <sup>th</sup> April 2023

## 3.5 Resources

---

### 3.5.1 Software requirements

- **Operating System** – Najmuddin et al. (1970) prefer Microsoft Windows OS due to its greater flexibility for the project implementation.
- **Python** – McKinney (2017) suggests using Python for machine learning models and APIs, as it's popular in data science and supported by frameworks like Flask and Django.
- **Flask** – Chauhan et al. (1970) recommend a flexible and lightweight backend web framework for easy data transfer in the prototype's API development.
- **Torch** – Paszke et al. (1970) stated that using lightweight libraries instead of TensorFlow during data science model development will make building models quicker and easier.
- **Jupyter Notebook & Google Colab** – Canesche et al. (1970) used Google Colab as IDE for on-the-go machine learning and deep learning model training.
- **TypeScript (React)** – JavaScript framework for frontend development of user interface allowing input and data viewing.
- **Zotero** – Reference software for managing research paper references and articles.
- **MS Office/ Google Docs/ Figma** – Software and tools for figures, reports, and documentation.
- **Google Drive/ GitHub** – Backup platform and code management system.

### 3.5.2 Hardware requirements

- **Core i5x Processor (8<sup>th</sup> 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.

### 3.5.4 Data requirements

- Gigaword, Xsum & CNN daily news dataset – from TensorFlow datasets which will be used for generalization model (for training).
- Amazon movie review data – from Stanford university education (for testing).
- Hotel review dataset – from Kaggle (for testing).

## 3.6 Risks & mitigation

The possible risks along with their mitigation steps were identified prior to starting the project.

Table 7: Risk Management Plan (Self-Composed)

Risk	Severity	Frequency	Mitigation Plan
Losing the development project codebase/repository	5	4	Storing project code using GitHub and external backup.
Computer breakdown during project timeline	5	4	Upload backup to GitHub & Drive, use UniLab service till personal machine recovery.
Project deliverables not completed on time.	4	4	Prioritize and create a timeline to complete the deliverables.
Project documentation corruption	5	3	Use a dedicated GitHub folder for documentation and update regularly. Consider cloud-based documentation.
Insufficient knowledge on the project domain	5	3	Intensively researching the problem and research domain.
Any unavoidable personal health risk – Sickness	3	1	Create weekly goals to complete and keep them updated.

## 3.7 Chapter summary

This chapter explains the project's methodology, including research and development approaches, project management strategies, and reasons behind those decisions. It also covers project requirements, work division plan, expected outcomes, potential risks, and their mitigation strategies.

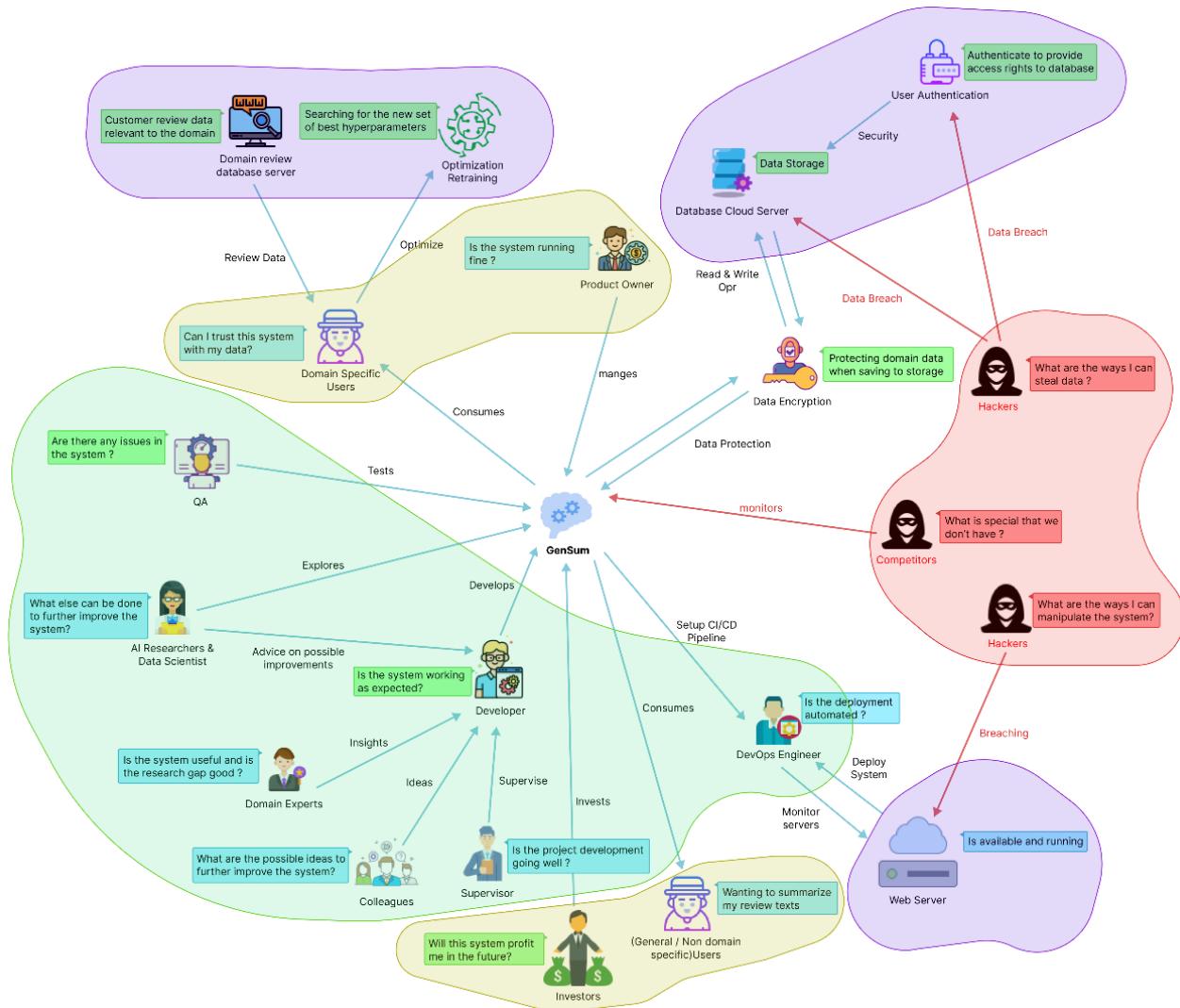
# CHAPTER 04. SOFTWARE REQUIREMENTS SPECIFICATION

## 4.1 Chapter overview

The chapter covers how to gather crucial needs from stakeholders using a rich picture diagram and a stakeholder onion model. It also discusses various techniques for requirement gathering, including creating functional and non-functional requirements, use case diagrams, and prototypes.

## 4.2 Rich picture

Figure 3: Rich Picture Diagram (Self-Composed) – (view high qual version)



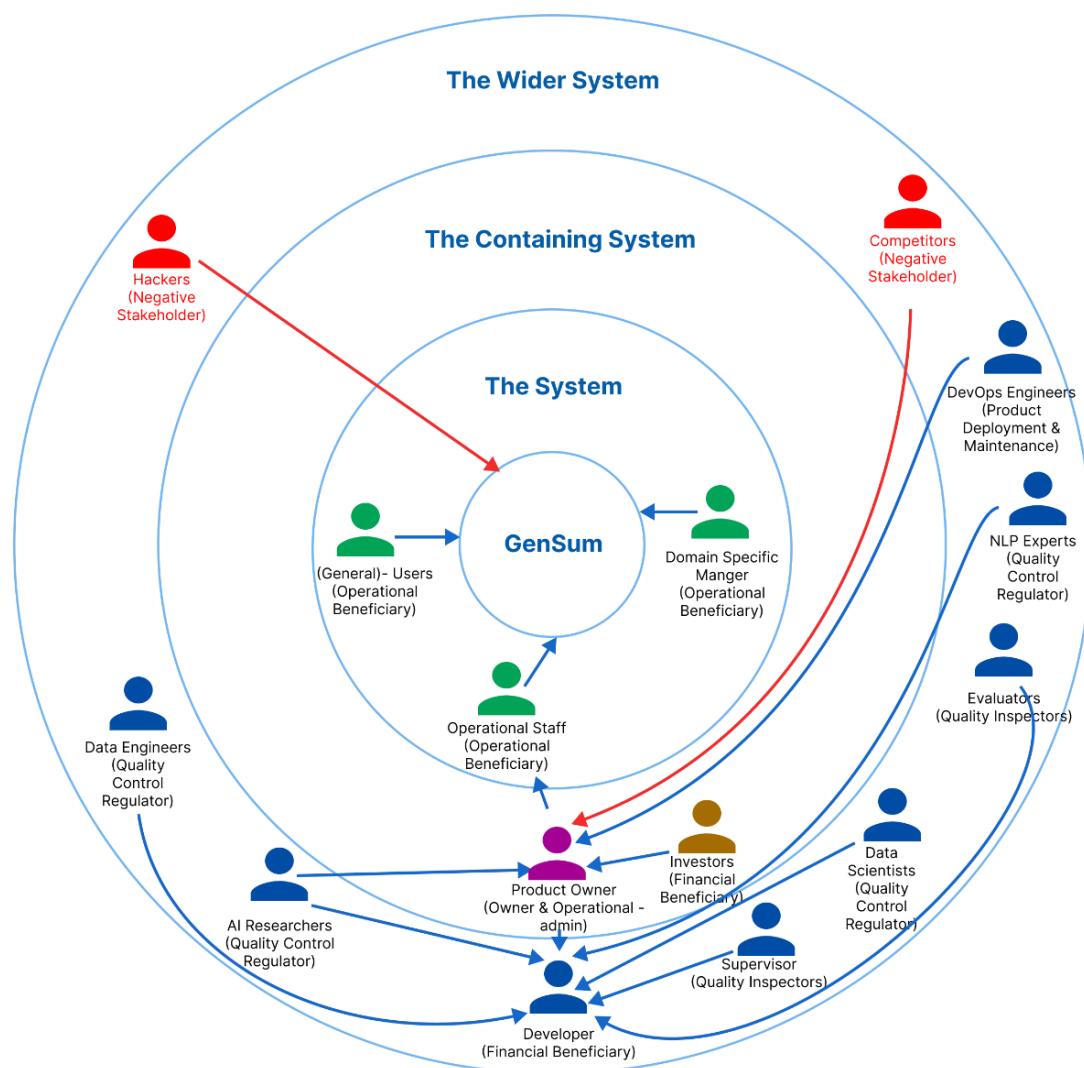
The diagram shows a bird's-eye view of the surrounding region and stakeholder interactions with the system. It also identifies potential negative impacts and prospective critical analyses in addition to the researcher's gained knowledge to improve the system.

### 4.3 Stakeholder analysis

The section identifies important stakeholders in the system, their relationships, and roles using the stakeholder onion model. It further elaborates on their perspectives.

#### 4.3.1 Stakeholder onion model

Figure 4: Stakeholder Onion Model (Self-Composed) – (view high qual version)



### 4.3.2 Stakeholder viewpoints

Table 8: Stakeholder Viewpoints &amp; Requirements (Self-Composed)

Stakeholder	Role	Benefits/Description
Developer	Functional	Works on developing the system
Investors	beneficiary	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, 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 9: 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** contains a complete breakdown of these stakeholders.

### 4.5.1 Literature review

Table 10: Literature Review Findings (Self-Composed)

Discussion of Findings	Citation
Upon concluding the review of existing literature, it was determined that abstractive text summarization systems applied to customer reviews can aid users in making more informed and expeditious decisions, whether those decisions pertain to purchasing products or selecting a movie. The summarization of user reviews has been shown to be a time-saving approach for customers.	(Boorugu, Ramesh and Madhavi, 2019)

During the investigation of technologies suitable for achieving the desired outcome, it became evident that conventional machine learning and deep learning methods were solely utilized for abstractive text summarization in the movie review domain, while more sophisticated deep learning techniques such as Transformers remained unexplored for this specific domain.	(Khan et al., 2020)
It was observed that the optimization of transformers has not been examined in the context of abstractive text summarization in general, rather than solely in the movie domain.	(Gupta et al., 2021)
Previously, datasets that are relevant to achieving model generalization have been utilized and are suggested for use by researchers who aim to explore the concept of generalization in the abstractive text summarization domain.	(Kouris, Alexandridis and Stafylopatis, 2019)

#### 4.5.2 Brainstorming

The author brainstormed with colleagues, supervisors, and through self-analysis across various project phases.

Table 11: Observations Findings (Self-Composed)

Criteria	Discussion of Findings
The author has identified several gaps or limitations in the research related to the current project domain, which could be addressed to broaden the scope of research.	After a brainstorming session, multiple ideas were generated. The author's supervisor suggested creating a performance adaptive generalization model and utilizing newly acquired data from domain users for retraining purposes. They also proposed amalgamating all data related to the common domain for retraining, which would increase data count to enhance the system's performance.

#### 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

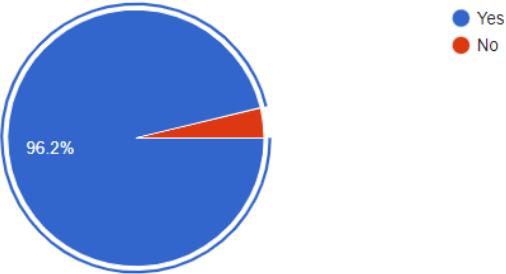
and 1 Lecturer with MSc completion were interviewed. The interview outcomes were processed to a thematic analysis based on the following themes. Interview participant details along with the related evidence & the theme-codes can be found at over **APPENDIX C.2**

<b>Theme</b>	<b>Conclusion</b>
Data handling	Data accessibility and preparation are crucial factors in a data science project. PhD candidates recommended using well-researched and validated datasets for the generalization field to ensure data quality. However, NLP experts raised concerns about including text data with characters from other languages, unless the project is restricted solely to English language support.
Transformer architectures	Interviewees suggested that transformer architectures like BERT, GPT-2, Roberta, and T5 can effectively address NLP tasks such as text summarization and sentiment analysis. They recommended using the latest versions of these models, as they are regularly updated and improved. To stay informed about these updates, they suggested monitoring daily analytics from websites like Hugging Face, including download and like counts.
Generalization	Software engineers and architects recommended utilizing NoSQL databases such as MongoDB or Firebase for storing data pertaining to domain-specific managers, citing their scalability and performance benefits.
Research scope	The experts concur that utilizing optimized transformers to address the issue is a commendable idea, but due to the time constraints of the project, they suggest giving priority to the movie domain and delaying the development of a more comprehensive adaptive solution.
Custom transformers	PhD applicants also mentioned that customizing the transformer architecture will also help contribute to the performance of the system.
Hyperparameter tuning	NLP researchers and lecturers proposed several methods of employing tools and libraries to facilitate hyperparameter tuning, as doing so manually can be excessively time-consuming and unnecessary.
Hybrid transformers	PhD applicants expressed a positive response towards the hybrid transformer combination with ensemble techniques. However, many believed that the

	project's scope was expanding beyond the available time and increasing its riskiness.
Prototype	The interviewees showed an interest in the novel domain-specific retraining system and suggested incorporating a pretrained model for sentiment analysis if time permits.
Business benefits	According to most interviewees, the movie domain, tourism, e-commerce, book industry, and researchers would find the use of text summarization systems for customer reviews beneficial in their businesses.
Evaluations	The PhD candidates and NLP experts stressed the importance of evaluations in the adaptive generalization model and proposed limiting the project to a maximum of three domains to facilitate comparison and demonstrate the model's effectiveness more clearly.

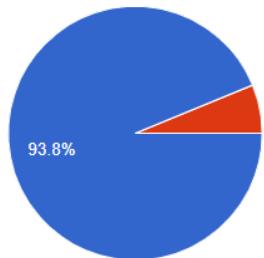
#### 4.5.4 Survey

Table 12: Survey Analysis (Self-Composed)

<b>Question</b>	Have you ever realized that reading lengthy reviews takes a significant amount of time?
<b>Aim of question</b>	To determine whether the audience as a whole considers reading lengthy reviews to be a time-consuming activity.
<b>Findings &amp; Conclusion</b>	 <p>The findings suggest that over 90% of the audience considers reading lengthy reviews to be a time-consuming and burdensome task, and they would appreciate a faster solution to this issue, such as summarization. These results confirm the author's initial hypothesis regarding a positive correlation..</p>
<b>Question</b>	Do you believe that developing a generic system for all domains would be a wise course of action?
<b>Aim of question</b>	Ensuring that developing a generic system would be beneficial in all domains

### Findings & Conclusion

The results suggest that over 90% of the participants are in favor of developing a generalized system that can adapt to the specific domain as it is being used. They believe that this effort is

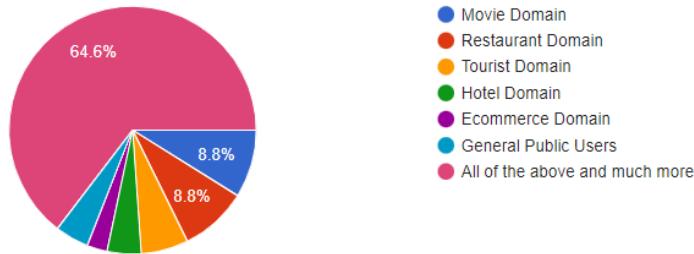


● Yes  
● No

valuable and worth pursuing in the context of the project. This finding is in line with the author's expectations and indicates a positive correlation between the intended goal of the project and the participants' views.

<b>Question</b>	Who do you think will most benefit from this system?
<b>Aim of question</b>	Getting to know about the thoughts of the participants about to whom the system would mostly benefit from?

### Findings & Conclusion



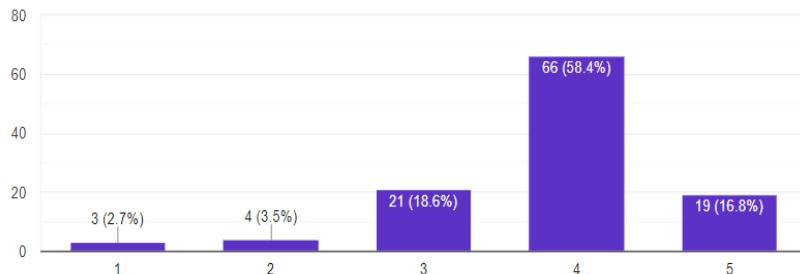
The conclusion is that more than 60% of the participants believe the proposed system would be beneficial for movie, restaurant, tourism, hotel, and e-commerce domains as they interact with users

on a daily basis and use customer reviews as part of their business. The system could also benefit general users.

<b>Question</b>	How much do you think that this system would benefit you?
<b>Aim of question</b>	Getting to know how much the system would benefit the general participants which are NOT domain specific

### Findings & Conclusion

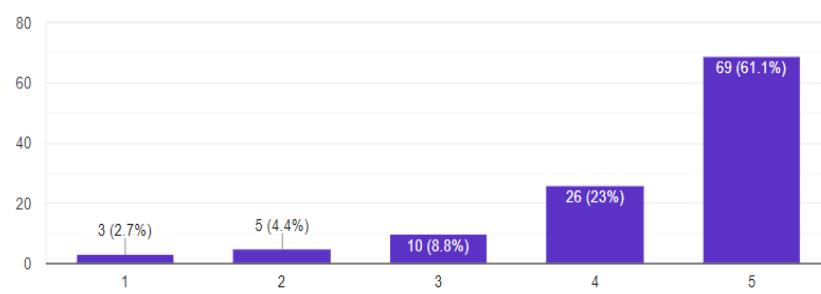
Based on the statistics graph, it can be inferred that approximately 75% of the respondents believe that the system would be helpful for their general work or needs, provided that it is not limited to a specific domain.



This result indicates a positive correlation with the collected statistics.

<b>Question</b>	How much do you think that this system would benefit businesses?
<b>Aim of question</b>	Getting to know from the participants as to how much the system would benefit businesses/domains in solving this problem.

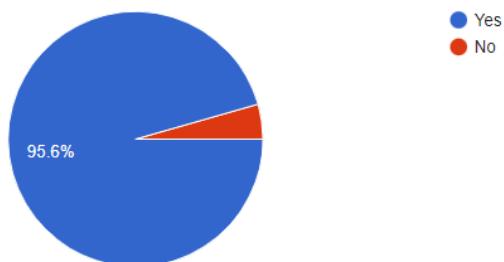
#### Findings & Conclusion



Based on the statistics graph, it can be inferred that around 84% of the participants believe that the system would be beneficial for businesses. This aligns with the author's initial expectation and is a positive correlation from the obtained statistics.

<b>Question</b>	Before making a reservation or booking a movie or a hotel, do you read the customer reviews?
<b>Aim of question</b>	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



Based on the results, it can be inferred that over 95% of the audience considers customer reviews a crucial element in their decision-making process when it comes to purchasing a product or service. This highlights the

significance of customer reviews in fostering business growth.													
<b>Question</b>	How much you think customer reviews are important with respect to any domain?												
<b>Aim of question</b>	Getting an idea from the audience to see how much they value customer reviews.												
<b>Findings &amp; Conclusion</b>													
<p>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.</p> <table border="1"> <thead> <tr> <th>Rating</th> <th>Percentage</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>1.8%</td> </tr> <tr> <td>2</td> <td>1.8%</td> </tr> <tr> <td>3</td> <td>4.4%</td> </tr> <tr> <td>4</td> <td>28.3%</td> </tr> <tr> <td>5</td> <td>63.7%</td> </tr> </tbody> </table>		Rating	Percentage	1	1.8%	2	1.8%	3	4.4%	4	28.3%	5	63.7%
Rating	Percentage												
1	1.8%												
2	1.8%												
3	4.4%												
4	28.3%												
5	63.7%												
<b>Question</b>	Which additional features would you want to see in this system.												
<b>Aim of question</b>	To identify the systems non-functional requirements which could potentially improve the system.												
<b>Findings &amp; Conclusion</b>													
<p>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.</p>													

Table 13: Survey Thematic Analysis (Self-Composed)

Code	Theme
Convenience	User-friendly
Adjustability	Flexibility
Insights	Reliability
Advancement	Future Enhancement

Theme	Conclusion
User-friendly	A group of participants required to upload more than one review and a time/bulk at once.
Flexibility	A majority of the participants requested for sentiment of the summary and the sentiment score to be also included with the output.
Reliability	A majority of the participants found the system being useful in the industry with time.
Future Enhancement	Few participants requested integrate a recommendation system for a specific domain.

#### 4.5.5 Self Evaluation

Author does competitor analysis, self-evaluates data protection from hackers and lists abstractive text summarization tools.

Table 14 Competitor Analysis (Self-Composed)

Competitor Analysis Table					
Tools Feature	Summarize Bot	Resoomer	Smmry	Text Compactor	GenSum
Summarizing Text	✓	✓	✓	✓	✓
Domain Specific Generalization	✗	✗	✗	✗	✓
Ease of Use via GUI	✗	✓	✓	✓	✓
Summary sentiment and score	✗	✗	✗	✗	✓

GenSum is **unique** due to its Domain Specific Generalization feature, not found in other tools. Competitor tools lack sentiment information, and encryption can protect text data in the database from theft during model retraining.

#### 4.5.6 Prototyping

Table 15: Prototyping Findings (Self-Composed)

Criteria	Discussion of findings
<p>To investigate the viability of proceeding with the project research, a prototype was scheduled for development.</p>	<p>Throughout the iterative prototype process, the author faced various requirements and difficulties. One such challenge was the search for an appropriate dataset with the desired metadata within the movie domain. Following a thorough assessment, the author identified a substantial dataset comprising 8 million records. However, its size and the presence of noisy text made it challenging to preprocess, although it was eventually accomplished. As the project's focus shifted towards generalization, the author explored different datasets to ensure it.</p> <p>The author conducted experiments using a framework known as "Optuna" to automate the hyperparameter search for the model training and model architecture customization. Additionally, the system will undergo retraining using fresh data from the domain user, and the author intends to investigate at least three high-quality transformer designs to determine the optimal one.</p> <p>In preparation for the testing phase and to assess the model's generalization ability, the author gathered datasets from two domains, specifically movies and hotels. Additionally, the author explored data encryption techniques to ensure the security of the data.</p>

#### 4.5.6 Summary of findings

Table 16: Summary of Findings (Self-Composed)

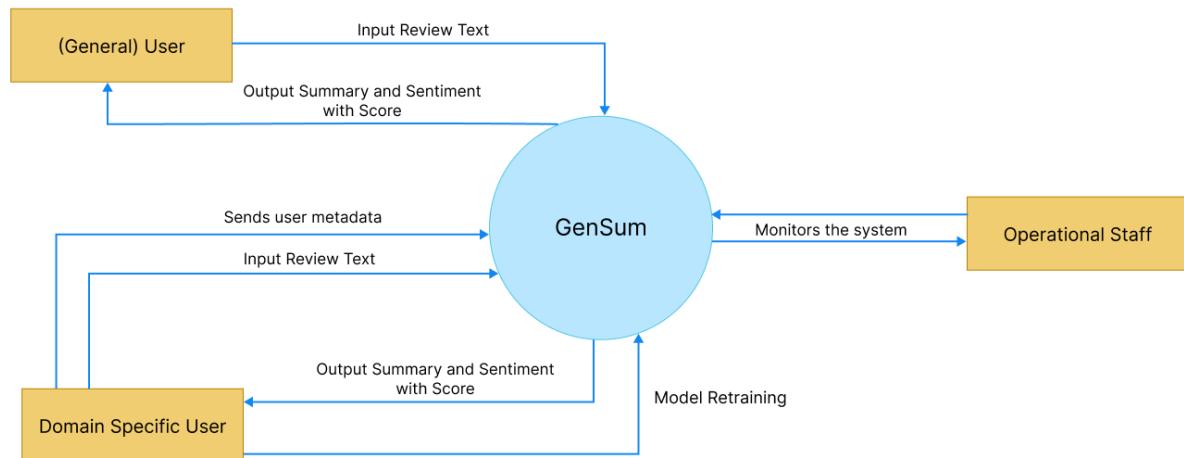
Id	Finding	LR	Survey	Self-Evaluation	Interview	Brainstorming	Prototyping
1	The proposed system benefits domain-specific and general users.	✓				✓	
2	Optimized transformers can enhance abstractive text summarization performance in the movie domain, which is currently limited.	✓			✓	✓	
3	Customer reviews are highly valued by the majority of users before using a product/service, across all domains.	✓	✓		✓	✓	
4	Users prefer shorter reviews to save time and make quicker decisions.	✓	✓			✓	
5	Hyperparameter tuning improves system performance and can be done manually or automatically.	✓			✓		✓
6	Preprocessing movie data for generalization requires considerable effort due to raw datasets that are difficult to obtain, especially with expected metadata.	✓					✓
7	Users mostly expect sentimental analysis and score for the review summary.		✓				
8	Suggest creating a hybrid transformer model to improve performance.				✓	✓	
9	Suitable evaluation metrics for abstractive text summarization are clear.	✓			✓		

10	It's clear on what the top tier transformer architecture that could be explored.	✓			✓		
11	Retrain using larger new data from businesses within the same domain (e.g., combine data from 50 restaurants for retraining if the domain is "Restaurants").				✓	✓	
12	Using data encryption to protect against data theft by hackers.			✓		✓	
13	Model architecture customization contributes to the increase in performance	✓			✓		✓

## 4.6 Context Diagram

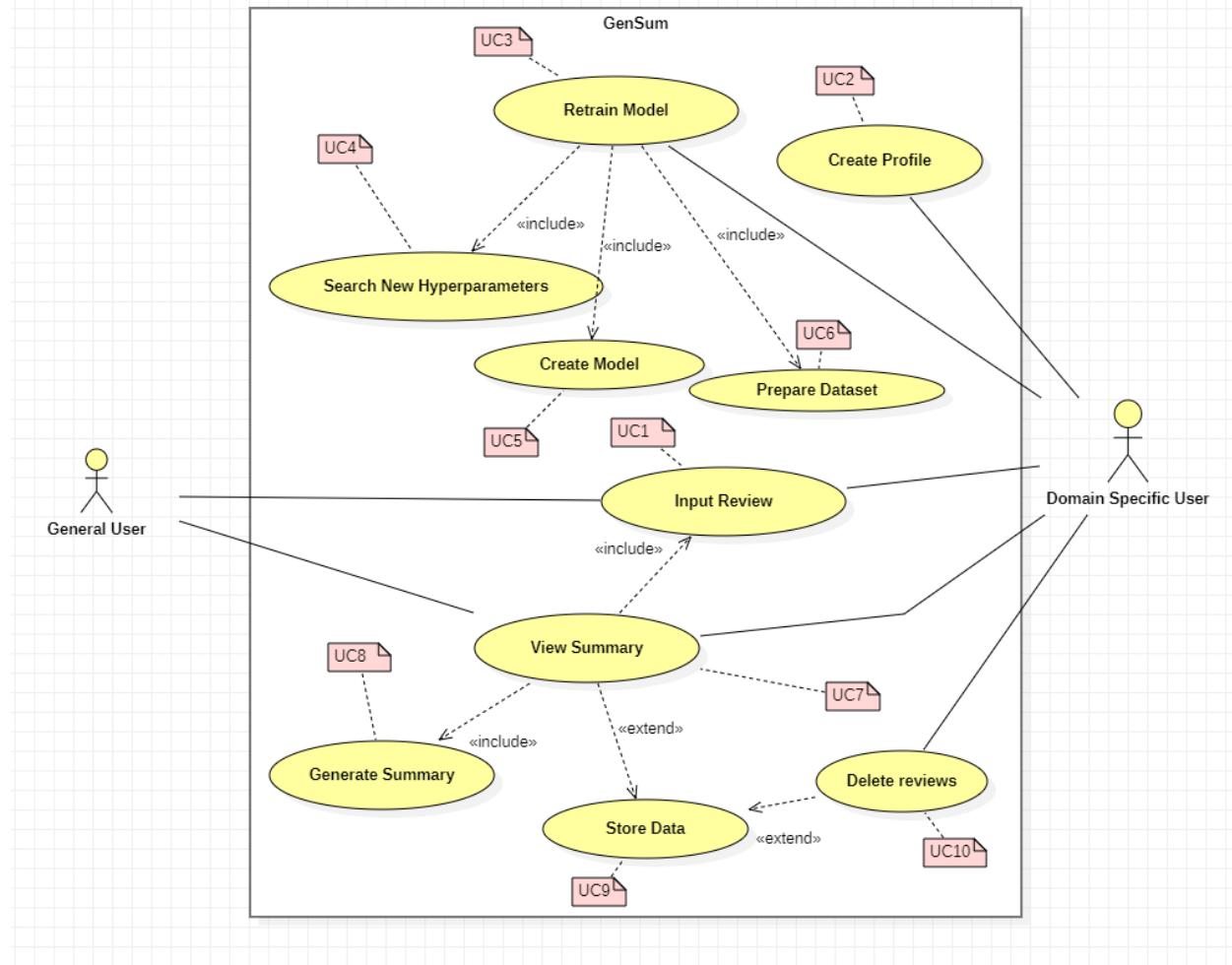
The boundaries and interactions of the system should be established before development. The graphic below shows how the system is situated.

Figure 5: Context Diagram (Self-Composed)



## 4.7 Use case Diagram

Figure 6: 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.3.**

Table 17: 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	UC:09	
Included use cases	UC:01, UC:08	
Trigger	A user selects to summarize a given customer/user review text.	
Main flow	Actor	System
	<ol style="list-style-type: none"> <li>1. The user enters the review text on the text field from the GUI.</li> <li>2. Clicks on “Generate Summary” from the GUI</li> </ol>	<ol style="list-style-type: none"> <li>3. The system does the data preprocessing for the input review text.</li> <li>4. Loads the generalized transformer model.</li> <li>5. Generates the summary using the model.</li> <li>6. (<b>If Domain Specific User</b>) stores the input review and summary into the database.</li> <li>7. Returns the summary response back to the GUI</li> </ol>
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 18: 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 for model customization and model training.
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	UC04, UC05, UC06

Trigger	The Domain Specific User clicks on the “Perform model retraining” button	
Main flow	<b>Actor</b>	<b>System</b>
	<ol style="list-style-type: none"> <li>1. Domain Specific logs into their account</li> <li>2. Clicks on “Perform model retraining” from the GUI</li> </ol>	<ol style="list-style-type: none"> <li>3. The system pulls all the data with respect to the user id from the database.</li> <li>4. Combines data of the common domains (only if user consent is given to use their data)</li> <li>5. Finds new set of hyperparameters for the model with respect to new data.</li> <li>6. Trains the model using the new hyperparameters.</li> <li>7. Saves the model with the user Id</li> <li>8. Updates the status in the database if succeed/fails</li> </ol>
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 C.4**. The Usecase description along its mapping id is also listed at **APPENDIX C.3**.

Table 19: Functional Requirements (Self-Composed)

FR ID	Requirement	Priority Level	Use Case
FR1	Both general and domain-specific users can input review text from the GUI for summary generation.	M	UC:01

FR2	Retrain system with new data for specific domains after obtaining user consent via GUI.	M	UC:03
FR3	The system must be able to find the new set of best hyperparameters with the usage of the new data.	M	UC:04
FR4	The system must be able to retrain the model with the new best hyperparameters and create the model	M	UC:05
FR5	The system must be able to pull the new data from the database to recreate the new dataset for retraining.	M	UC:06
FR6	The system must be able to process the review text and display the summary output on the GUI	M	UC:07
FR7	The system must be able to use the latest trained model to generate the summary for the review text	M	UC:08
FR8	User review and generated summary should be stored in the database for retraining.	M	UC:09
FR9	The system should encrypt the data when saving into the database (both the review and summary)	S	UC:09
FR10	Only domain-specific users can create an account after providing required details.	S	UC:02
FR11	The system could allow the ability to update the account details of the domain user after creating the account	C	UC:02
FR12	The system could be able to perform model retraining automatically during off peak hours every day.	C	UC:03
FR13	The system could also find the sentiment of the generated summary if its positive or negative and return the result.	C	UC:08
FR14	The system could make use of a hybrid model for the text summarization.	C	UC:08
FR15	Combine data from common domain groups for dataset creation only with approved consent.	C	UC:06
FR16	The system could allow the domain users to delete the reviews from the database.	C	UC:10

FR17	The system could automate and update the model architecture to increase performance.	C	UC:04
------	--	---	-------

#### 4.9.2 Non-Functional Requirements

The non-functional requirements are also prioritized based on the ‘MoSCoW’ approach.

Table 20: 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	M
NFR2	Summary generation should be done within 3000ms	Performance	M
NFR3	Following coding standards and best practices	Maintainability	M
NFR4	Application can be used by any domain users and model adapts to the domain.	Generalization	M
NFR5	Meaningful error messages should be displayed if anything goes wrong	Usability	C
NFR6	The system should protect against data corruption by attackers, and testing can ensure this.	Security	C
NFR7	The prototype must support many concurrent user-requests from multiple businesses under a single domain.	Scalability	C

#### 4.10 Chapter summary

A Rich Picture Diagram was used to show the system's interaction with society and stakeholders, with the Saundar's Onion model used to depict stakeholders. Requirement gathering techniques were used to collect data and opinions, and use cases, functional and non-functional requirements were specified accordingly.

# 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 21: SLEP Issues & Mitigation

Social	Legal
The thesis included a summary of questionnaire responses without exposing personal opinions. Interviewees were informed that their responses would be used and their consent was obtained for including their name or job title.	Study datasets are free to the public, and the research tools and technologies are open source or free to students, making them widely available. The codebase used in the research is also open source and accessible on GitHub under the MIT license.
Ethical	Professional
The participants were given clear information about the use of their collected data. The report contains original work and all external sources have been appropriately cited and credited.	The author maintained professional standards for the project and did not alter any collected responses or evaluations. Any limitations encountered during the study were 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

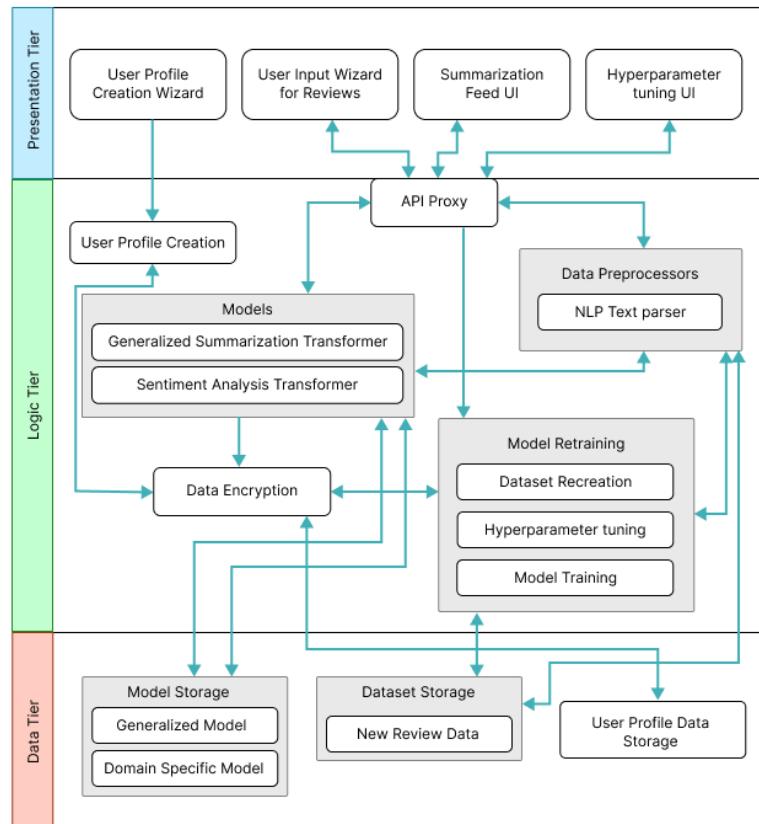
The design goals that should be achieved by the system are specified at [APPENDIX D.1](#).

## 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.

Figure 7: 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.
  - a) **Generalized Model** – The model which will be used by general users to generate review summarized, this model will be customized for generalized purpose.
  - b) **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.
2. **Dataset Storage** – The data which is required for model training will be available.
  - a) **New Review Data** – The data stored here are from the domain users when they use the application, the data will be stored and used for retraining.
3. **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.
  - a) **NLP Text parser** – Responsible for cleaning the input text review when received from the API endpoint.
4. **Models** – The model which will be responsible in generating the summary from the input review and find the sentiment of the summary generated.
  - a) **Generalized Summarization Transformer** – This is the summarization model which will be used, an adaptive custom model depending on the domain and type of user interacting with the model with optimized hyperparameters.
  - b) **Sentiment Analysis Transformer** – This model will be used to classify the generated summary into positive or negative sentiment.

5. **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.
6. **Model Retraining** – Responsible for retraining the model with new data and finding new set of hyperparameters.
  - a) **Dataset Recreation** – Responsible for recreating the dataset with new data which has been given as input from the domain users
  - b) **Hyperparameter tuning** – Responsible for finding the new best set of hyperparameters using the new data.
  - c) **Model Training** – Responsible for training the new model with the new set of hyperparameters found.

### Presentation Tier

1. **User Profile Creation Wizard** – UI offers users two options: create a new profile for domain business use, or skip sign up for a general summary.
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 author chose SSADM over OOAD for building a prototype due to its ability to easily extend system features in future development. Other reasons for choosing SSADM include its compatibility with existing company systems, availability of skilled personnel, and ability to produce comprehensive documentation. SSADM's phased approach and focus on requirements analysis were also seen as beneficial:

- Due to the main focus of the project being on Data Science, Object Oriented approaches are not expected to provide significant advantages.
- 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.

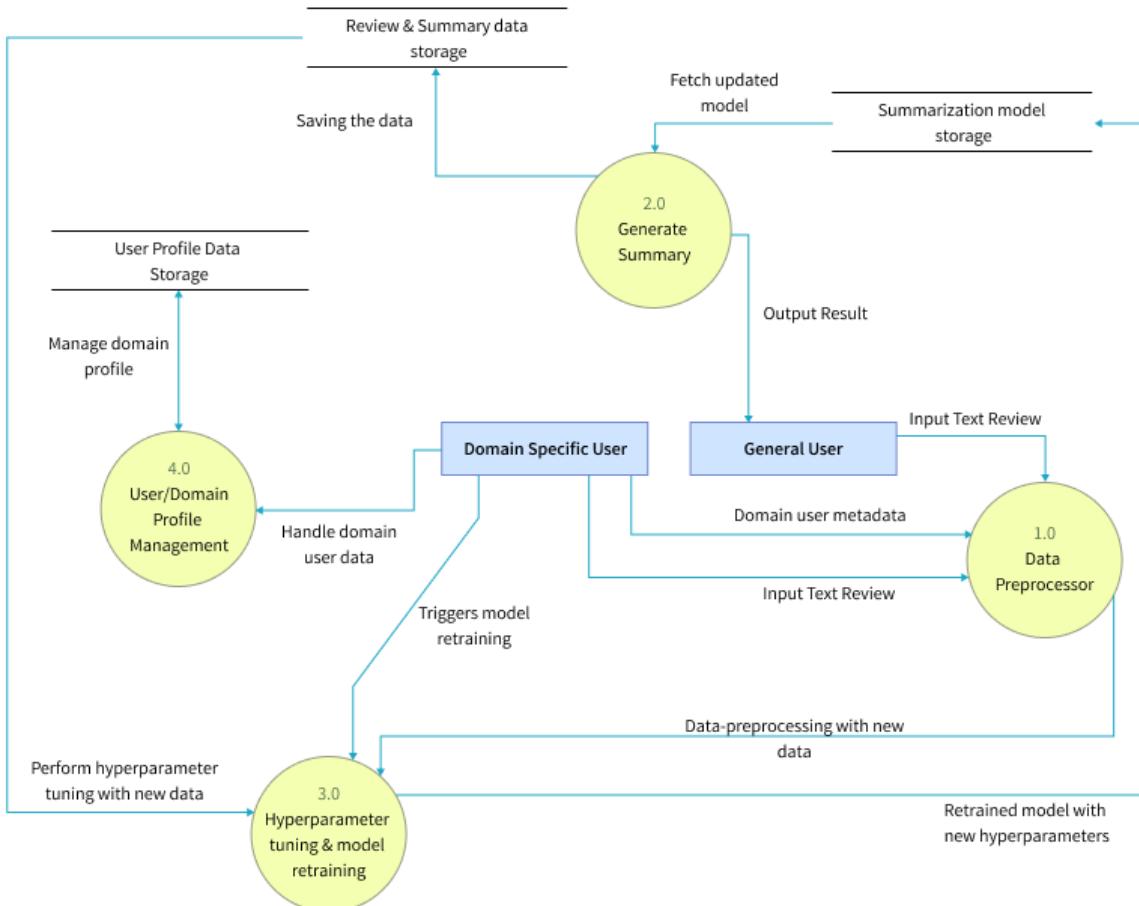
## 6.5 Design diagrams

### 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

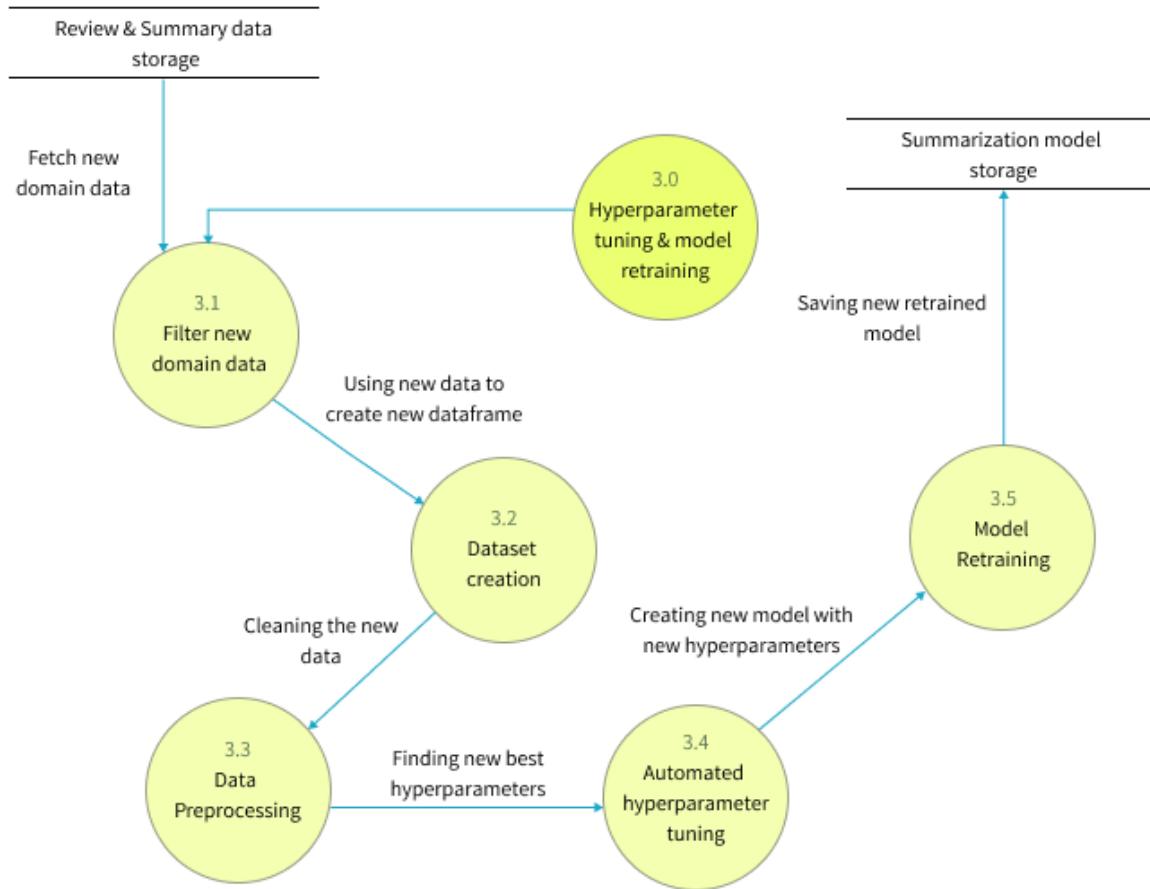
Figure 8: Data Flow Diagram - Level 01 (Self-Composed)



### 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 9: Data Flow Diagram - Level 02 (Self-Composed)



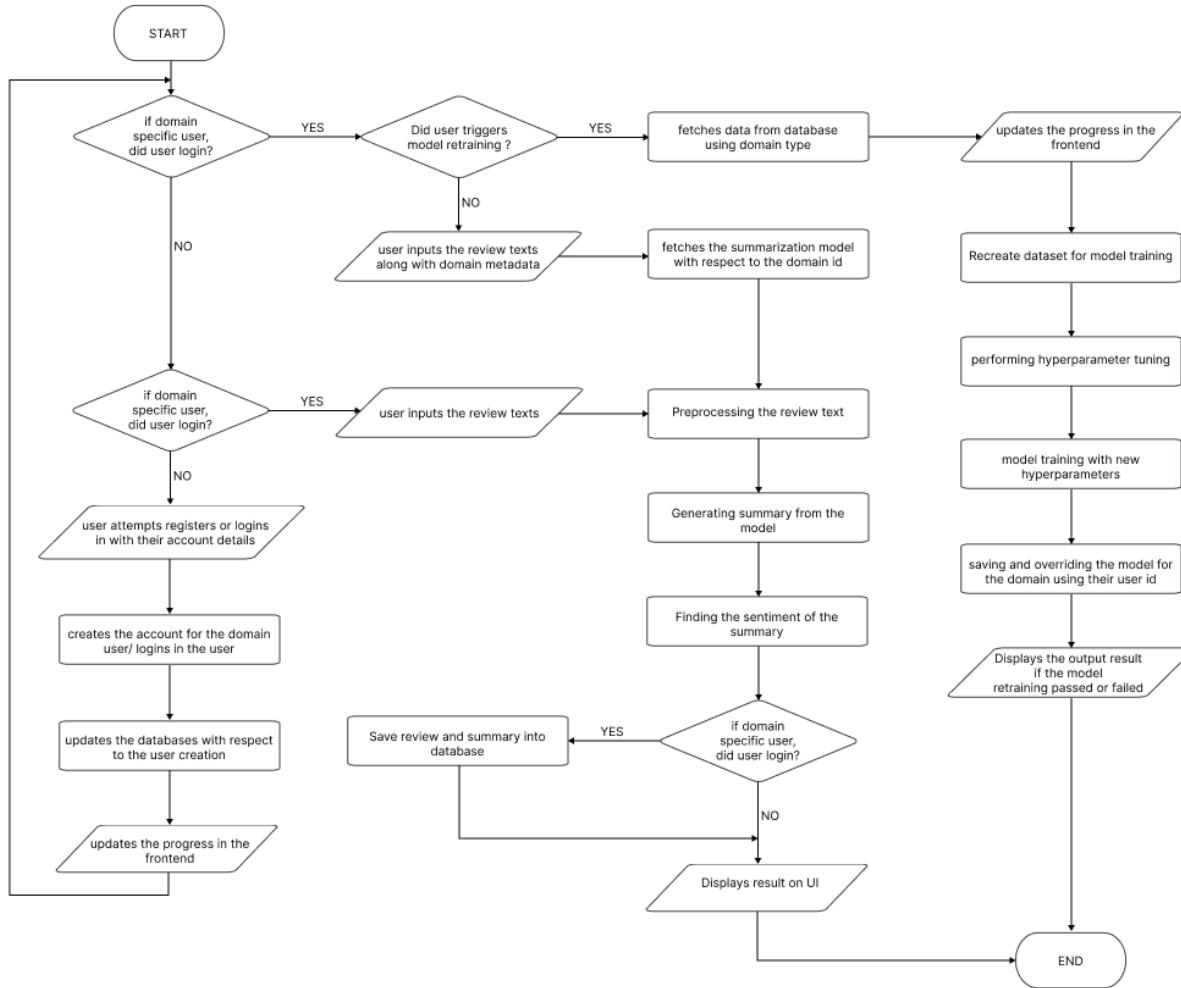
### 6.5.4 UI design

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 D.2**.

### 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 10: System Process Flow Chart (Self-Composed) – (*view high qual version*)



### 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.

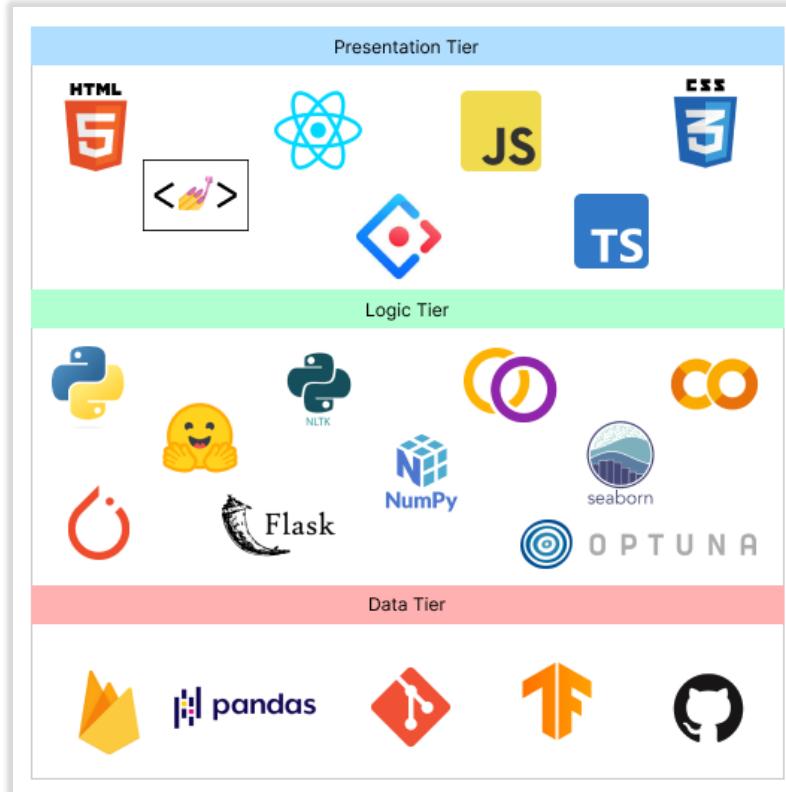
## 7.2 Technology selection

---

### 7.2.1 Technology stack

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

Figure 11: 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

To ensure the accuracy of the generalized text summarization model, it is essential to use trustworthy data sources for training. The project relies heavily on data science for this purpose. To develop an adaptive generalized text summarization model, a generalized dataset was necessary for establishing the base model. TensorFlow datasets provided several options for this dataset, being a reputable source of data.

Table 22: Dataset Sources (Self-Composed)

Dataset	Source	Purpose
CNN Dailymail	TensorFlow Datasets	Model Generalization
Gigaword	TensorFlow Datasets	Model Generalization
Xsum	TensorFlow Datasets	Model Generalization
Amazon movie reviews	Stanford University Data	Domain Testing
Hotel Reviews	Kaggle	Domain Testing

All three datasets, CNN Dailymail, Gigaword, and Xsum, were used during the training process with various transformer architectures to determine which one had the best evaluation metrics. Xsum performed the best, so it was selected as the final dataset for the project. Resources related to the dataset used can be found in **APPENDIX E.1**.

### 7.2.3 Programming Language Selection

**Python** was used for the implementation of machine learning models and backend APIs in this study due to its readability, simplicity, versatility, and wide range of use cases, including web development, data analysis, scientific computing, and machine learning. Python also has a large and active community, providing ample resources and support. The availability of various libraries and frameworks, such as NumPy, pandas, and TensorFlow, made Python a powerful tool for data science and machine learning tasks. **TypeScript** (it's a superset of JavaScript) was chosen for the frontend development in order to display dynamic content and create a highly interactive and engaging user experience. A further analysis of this can be found at **APPENDIX E.2**

### 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 what's it used for in the project.

The author has conducted an analysis on the deep learning frameworks at **APPENDIX E.3**, an analysis on the UI frameworks at **APPENDIX E.4** and about the API frameworks at **APPENDIX E.5**

Table 23: Development Framework Utilized (Self-Composed)

<b>Framework</b>	<b>Reason for choosing</b>
React	ReactJS offers reusable components for efficient application development and has strong community support, making it a handy solution for developers.
Ant Design	Ant Design: React UI framework with pre-built components, customization, and tree-shaking compatibility for efficient frontend development.
Flask	Flask is a Python micro web framework for backend API development. It's lightweight, flexible, and offers a simple way to manage routing and processing with built-in server and extensions.
Optuna	Optuna: Python framework for efficient hyperparameter optimization with parallelization, ML library support, and automated features.
PyTorch	PyTorch is a Python-based machine learning framework with GPU support, pre-built neural network layers, and dynamic computation graphs. It's popular for its simplicity and powerful features in business and academia.

## 7.2.5 Libraries Utilized

Table 24: Libraries Used with Reasonings (Self-Composed)

<b>Library</b>	<b>Reasoning for selection</b>
Firebase	Backend services for mobile and web app dev.
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 offers pre-trained models and tools for NLP tasks.
NLTK	NLTK is a NLP library with tools for tokenization, stemming, part-of-speech tagging, and language model training/evaluation.

Rouge	Library to evaluate text summaries by comparing them to reference summaries.
Pandas	Pandas is a data manipulation and analysis library widely used in data science for handling numerical tables and time series data.
NumPy	NumPy is a Python library for scientific computing that supports large multi-dimensional numerical arrays and mathematical functions to operate on them.
Matplotlib & Seaborn	Used for creating static, animated, and interactive visualizations in Python
Gramformer	Hugging Face's API allows easy fine-tuning and use of GPT-3 models for various text generation tasks.
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 25: Ide's Used Along with Justifications (Self-Composed)

IDE	Justification for selection
VSCode	VSCode is a highly adaptable and efficient code editor with features such as debugging, Git integration, syntax highlighting, and customizable extensions.
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 26: 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 & model customization, 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 customization

The author conducted a study on automating hyperparameter search for model training and customization to avoid the time waste of manual tuning. Optuna was selected as the preferred framework due to its flexibility and user-friendliness.

Figure 12: Model customization parameter initialization

```
# Model customization parameters
DECODER_ATTENTION_HEADS_RANGE = [2, 3, 4, 6, 8, 12, 16]
DECODER_FFN_DIM_MIN = 1024
DECODER_FFN_DIM_MAX = 4096
DECODER_LAYERDROP_MIN = 0.0
DECODER_LAYERDROP_MAX = 0.3
DECODER_LAYERS_MIN = 4
DECODER_LAYERS_MAX = 12
```

The code initializes the range of model architecture parameters based on the transformer decoder. These parameters will be utilized in automated hyperparameter search to determine the optimal parameters for the data input.

Figure 13: Training hyper-parameter initialization

```

# Model training parameters
LR_MIN = 4e-5
LR_CEIL = 0.01
WD_MIN = 4e-5
WD_CEIL = 0.01
WARMUP_RATIO_MIN = 0.0
WARMUP_RATIO_MAX = 1.0
MIN_EPOCHS = 8
MAX_EPOCHS = 15
PER_DEVICE_EVAL_BATCH = 4
PER_DEVICE_TRAIN_BATCH = 4
MIN_BATCH_SIZE = 4
MAX_BATCH_SIZE = 6
NUM_TRIALS = 1
MAX_INPUT = 512
MAX_TARGET = 128
SAVE_DIR = 'checkpoints'
MODEL_NAME = 'bart-base_model'
TOKENIZER_NAME = 'bart-base_tokenizer'
```

The code segment provided initializes the range of hyperparameters used during the training phase, taking into consideration the data and customized model, in order to obtain an optimized output.

Figure 14: Model decoder customization

```

def objective(trial: optuna.Trial):
    # Load the base configuration for BART
    config = AutoConfig.from_pretrained(model_path)

    # Set the decoder parameters to values suggested by Optuna
    config.decoder_attention_heads = trial.suggest_categorical('decoder_attention_heads', DECODER_ATTENTION_HEADS_RANGE)
    config.decoder_ffn_dim = trial.suggest_int('decoder_ffn_dim', DECODER_FFN_DIM_MIN, DECODER_FFN_DIM_MAX)
    config.decoder_layerdrop = trial.suggest_uniform('decoder_layerdrop', DECODER_LAYERDROP_MIN, DECODER_LAYERDROP_MAX)
    config.decoder_layers = trial.suggest_int('decoder_layers', DECODER_LAYERS_MIN, DECODER_LAYERS_MAX)
    config.decoder_start_token_id = 1

    # Ensure that embed_dim is divisible by decoder_attention_heads
    if config.d_model % config.decoder_attention_heads != 0:
        config.decoder_attention_heads = config.d_model // 64

    # Instantiate the model with the modified configuration
    model = AutoModelForSeq2SeqLM.from_pretrained(model_path, config=config)
```

The presented code segment demonstrates the range of decoder parameters that are designated for searching using the Optuna framework. Once the best parameters are identified, they are configured and the model is initialized for the subsequent phase.

Figure 15: Model training hyperparameter tuning

```
# Specify the training arguments and hyperparameter tune every arguments which are possible to tune
training_args = Seq2SeqTrainingArguments(
    output_dir=SAVE_DIR,
    save_strategy="epoch",
    evaluation_strategy="epoch",
    learning_rate=trial.suggest_float("learning_rate", LR_MIN, LR_CEIL, log=True),
    weight_decay=trial.suggest_float("weight_decay", WD_MIN, WD_CEIL, log=True),
    num_train_epochs=trial.suggest_int("num_train_epochs", MIN_EPOCHS, MAX_EPOCHS),
    warmup_ratio=trial.suggest_float("warmup_ratio", WARMUP_RATIO_MIN, WARMUP_RATIO_MAX),
    per_device_train_batch_size=trial.suggest_int("per_device_train_batch_size", MIN_BATCH_SIZE, MAX_BATCH_SIZE),
    per_device_eval_batch_size=trial.suggest_int("per_device_eval_batch_size", MIN_BATCH_SIZE, MAX_BATCH_SIZE),
    save_total_limit=1,
    load_best_model_at_end=True,
    greater_is_better=True,
    predict_with_generate=True,
    run_name=MODEL_NAME,
    report_to="none",
)
```

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 for model training are considered for the hyperparameter searching this includes learning rate, weight decay, num of training epochs, warmup ratio, batch size.

Figure 16: Hyperparameter Results and Training Arguments (Self-Composed)

```
# Decoder-related hyperparameters:
decoder_attention_heads = study.best_params['decoder_attention_heads']
decoder_layerdrop = study.best_params['decoder_layerdrop']
decoder_ffn_dim = study.best_params['decoder_ffn_dim']
decoder_layers = study.best_params['decoder_layers']

# Training-related hyperparameters:
weight_decay = study.best_params['weight_decay']
warmup_ratio = study.best_params['warmup_ratio']
learning_rate = study.best_params['learning_rate']
num_train_epochs = study.best_params['num_train_epochs']
per_device_train_batch_size = study.best_params['per_device_train_batch_size']
per_device_eval_batch_size = study.best_params['per_device_eval_batch_size']

# BART-base configuration
config = AutoConfig.from_pretrained(model_path)

config.decoder_attention_heads = decoder_attention_heads
config.decoder_layerdrop = decoder_layerdrop
config.decoder_ffn_dim = decoder_ffn_dim
config.decoder_layers = decoder_layers

model = AutoModelForSeq2SeqLM.from_pretrained(model_path, config=config)
```

The above snippet demonstrates how to result of the hyperparameter search is used, to initialize the model.

Figure 17: Model Training (Self-Composed)

```

args = transformers.Seq2SeqTrainingArguments(
    'model-retraining',
    evaluation_strategy='epoch',
    save_total_limit=2,
    gradient_accumulation_steps=2,
    weight_decay=hyperparameters['weight_decay'],
    warmup_ratio=hyperparameters['warmup_ratio'],
    learning_rate=hyperparameters['learning_rate'],
    num_train_epochs=hyperparameters['num_train_epochs'],
    per_device_train_batch_size=batch_size, # due to GPU lim
    per_device_eval_batch_size=batch_size, # due to GPU lim
    predict_with_generate=True,
    eval_accumulation_steps=1,
    fp16=False
)

trainer = transformers.Seq2SeqTrainer(
    model,
    args,
    train_dataset=tokenize_data['train'],
    eval_dataset=tokenize_data['validation'],
    data_collator=data_collator,
    tokenizer=tokenizer,
    compute_metrics=compute_rouge
)

torch.cuda.empty_cache()

trainer.train()

```

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

### 7.3.2 Model Usage General & Domain Specific Users

Figure 18: General User Review Text Summarization (Self-Composed)

```

@app.route('/api/gensum/general', methods=['POST'])
def getGeneralizedSummary():
    try:
        data = request.get_json()
        review = data['review']
        inputs = generalized_tokenizer.encode(review, return_tensors='pt',
                                              max_length=MAX_INPUT, truncation=True)
        outputs = generalized_model.generate(inputs, max_length=150,
                                              min_length=40, length_penalty=2.0, num_beams=4, early_stopping=True)
        summary = generalized_tokenizer.decode(outputs[0],
                                                skip_special_tokens=True)

        sentimentAnalysisOutput = query({ "inputs": summary })
        sentiment, score = getOverallSentimentWithScore
        (sentimentAnalysisOutput)
        return {'summary': summary, 'sentiment': {
            'sentiment': sentiment,
            'score': score
        } }, 200
    except Exception as e:
        return {'message': str(e)}, 500

```

The provided code snippet above depicts an API endpoint that is responsible for generating text summaries (reviews) for regular users who are not required to create an account or have a specialized model allocated to them. Instead, a general model is employed.

Figure 19: Assigning A Specific Model for The New Domain User (Self-Composed)

```
@app.route('/api/gensum/domain-profile', methods=['POST'])
def createDomainUserProfile():
    try:
        data = request.get_json()
        userId = data['userId']

        folder_path = 'model/' + userId
        model_path = folder_path + '/' + MODEL_NAME
        tokenizer_path = folder_path + '/' + TOKENIZER_NAME

        if not os.path.exists(folder_path):
            os.mkdir(folder_path)

        generalized_model.save_pretrained(model_path)
        generalized_tokenizer.save_pretrained(tokenizer_path)

        return {'message': "Successfully created the model"}, 200
    except Exception as e:
        return {'message': str(e)}, 500
```

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 20: Domain Specific Text Review Summarization (Self-Composed)

```

@app.route('/api/gensum/domain-specific', methods=['POST'])
def getDomainSpecificSummary():
    try:
        data = request.get_json()
        review = data['review']
        userId = data['userId']

        folder_path = 'model/' + userId
        model_path = folder_path + '/' + MODEL_NAME
        tokenizer_path = folder_path + '/' + TOKENIZER_NAME

        if not os.path.exists(folder_path):
            return {'message': "Model not found"}, 404

        with open('encryption_key.key', 'rb') as file:
            key = file.read()

        model = transformers.AutoModelForSeq2SeqLM.from_pretrained(
            (model_path)
        )
        tokenizer = transformers.AutoTokenizer.from_pretrained(
            (tokenizer_path)
        )
        fernet = Fernet(key)

        inputs = tokenizer.encode(review, return_tensors='pt',
            max_length=MAX_INPUT, truncation=True)
        outputs = model.generate(inputs, max_length=150, min_length=40,
            length_penalty=2.0, num_beams=4, early_stopping=True)
        summary = tokenizer.decode(outputs[0], skip_special_tokens=True)

        sentimentAnalysisOutput = query({ "inputs": summary })
        sentiment, score = getOverallSentimentWithScore(
            (sentimentAnalysisOutput)
        )
        score = round(score, 4)
        score = str(score)

        db.collection('users').document(userId).collection('reviews').add
        ({
            'review': fernet.encrypt(review.encode()),
            'summary': fernet.encrypt(summary.encode()),
            'sentiment': fernet.encrypt(sentiment.encode()),
            'score': fernet.encrypt(score.encode()),
            'createdAt': firestore.SERVER_TIMESTAMP,
        })

        return {'summary': summary, 'sentiment': {
            'sentiment': sentiment,
            'score': score
        }, 200}
    
```

The code uses a domain-specific model to generate a summary, saves the input/output to a database, and analyzes sentiment using a pre-trained transformer from Hugging Face API.

### 7.3.3 Model Retraining

Figure 21: Fetching Related Data for Model Retraining (Self-Composed)

```

@app.route('/api/gensum/retrain', methods=['POST'])
def retrainDomainSpecificModel():
    try:
        data = request.get_json()
        newReviewSummaryData = []
        userId = data['userId']

        print('Finding user from database...')
        if not db.collection('users').document(userId).get().exists:
            return {'message': "User not found"}, 404

        userMetadata = db.collection('users').document(userId).get().to_dict()
        print('Get user metadata...', userMetadata)

        print('Getting receiver email...')
        email_receiver = userMetadata['email']

        print('Getting domain type...')
        domainType = userMetadata['type']

        isUseOtherData = data['isUseOtherData']
        print('Email trigger for retraining the model...')
        triggerEmailNotification("Retraining the gensum model", "Your model is
        preparing for retraining, you will be notified once the model is retrained",
        email_receiver)
        print('Notification sent to the user...')

        # Steps to be considered for retraining the model and dataset recreation
        # 1. By checking the isAccessible flag, we can decide whether to use the data
        # for model retraining, then we get all the data from the database which
        # isAccessible = true for the given domainType
        isAccessible = True
        print('Fetching data from the database...')
        if isUseOtherData == True:
            users = db.collection('users').where('type', '==', domainType).where
            ('isAccessible', '==', True).get()
            for user in users:
                reviews = db.collection('users').document(user.id).collection
                ('reviews').get()
                for review in reviews:
                    newReviewSummaryData.append(review.to_dict())
        else:
            user = db.collection('users').document(userId).get()
            if user.exists:
                reviews = db.collection('users').document(userId).collection
                ('reviews').get()
                for review in reviews:
                    newReviewSummaryData.append(review.to_dict())
    except Exception as e:
        print(f'An error occurred: {e}')
        return {'error': str(e)}, 500

```

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 model customization & 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 E.6**.

### 7.3.5 Generalized model training

The steps taken to create the generalized model after the top tier transformer model experimentations can be found at **APPENDIX E.7**

## 7.4 User interface

---

Screenshots of the final GUI are placed under **APPENDIX E.8**.

## 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.
- 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 were trained and tested on the datasets. Validation accuracy and loss for each model are displayed below to select the best-performing model.

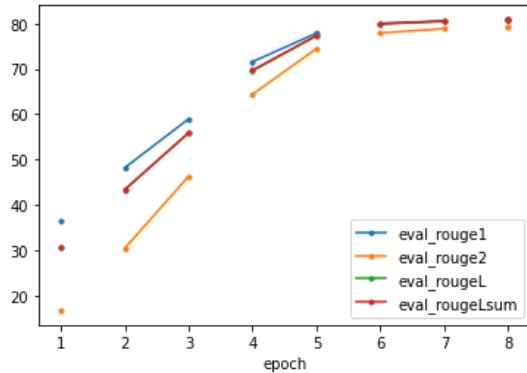


Figure 22: Validation Accuracy by number of epochs – bart model (*Self-Composed*)

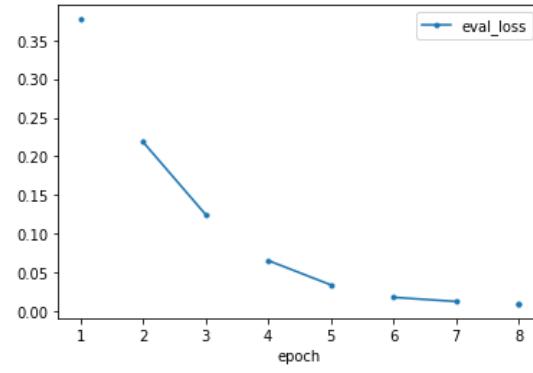


Figure 23: Validation Loss by number of epochs – bart model (*Self-Composed*)

The BART model was tested and the rouge scores increased with more epochs after hyperparameter tuning, yielding higher scores than previous research benchmarks. The validation loss decreased, indicating improved prediction accuracy.

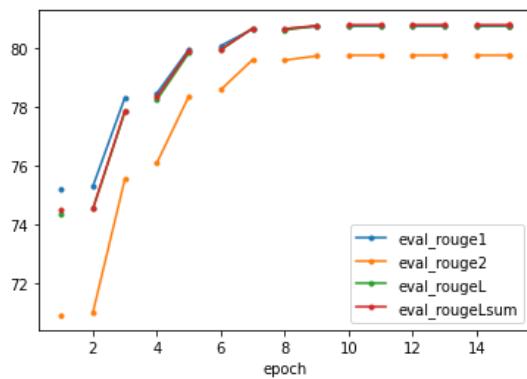


Figure 24: Validation Accuracy by number of epochs – t5 model (*Self-Composed*)

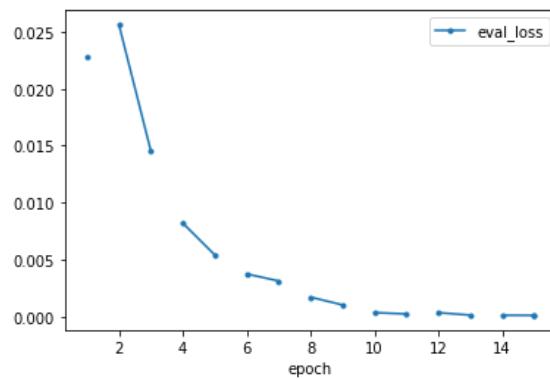


Figure 25: Validation Loss by number of epochs – t5 model (*Self-Composed*)

Testing results of the T5 model are shown, with all ROUGE scores increasing exponentially and validation loss decreasing exponentially with the number of epochs. However, the results suggest that the BART model performs slightly better than the T5 model.

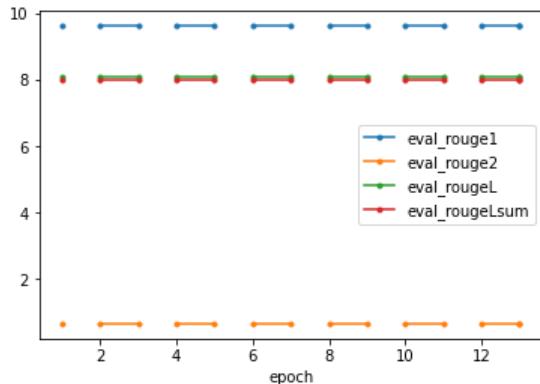


Figure 26: Validation Accuracy by number of epochs – Pegasus model (*Self-Composed*)

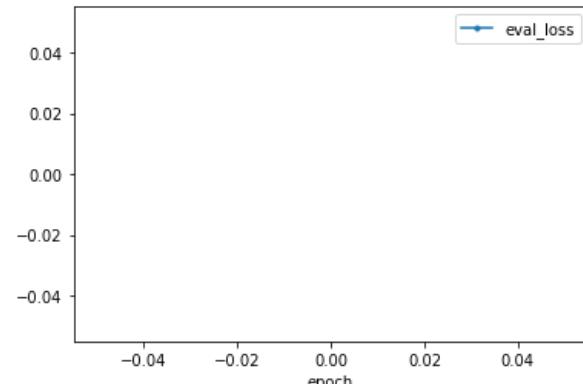


Figure 27: Validation Loss by number of epochs – Pegasus model (*Self-Composed*)

Pegasus model performed poorly in testing despite optimizing hyperparameters. Rouge scores remained low, and the validation loss graph showed no improvement in accuracy.

#### 8.4.2 Model Evaluation

The models were evaluated based on recommended criteria from literature review and author's proficiency, and the evaluation metrics were presented in the **LR Chapter** under the "**Evaluation**" topic. Previous research by Steinberger and Jezek (2009) suggests that ROUGE and its metric versions are more appropriate for achieving optimal results compared to BLEU among the primary scoring methods.

Table 27: 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

The evaluation shows that Bart performs better than the other two models, while Pegasus has the poorest performance, which could be due to various reasons despite optimization and suitable preprocessing. Bart and Pegasus have different model architectures, which could affect their understanding and representation of text data. One architecture may be more suitable for a specific dataset than the other.

## Domain Model Evaluation

The adaptive generalized solution was tested with two different domains - **movies** and **hotels** - to demonstrate the effectiveness of the approach.

Testing dataset:

1. **Movie Review:** The dataset was provided by Stanford University, which are amazon movie review data (SNAP, no date).
2. **Hotel Review:** The dataset was provided by Kaggle (Jonathanoheix, 2018).

The following are the evaluation results for each domain when tested with the corresponding dataset listed above after model retraining with unseen data.

Figure 28: Hotel Domain (Cinnamon Type) Evaluation

This screenshot shows a Google Cloud Firestore interface. The path in the top navigation bar is: Home > users > qs9zuV4NIJVoXy061C0vk82XNM2 > model-retraining-evaluations > vAOPNpmDrJDDeP2mnCgAQ. The right-hand panel displays a document with the following fields and their values:

created_at:	April 25, 2023 at 6:46:44 AM UTC+5:30
epoch:	"10"
eval_gen_len:	"22.3"
eval_loss:	"0.201"
eval_rouge1:	"85.76"
eval_rouge2:	"82.23"
eval_rougeL:	"84.63"
eval_rougeLsum:	"86.12"
eval_runtime:	"10.50"
eval_samples_per_second:	"62.0"
eval_steps_per_second:	"9.8"

Figure 29: Movie Domain (Scope Cinema Type) Evaluation

This screenshot shows a Google Cloud Firestore interface. The path in the top navigation bar is: Home > users > 8pMjn1T9AmOgQkRezKten5qFse2 > model-retraining-evaluations > uCk6bD5FRk3dj0Tt9FnT. The right-hand panel displays a document with the following fields and their values:

created_at:	April 25, 2023 at 6:53:41 AM UTC+5:30
epoch:	"7"
eval_gen_len:	"25.6"
eval_loss:	"0.053"
eval_rouge1:	"81.56"
eval_rouge2:	"83.63"
eval_rougeL:	"82.10"
eval_rougeLsum:	"82.76"
eval_runtime:	"15.35"
eval_samples_per_second:	"58"
eval_steps_per_second:	"8.9"

Adding new data improves the performance of the domain-specific model, indicating the need for each domain to have its own continually improving model. The results support the idea of generalization.

## 8.5 Benchmarking

---

The authors compared their benchmarking results with previous research on training transformers for abstractive text summarization. The results were presented in a table in the "**Benchmarking**" section of the literature review chapter.

Table 28: 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 table compares the benchmarking results of transformer models on the Xsum dataset between previous researchers (2019-2020) and the author's evaluation (2023), showing a significant improvement in performance due to model optimization.

## 8.6 Functional testing

---

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

- **11/11 test case passed.**

## 8.7 Module & integration testing

---

The high-level architecture diagram depicted in **Design Chapter** illustrated that the system's logic was divided into modules. To ensure that each module operates as intended, they underwent testing. The table give below indicates the module & integration testing results.

- **6/6 test cases passed.**

Table 29: Module &amp; integration testing

Test case	Module	Input	Expected result	Actual result	Status
1	NLP Text parser	Triggered when input review text	Preprocessing the input data	Preprocessing the input data.	Passed
2	Dataset recreation	Triggered during model retraining	Decrypts data from the database and creates new dataset	Decrypts data from the database and creates new dataset	Passed
3	Data Encryption	Triggered when domain user generates summary	Review text gets encrypted and stored into the database	Review text gets encrypted and stored into the database	Passed
4	Hyperparameter tuning	Triggered during model retraining	Using the new dataset, new hyperparameters are found along with the model customization	Using the new dataset, new hyperparameters are found along with the model customization	Passed
5	Model training	Triggered during model retraining	Using the new dataset and new hyperparameters, custom model retraining is triggered	Using the new dataset and new hyperparameters, custom model retraining is triggered	Passed
6	User profile creation	Triggered when domain user firstly signup	Assigns a copy of the generalized model for the user	Assigns a copy of the generalized model for the user	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 **SRS Chapter**. **APPENDIX F.2** details the specific breakdown of the non-functional testing that was executed.

- **Performance testing results**

CPU: 19% at 2.22 GHz	Internet required: Yes
RAM: 71% at 11.3 Gb	GPU: 14%
SSD: 3%	

- **GUI testing results**

Page	Performance	Accessibility	Best Practices	SEO
Landing page	78%	90%	100%	91%
Records page	93%	92%	100%	91%
User profile page	78%	91%	100%	91%

- **Maintainability testing results**

1. CodeQL: passed
2. CodeFactor: passed

- **Non-functional requirements testing results:**

1. 6/6 testcases passed.

## 8.9 Limitations of the testing process

Although the system underwent comprehensive testing, it has not been deployed or hosted yet, which makes it difficult to conduct production-grade load testing within the given timeframe.

## 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

---

The primary focus of this research to design and implement an adaptive generalized abstractive text summarization system using optimized transformers. Although the system produces text output, it employs numerical data for decision-making, rendering it a quantitative analysis for assessing its efficacy. Additionally, a qualitative analysis by domain experts is necessary to establish its reliability. This chapter will present the results of a **thematic analysis** of the feedback obtained through the aforementioned qualitative analysis.

The demonstration video of the research, which was utilized for assessments, can be located at the following URL: <https://youtu.be/V5TVzauaU6A>.

### 9.3 Evaluation Criteria

---

In preparation for the evaluation process, it is essential to establish unambiguous criteria to evaluate all aspects of the research thoroughly. The following table delineates the specific criteria that the author established before initiating the evaluation.

Table 30: Evaluation criteria

CR ID	Criterion	Purpose
CR1	Selection of research domain	To confirm the importance of the selected area of study, subject matter, and gap in research.
CR2	Research contribution	To assess the importance of the discovered results and their contributions to the existing pool of knowledge.

CR3	Research documentation quality	To verify that sufficient literature has been examined and that the entire research procedure has been recorded and presented with an acceptable level of quality.
CR4	Development approach	To verify that sufficient literature has been examined and that the entire research procedure has been recorded and presented with an acceptable level of quality.
CR5	Quantitative analysis of results	To verify the metrics employed to assess and analyze the outcomes generated by the research.
CR6	Limitations & Improvements	To recognize any constraints or shortcomings and opportunities for future research.
CR7	Usability & UI/UX	To confirm that the product created for demonstration purposes is user-friendly for end-users.

## 9.4 Self-Evaluation

The table presented below depicts the author's self-assessment based on the aforementioned standards.

Table 31: Self-evaluation of the author

CR ID	Author's self-evaluation
CR1	The selected research domain pertained to a technical application of significant utility, which holds promise for adoption by various users across diverse domains. However, given its relatively recent introduction, identifying domain experts presents a challenge.
CR2	This research has made significant contributions in several areas aimed at enhancing the performance of text summarization using transformers. Firstly, it has made technical contributions in developing a generalized system for text summarization that can enhance domain-specific performance. Secondly, the contribution to the domain being to improve performance for review summarization for the domain of movies.
CR3	The documentation has been produced to an exceptional level of quality, with Microsoft Word being utilized for all report documentation in order to ensure consistency across diagrams, tables, and internal links along with external links (video

	demonstration links). Nevertheless, LaTeX has been employed for the creation of both review papers and research papers.
CR4	Considerable resources have been dedicated to gathering and preparing data to achieve optimal outcomes. Extensive experimentation has been conducted on several high-level transformer structures with diverse comprehensive datasets. Furthermore, state-of-the-art technologies and software have been employed throughout the process.
CR5	The assessment outcomes of the model were produced within the Google Colab notebooks that were utilized for training the model.
CR6	After finalizing the system, the author recognized a few areas where enhancements and improvements could be made, and suggested these as potential areas for future research.
CR7	The user interface and experience of the final product has been designed in a way that is both usable and minimalistic.

## 9.5 Selection of Evaluators

Evaluators were selected based on grouping, which was necessary to receive feedback on all aspects of the project. The table below illustrates the breakdown of the groups. The list of evaluators considered for the project can be found at [APPENDIX G.1](#)

Table 32: Categorization of selected evaluators

CAT ID	Category
CAT1	Experts who have conducted research in the areas of Text Summarization Systems, Data Science, Data Engineering, and Machine Learning.
CAT2	Experts who possess expertise in NLP (Natural Language Processing) and transformers.
CAT3	Possible end users of the application

## 9.6 Evaluation Results & Expert Opinions

The opinions and feedback from the experts mentioned earlier were analyzed using thematic analysis. The results of this analysis are presented in the table below.

Table 33: Thematic analysis of expert feedback

CR ID	CAT ID	Theme	Summary of Opinions
CR1	CAT1	Text Summarization System choice gap	The utilization of text summarization can be highly advantageous for businesses, particularly in the realm of ecommerce where reviews play a crucial role in driving sales.
		Technical research gap	The adoption of transformers as an approach for the domain is preferred over conventional methods, and further investigation into its optimization can prove advantageous.
	CAT2	Domain research gap	The identified objective is to bridge the gap in performance optimization and develop a solution that is more generalized.
	CAT3	Usefulness of domain research.	As it is an adaptive solution, there currently exists no generalized application to address the performance issue in this new domain application.
CR2	CAT1	Contribution of technology to text summarization systems.	The issues with conventional approaches have been clearly identified, and there is a clear understanding of the methods required to resolve this problem.
	CAT2 & CAT3	Domain Contribution	The proposed solution is a valuable contribution since there is currently no such solution available, and creating it as a plugin for businesses could provide significant benefits.
CR3	CAT1	Displaying content	The utilization of Microsoft Word documents to their fullest potential was acknowledged, and the

			research conducted was thorough, including the presentation of statistical data.
	CAT1 & CAT2 & CAT3	The problem-solving approach used	A comprehensive analysis of various angles was undertaken to approach the solution, and a more logical approach has been adopted.
CR4	CAT1	Data preprocessing	Significant effort has been devoted to data preprocessing, as experimentation was conducted using multiple datasets, requiring additional time and resources.
	CAT1 & CAT2	Experimentation of transformers with datasets for generalization	The selection of transformer models for experimentation was thoroughly considered and adequately justified.
	CAT1 & CAT2 & CAT3	Development approach for adaptive generalization	The chosen direction was deemed suitable by all, and a systematic and methodical approach was employed within the given timeframe.
CR5	CAT1 & CAT2 & CAT3	Analysis of the text summarization model	The current evaluation, using the ROUGE metric, indicates a significant improvement in performance compared to previous benchmarking results, with a performance gap exceeding 50%. ROUGE is considered the most suitable metric for evaluating such models.
CR6	CAT1 & CAT2	Model modifications: hybrid approach or customizing layers	The author attempted to explore the use of hybrid models and customization of existing models, but due to the limited timeframe, it was challenging to implement these approaches effectively.
	CAT3	New feature additions	Prior to generating a summary, it would be advisable to incorporate a paraphrasing function, considering the possibility that users might input text with grammatical errors.

CR7	CAT1 & CAT2	Great to have a working application for the model	As the model appears to be functioning well during testing, developing a functional application for users would be a wise choice.
	CAT 3	Minimalistic UI design	The design of the UI is very simple and clean, where most of the users found it attractive to use.

## 9.7 Limitations of Evaluation

As there is a scarcity of experts in the research domain (Transformers), only a limited number of expert opinions were obtained. This was apparent during the requirement gathering phase, as the author reached out to numerous ML/DL domain experts for more insights on the research concept, but only a few were able to provide relevant information. Therefore, only the experts specified in **APPENDIX G.1** possessed the required knowledge and expertise to provide constructive feedback. Nevertheless, traditional ML/DL experts were also approached to obtain feedback, as it could still prove to be beneficial.

## 9.8 Evaluation of Functional requirements

The breakdown of implementation of functional requirements is provided in **APPENDIX G.2**. It's clear that **16 out of 17** functional requirements have been completed (**94.11% implemented**)

## 9.9 Evaluation of non-functional requirements

The breakdown of completed non-functional requirements and the attainment of the design goals are presented in **APPENDIX G.3**. It's clear that **6.5 out of 7** non-functional requirements have been completed which is of (**92.9% implemented**)

## 9.10 Chapter summary

In this chapter, the implemented system was evaluated by establishing evaluation criteria that comprehensively covered all aspects of the system. The author conducted self-evaluation and obtained feedback from evaluators, which was analyzed through thematic analysis. The evaluation report presented the achievement of the proposed design goals, along with the evaluation of functional and non-functional requirements.

# CHAPTER 10. CONCLUSION

---

## 10.1 Chapter Overview

---

This chapter concludes the research project, highlighting the core functionality of the MVP's implementation. It reviews the project's achievements, obstacles encountered, and documents the author's prior knowledge and modules of the program that supported the project, 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 adaptive generalized transformer architecture from a range of popularly used architectures by automating the processing of model customization and hyperparameter tuning for optimization, therefore obtaining the recommended architecture's optimum performance.”*

The aim of the research was achieved by **designing, developing, and evaluating** a performance adaptive generalized transformer. The core functionality was automated to meet the project requirements, and the evaluation of the work is presented in the implementation chapter.

### 10.2.2 Achievement of Objectives

**APPENDIX H.1** lists the achievement status of the research objectives mentioned in Chapter 01, with "Completed" next to successfully completed tasks and "Incomplete" next to unfinished ones.

## 10.3 Utilization of knowledge from the degree

---

Table 34: Utilization of Knowledge Gained from The Course

Module(s)	Utilized Knowledge
Machine Learning	Having a solid grasp of the concept of data collection and preprocessing, as well as machine learning model training, proved to be instrumental in creating the models used in this research project.

Applied AI	The author gained a comprehensive understanding of the theoretical principles behind ML models through their knowledge of how algorithms interact during the model-building process.
Software Development Group Project	The module acted as a trial run for the Final Year Project, giving students the basic skills needed to plan, conduct and evaluate research, thus providing them with the necessary confidence and knowledge for their final year project.
Object Oriented Programming	The understanding of object-oriented programming and the significance of creating classes helped improve the development aspect of the project.
Python Programming (PP1)	This project involves the use of Flask, a web framework for the Python programming language. The PP1 module introduced the author to working with Python.
Database Systems	The understanding of using queries to communicate with the database from the webserver system facilitated read and write operations for the project.
Web Design & Development	The concepts learned from this module were used to develop the user interface (UI) of the prototype, and the foundational knowledge of using HTML, CSS, and JS was instrumental in transitioning to working with more advanced frameworks like React.

## 10.4 Use of Existing Skills

- **Full-Stack Web Development** – The author utilized advanced technologies for a full stack web development project while working on several R&D projects during their internship at 99x.
- **Machine Learning / Deep Learning** – During the author's internship, they were involved in several R&D projects related to data science and also utilized various online learning resources to develop their skills in machine learning.
- **Documentation Writing** – The author became proficient in project documentation during their internship and while working on the SDGP module report.

## 10.5 Use of New Skills

---

- **Text Summarization Systems** – The author lacked prior experience in text summarization systems and thus conducted research on various techniques using publicly accessible online resources from YouTube, GitHub, Google Colab, and others.
- **Data Preprocessing Techniques** – The project domain being text summarization, the author had to acquire new text preprocessing techniques to ensure the data was meaningful.
- **Hyperparameter Optimization** – The author explored and experimented with hyperparameter tuning frameworks to automate the hyperparameter search, referring to tutorials and technical articles to implement this in the project.

## 10.6 Achievement of learning outcomes

---

In APPENDIX H.2, there is a presentation of the accomplishment of the learning objectives.

## 10.7 Problems and Challenges faced

---

Table 35: Mitigations to Problems and Challenges Faced

Problem/Challenge	Mitigation
Significant training time and computing resource limitations.	The author uses Google Colab to train transformer-based models due to the high GPU power demands of the training process.
Limited experts for the domain	The author conducted interviews with domain experts for requirement gathering by reaching out to them via LinkedIn since there were limited opportunities for in-person interactions with domain experts.
Power outages were frequent, causing battery and internet connectivity issues.	The author persevered with the project despite power outages by working late or early in the morning at co-working spaces.

## 10.8 Deviations

---

The author's original goal was to develop an optimized solution for ***movie review summarization*** using transformers. However, after discussions with supervisors, it was decided that the author's technical contribution of automated hyperparameter tuning for transformers was not significant enough. As a result, the idea of creating a ***performance adaptive generalized solution*** was considered to continue the research implementation. The Gantt chart plan at the beginning and end of the project can be located in **APPENDIX H.3**.

It's great to see that the author was able to complete the core functionalities for the prototype, even with the challenge of time constraints mentioned by the experts interviewed for the requirement gathering. It shows the author's determination and ability to overcome obstacles to achieve the project's goals.

## 10.9 Limitations of the research

---

- The author tried to implement additional performance improvements like model hybridization & model layer customization after completing the core implementation. However, due to the limited time available, the amount of research required for transformer hybridization was significant, which prevented the author from continuing.
- The author did not explore various other transformer models for abstractive text summarization due to limitations of GPUs.

## 10.10 Future enhancements

---

- It's suggested that transformer hybridization can be used to improve the performance of text summarization models in the future.
- The author suggests that including text paraphrase models for user reviews would be a sensible approach since there is a potential that user reviews entered are not always accurate.
- The author suggests applying key word extraction for the sentiment classification of the review summary in order to identify which key words contributed to the sentiment classification and to help domain users improve their service.

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

---

The author made successful contributions to the problem domain of movie review summarization by implementing the system and making deviations from the initial goal to create a generalized solution. They also made technical contributions to improve the system's performance, but due to limited time, they were unable to explore all possibilities. Finally, the author made additional contributions to bring the research project to a conclusion.

### 10.11.1 Domain Contributions

The author was 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 addressed 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 which is adaptive with respect to the domain.

*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.
4. Integrating email notifications to inform domain users about the status of model retraining.

## 10.12 Implementation code

---

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

### 10.13 Concluding remarks

The study designed, built, and evaluated an adaptive generalized abstractive text summarization system using optimized transformers and automated hyperparameter tuning and model retraining. The author met the goals and objectives of the project and discussed the role of their prior knowledge and academic background in supporting the research. The author acquired new skills during the project and faced challenges and obstacles. They discussed deviations taken and limitations of the research, as well as opportunities for future improvements. The research contributed to the body of knowledge with domain, technical, and additional contributions made. The author plans to publish a research paper based on their findings.

## REFERENCES

---

- Abolghasemi, M., Dadkhah, C. and Tohidi, N. (2022). HTS-DL: Hybrid Text Summarization System using Deep Learning. 2022 27th International Computer Conference, Computer Society of Iran (CSICC). 23 February 2022. Tehran, Iran, Islamic Republic of: IEEE, 1–5. Available from <https://doi.org/10.1109/CSICC55295.2022.9780395> [Accessed 26 October 2022].
- Alsaqer, A.F. and Sasi, S. (2017). Movie review summarization and sentiment analysis using rapidminer. 2017 International Conference on Networks & Advances in Computational Technologies (NetACT). July 2017. Thiruvananthapuram, India: IEEE, 329–335. Available from <https://doi.org/10.1109/NETACT.2017.8076790> [Accessed 10 October 2022].
- Abadi, M. et al. (1970) [PDF] tensorflow: A system for large-scale machine learning: Semantic scholar, ArXiv. Available at: <https://www.semanticscholar.org/paper/TensorFlow%3A-A-system-for-large-scale-machine-Abadi-Barham/46200b99c40e8586c8a0f588488ab6414119fb28> (Accessed: April 25, 2023).
- aws.amazon.com. (2022). Hyperparameter optimization for fine-tuning pre-trained transformer models from Hugging Face | AWS Machine Learning Blog. [online] Available at: <https://aws.amazon.com/blogs/machine-learning/hyperparameter-optimization-for-fine-tuning-pre-trained-transformer-models-from-hugging-face/> [Accessed 9 Nov. 2022].
- Barna, N.H. and Heickal, H. (2022). An Automatic Abstractive Text Summarization System. Dhaka University Journal of Applied Science and Engineering, 6 (2), 39–48. Available from <https://doi.org/10.3329/dujase.v6i2.59217>.
- Boorugu, R., Ramesh, G. and Madhavi, K. (2019). Summarizing Product Reviews Using Nlp Based Text Summarization. International Journal of Scientific & Technology Research Volume, 8 (10), 1127–1133.
- Brasoveanu, A.M.P. and Andonie, R. (2020). Visualizing Transformers for NLP: A Brief Survey. 2020 24th International Conference Information Visualisation (IV). September 2020. Melbourne, Australia: IEEE, 270–279. Available from <https://doi.org/10.1109/IV51561.2020.00051> [Accessed 2 November 2022].

Chauhan, N. et al. (1970) [PDF] implementation of database using Python Flask Framework: Semantic scholar, International Journal of Engineering and Computer Science. Available at: <https://www.semanticscholar.org/paper/Implementation-of-database-using-python-flask-Chauhan-Singh/7de6ad96c6f431cc9084be0a0d788149ad267ffc> (Accessed: April 25, 2023).

Canesche, M. et al. (1970) Google colab CAD4U: Hands-on cloud laboratories for digital design: Semantic scholar, 2021 IEEE International Symposium on Circuits and Systems (ISCAS). Available at: <https://www.semanticscholar.org/paper/Google-Colab-CAD4U%3A-Hands-On-Cloud-Laboratories-for-Canesche-Bragan%C3%A7a/6d0cbb5d66d8dc80a447d68b86fa1bddf0498494> (Accessed: April 25, 2023).

Etemad, A.G., Abidi, A.I. and Chhabra, M. (2021). A Review on Abstractive Text Summarization Using Deep Learning. 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO). 3 September 2021. Noida, India: IEEE, 1–6. Available from <https://doi.org/10.1109/ICRITO51393.2021.9596500> [Accessed 10 October 2022].

Gupta, A. et al. (2021). Automated News Summarization Using Transformers. ArXiv, abs/2108.01064.

Gupta, V. and Lehal, G.S. (2010). A Survey of Text Summarization Extractive Techniques. Journal of Emerging Technologies in Web Intelligence, 2 (3), 258–268. Available from <https://doi.org/10.4304/jetwi.2.3.258-268>.

Jonathanoheix (2018) Sentiment analysis with Hotel Reviews, Kaggle. Kaggle. Available at: <https://www.kaggle.com/code/jonathanoheix/sentiment-analysis-with-hotel-reviews/input> (Accessed: April 25, 2023).

Joy, J. and Selvan, M.P. (2022). A comprehensive study on the performance of different Multi-class Classification Algorithms and Hyperparameter Tuning Techniques using Optuna. 2022 International Conference on Computing, Communication, Security and Intelligent Systems (IC3SIS). 23 June 2022. Kochi, India: IEEE, 1–5. Available from <https://doi.org/10.1109/IC3SIS54991.2022.9885695> [Accessed 24 October 2022].

Khan, A. et al. (2020). Movie Review Summarization Using Supervised Learning and Graph-Based Ranking Algorithm. *Computational Intelligence and Neuroscience*, 2020, 7526580. Available from <https://doi.org/10.1155/2020/7526580>.

Kirmani, M. et al. (2019). Hybrid Text Summarization: A Survey. In: Ray, K. Sharma, T.K. Rawat, S. et al. (eds.). *Soft Computing: Theories and Applications. Advances in Intelligent Systems and Computing*. Singapore: Springer Singapore, 63–73. Available from [https://doi.org/10.1007/978-981-13-0589-4\\_7](https://doi.org/10.1007/978-981-13-0589-4_7) [Accessed 1 November 2022].

Kouris, P., Alexandridis, G. and Stafylopatis, A. (2019). Abstractive Text Summarization Based on Deep Learning and Semantic Content Generalization. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019. Florence, Italy: Association for Computational Linguistics, 5082–5092. Available from <https://doi.org/10.18653/v1/P19-1501> [Accessed 24 October 2022].

Lackermair, G., Kailer, D. and Kanmaz, K. (2013). Importance of Online Product Reviews from a Consumer's Perspective. *Advances in Economics and Business*, 1 (1), 1–5. Available from <https://doi.org/10.13189/aeb.2013.010101>.

Lin, C.-Y. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. 8.

Liu, X. and Wang, C. (2021). An Empirical Study on Hyperparameter Optimization for Fine-Tuning Pre-trained Language Models. Available from <http://arxiv.org/abs/2106.09204> [Accessed 24 October 2022].

McKinney, W. (2017) Python for Data Analysis: Data wrangling with pandas, NumPy, and ipython, Google Books. "O'Reilly Media, Inc.". Available at: [https://books.google.com/books/about/Python\\_for\\_Data\\_Analysis.html?id=BCc3DwAAQBAJ](https://books.google.com/books/about/Python_for_Data_Analysis.html?id=BCc3DwAAQBAJ) (Accessed: April 25, 2023).

Mahajan, R. et al. (2021). Text Summarization Using Deep Learning. *International Research Journal of Engineering and Technology (IRJET)*, 08 (05th May 2021), 1737–1740.

McAuley, J.J. and Leskovec, J. (2013). From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. *Proceedings of the 22nd international conference on World Wide Web*, 2013, 519–528. Available from <https://doi.org/10.1145/2487575.2488142> [Accessed 24 October 2022].

Wide Web - WWW '13. 2013. Rio de Janeiro, Brazil: ACM Press, 897–908. Available from <https://doi.org/10.1145/2488388.2488466> [Accessed 19 November 2022].

Mukherjee, R. et al. (2020). Read what you need: Controllable Aspect-based Opinion Summarization of Tourist Reviews. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 25 July 2020. 1825–1828. Available from <https://doi.org/10.1145/3397271.3401269> [Accessed 10 October 2022].

Najmuddin, S., Atal, Z. and Ziar, R.A. (1970) [PDF] comparative analysis of power consumption of the linux and its distribution operating systems vs windows and mac operating systems.: Semantic scholar, Kardan journal of engineering and technology. Available at: <https://www.semanticscholar.org/paper/Comparative-Analysis-of-Power-Consumption-of-the-vs-Najmuddin-Atal/ae640a6873317596bd8869f0a2155e7bc4d5e899> (Accessed: April 25, 2023).

Neyshabur, B. et al. (2017). Exploring Generalization in Deep Learning. undefined. Available from <https://www.semanticscholar.org/reader/d53fb3feeeab07a0d70bf466dd473ec6052ecc07> [Accessed 9 November 2022].

Pai, A. (2014). Summarizer Using Abstractive and Extractive Method. International Journal of Engineering Research, 3 (5), 5.

Pizam, A. and Ellis, T. (1999). Customer satisfaction and its measurement in hospitality enterprises. International Journal of Contemporary Hospitality Management, 11 (7), 326–339. Available from <https://doi.org/10.1108/09596119910293231>.

Paszke, A. et al. (1970) [PDF] automatic differentiation in pytorch: Semantic scholar, [PDF] Automatic differentiation in PyTorch | Semantic Scholar. Available at: <https://www.semanticscholar.org/paper/Automatic-differentiation-in-PyTorch-Paszke-Gross/b36a5bb1707bb9c70025294b3a310138aae8327a> (Accessed: April 25, 2023).

Paszke, A. et al. (1970) [PDF] pytorch: An imperative style, high-performance deep learning library: Semantic scholar, [PDF] PyTorch: An Imperative Style, High-Performance Deep Learning Library | Semantic Scholar. Available at: <https://www.semanticscholar.org/paper/PyTorch%3A-An-Imperative-Style%2C-High-Performance-Deep-Paszke-Gross/3c8a456509e6c0805354bd40a35e3f2dbf8069b1> (Accessed: April 25, 2023).

- Relan, K. (1970) Deploying flask applications: Semantic scholar, Building REST APIs with Flask. Available at: <https://www.semanticscholar.org/paper/Deploying-Flask-Applications-Relan/f55aa0f7c950e7471068e5fe9f72f6221ca6e3f0> (Accessed: April 25, 2023).
- Shi, T. et al. (2020). Neural Abstractive Text Summarization with Sequence-to-Sequence Models. Available from <http://arxiv.org/abs/1812.02303> [Accessed 10 October 2022].
- Shorten, C. and Khoshgoftaar, T.M. (2019). A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*, 6 (1), 60. Available from <https://doi.org/10.1186/s40537-019-0197-0>.
- Socher, R., Bengio, Y. and Manning, C.D. (2012). Deep Learning for NLP (without Magic). Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts. July 2012. Jeju Island, Korea: Association for Computational Linguistics, 5. Available from <https://aclanthology.org/P12-4005> [Accessed 2 November 2022].
- Steinberger, J. and Jezek, K. (2009). Evaluation Measures for Text Summarization. *Comput. Informatics*, 28 (2), 251–275.
- Srivastava, A. (1970) Django, the python web framework: Semantic scholar, INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT. Available at: <https://www.semanticscholar.org/paper/Django-%2C-The-Python-Web-Framework-Srivastava/231c909120972252411f85b4bb8c13657ede82c6> (Accessed: April 25, 2023).
- Verma, A. et al. (1970) Web application implementation with Machine Learning: Semantic scholar, 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM). Available at: <https://www.semanticscholar.org/paper/Web-Application-Implementation-with-Machine-Verma-Kapoor/9f9dc62da64b368e2503eb312b8ef9b687bc5b1a> (Accessed: April 25, 2023).
- Virtanen, P. et al. (1970) [PDF] scipy 1.0: Fundamental algorithms for scientific computing in python: Semantic scholar, Nature Methods. Available at: <https://www.semanticscholar.org/paper/SciPy-1.0%3A-fundamental-algorithms-for-scientific-in-Virtanen-Gommers/f0d35b37fec26c3f1ed09253cbb9304fb62208d1> (Accessed: April 25, 2023).

Web data: Amazon Movie Reviews (no date) SNAP. Available at:  
<https://snap.stanford.edu/data/web-Movies.html> (Accessed: April 25, 2023).

Waranashiwar, J. and Ukey, M. (1970) Ionic framework with angular for hybrid app development, Semantic Scholar. Available at: <https://www.semanticscholar.org/paper/Ionic-Framework-with-Angular-for-Hybrid-App-Waranashiwar-Ukey/92847c13951213e701e5b3a834e7e45b5c873644> (Accessed: April 25, 2023).

Wahyudi, E.T., Erwin, A. and Lim, C. (1970) [PDF] development of API middleware and mobile application for a job marketplace by using restful API and mobile development framework: Semantic scholar, [PDF] Development of API Middleware and Mobile Application for a Job marketplace by Using RESTful API and Mobile Development Framework | Semantic Scholar. Available at: <https://www.semanticscholar.org/paper/Development-of-API-Middleware-and-Mobile-for-a-Job-Wahyudi-Erwin/03166228259cd38cc5c433b77e7d2510fed72764> (Accessed: April 25, 2023).

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. 2020. Online: Association for Computational Linguistics, 38–45. Available from <https://doi.org/10.18653/v1/2020.emnlp-demos.6> [Accessed 10 October 2022].

Zhang, J. et al. (2020). PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. Available from <http://arxiv.org/abs/1912.08777> [Accessed 18 October 2022].

Zhou, K. et al. (2021). Domain Generalization with MixStyle. undefined. Available from <https://www.semanticscholar.org/reader/4f6eafafc9563a5b904535078df7e74afe39ef59> [Accessed 5 December 2022].

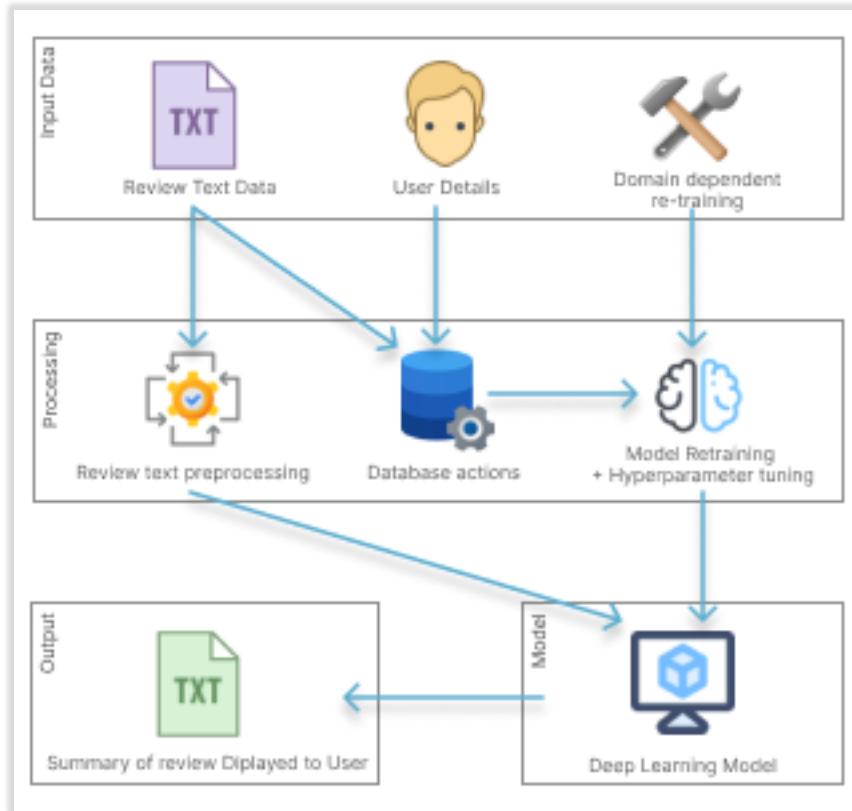
## APPENDIX A – INTRODUCTION

---

### A.1. Prototype feature diagram

The illustration presented underneath illustrates the preliminary characteristic diagram suggested in the proposal manuscript.

Figure 30: Prototype Feature Diagram (Self-Composed)



### A.2. Project scope

#### In scope

- **System Generalization** – Creating a generalized system to be able to adapt to any domain.
- **Dataset reconstruction** – Reconstructing the dataset to a format structure which can be used for data preprocessing + model training.
- **Model refinement on hyperparameter tuning** – Performing hyperparameter tuning on the top tier transformer architecture models.

- ***Evaluating the models*** – Evaluating all the architectures using appropriate metrics to filter out the best architecture from the rest.
- ***API integration development*** – REST API endpoints will be created to serve/call the final chosen model for interactions.
- ***GUI development*** – A graphical user interface will be developed; therefore, the end user will be able to perform abstractive text summarization and get visual results.

## Out scope

- ***Limited architecture explored*** – The system will only be explored with few of the top tier architectures (roughly around 3 or 5 maximum), and will not be exploring more than that.
- ***Only single model integration*** – The final model which outperforms the rest with the best set of hyperparameters will be used as the summary generation model, options to select other architectures explored with their hyperparameters aren't included.

## APPENDIX B – LITERATURE REVIEW

---

### B.1. Related work in abstractive text summarization

Table 36: Related Work in Abstractive Text Summarization

Ref.	Summary	Contribution	Limitations
(Khan, Gul, Zareei, et al., 2020)	An automatic approach to summarize lengthy movie reviews and allow users to quickly recognize the positive and negative aspects of a movie.	Worked on feature extraction and converting reviews into vector space, followed by the Naïve Bayes machine learning algorithm used for review classification, using an undirected weighted graph based ranking algorithm to rank score for each review sentence in graph. Finally, the top ranked sentences are chosen based on highest rank scores to produce <i>extractive summary</i> .	To use advanced deep learning approaches.
(Boorugu, Ramesh and Madhavi, 2019)	Using customer reviews on products when making purchasing decisions to give a proper summarization of the reviews to the customer, so that he doesn't need to go through all the reviews to figure out if the product is what he is looking for and save time.	Using seq2seq model for summarization along with attention mechanism for increased accuracy, also using word embedding model Concept net Number batch which is better than Glove. Finally, using a 1D convolutional layer followed by max pooling layer, LSTM layer and then at the end a fully connected layer.	Focused on improving the accuracy by using the latest models in the field of text summarization. By using transformers

			architecture, we could improve this.
(Mukherjee et al., 2020)	A solution for generating personalized aspect-based opinion summaries from large collections of online tourist reviews, also able to customize the attributes of the summary based on the user's interest.	Using an Integer Linear Programming (ILP [Unsupervised method]) based extractive technique to select an informative subset of opinions around the identified aspects. Evaluate and compare the summaries using ROUGE based metrics and obtain competitive results.	Motive for the need to create tourist review dataset for our experiments. The need for also experimenting with the data of lesser known places (Tourist locations)
(Gupta et al., 2021)	A comprehensive comparison of a few transformer architecture based pre-trained models for text summarization.	Using the pretrained models such as Pipeline BART, BART modified, T5 and PEGASUS to work with the text summarization. Evaluation metrics we done using the ROUGE Scores.	Future work should focus on building more robust models which can further extend the algorithm to create summaries of variable length and apply for multi-document summarization.

(Mahajan et al., 2021)	Generate a text summary along with proper grammar and no repeated words using the Encoder-Decoder model with the attention layer	Developed an encoder-decoder model using Gated Recurrent Units and trained the model to generate abstractive summary from an article.	Real time training required if this is used in production, in order to train with the latest articles with time.
(Etemad, Abidi and Chhabra, 2021)	Experimenting the text summarization domain with deep learning approaches and finding which performs the best, from RNN, CNN, Transformers etc....	Experimenting with RNN based models' architectures, working with pre-trained transformer-based model architectures. Finally, using evaluation metrics such as BLEU and ROUGE to evaluate the models	NA

## APPENDIX C – SRS

---



---

### C.1. Requirement Elicitation Methodologies

Table 37: 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 38: Interview Thematic Analysis (Self-Composed)

Theme	Code
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

Table 39: Interview evidence

Theme	Evidence
Data handling	“Make sure to find a reliable and validated dataset, since a generalized model is to be built initially”. “Will the system support multi-languages, or only English language”
Transformer architectures	“Using transformers for NLP is a very good approach” “All these models get updated frequently, therefore keep an eye on the most recent versions during the experimentations”
Generalization	“Using a scalable datastore would be preferable since the application will be used in a very large scale due to the concept of generalization”
Research scope	“Idea is good but challenging with the timeframe since there are many other components as well”
Hyperparameter tuning	“There are several tools where you could integrate to perform the automation with”
Hybrid transformers	“Using ensemble approach is another way where you can explore to increase performance of the existing system” “However, focus on the main scope of the project first and only if time permits go ahead with this”
Custom transformers	“Try to do some customization with the models to improve its performance therefore it will add more weightage for the research”
Prototype	“It will be nice to see the outcome on a web app or a mobile app”
Business benefits	“Movie domain, Tourism, Ecommerce, Book would benefit a lot, since they are mostly the active domains all the time”
Evaluations	“Keep the evaluation scope limited to around 2 or 3 domains, else it will consume a lot of time”

Table 40: Interview participant details

ID	Affiliation	Expertise related to the research
P1	PhD Research Student in Computational Linguistics	NLP
P2	Machine Learning Expertise   Lecturer with PhD	ML and Neural Networks
P3	NLP Researcher	NLP
P4	Software Architect	Algorithms
P5	Software Architect & ML Researchers	ML & Algorithms
P6	VP Innovations, Software Engineer	ML & Neural Networks
P7	Lecturer with MSc	NLP

### C.3. Use case descriptions

Table 41: Usecase mappings (*Self-Composed*)

Use case Id	Use case name
UC01	Input Review
UC02	Create Profile
UC03	Retrain Model
UC04	Search New Hyperparameters
UC05	Create Model
UC06	Prepare Dataset
UC07	View Summary
UC08	Generate Summary
UC09	Store Data
UC10	Delete reviews

Table 42: Use case description UC:01 (*Self-Composed*)

Use Case Name	Input Review
Use Case Id	UC:01
Description	Requested the user to input a text review
Primary Actor	General User, Domain Specific User
Pre-Conditions	Domain Specific user needs to be login in before this action
Extended use cases	None
Included use cases	None
Trigger	A user selects the text input field to enter text review.
Main flow	The general user clicks on the input field to enter the review text, if it's a domain specific user then user needs to login into the application for this action
Alternative flows	None
Expectational flows	Displays an error message if the network request fails (server is down, or internet issues from client).
Post Conditions	None

Table 43: Use case description UC:02 (*Self-Composed*)

Use Case Name	Create Profile
Use Case Id	UC:02
Description	Domain users will be able to create a unique profile to manage their content
Primary Actor	Domain Specific User
Pre-Conditions	None
Extended use cases	None
Included use cases	None
Trigger	The domain user signups an account with in the system

Main flow	Actor	System
	<ol style="list-style-type: none"> <li>1. The domain user navigates to the sign-in page.</li> <li>2. The domain user clicks on sign in, to register their self or login to the application</li> </ol>	<ol style="list-style-type: none"> <li>3. Create a new user in the database and notify the user.</li> </ol>
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 message displayed.	

Table 44: Use case description UC:10 (*Self-Composed*)

Use Case Name	Delete reviews	
Use Case Id	UC:10	
Description	Domain users will only be able to perform this action to manage their own data reviews and delete	
Primary Actor	Domain Specific User	
Pre-Conditions	Domain user should be logged into the application	
Extended use cases	UC:09	
Included use cases	None	
Trigger	Clicking on the delete action button on the review card list	
Main flow	Actor	System
	<ol style="list-style-type: none"> <li>1. The domain user logins into the application</li> <li>2. Navigates to the manage reviews area</li> <li>3. Clicks on ‘Delete’ on the choice of review by the domain user</li> </ol>	<ol style="list-style-type: none"> <li>4. Searches for the review with the user id and the review id on the database.</li> <li>5. Deletes the review from the database.</li> </ol>

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 message displayed.

Table 45: Use case description UC:04 (*Self-Composed*)

Use Case Name	Search new hyperparameters
Use Case Id	UC:04
Description	Searching for new set of hyperparameters during model retraining process.
Primary Actor	Domain Specific User
Pre-Conditions	Domain user should have entered enough data into the system
Extended use cases	None
Included use cases	Retrain Model
Trigger	Domain user have triggered the model retraining from the UI by clicking on to the “Retrain model” button
Main flow	<ol style="list-style-type: none"> <li>1. New unseen data is fetched from the database.</li> <li>2. The data is used for automated hyperparameter model training.</li> <li>3. New hyperparameter is used for model training</li> </ol>
Alternative flows	None
Expectational flows	None
Post Conditions	None

Table 46: Use case description UC:05 (*Self-Composed*)

Use Case Name	Create model
Use Case Id	UC:05
Description	Using the new set of hyperparameters found the model is retrained to create a new updated version

Primary Actor	Domain Specific User
Pre-Conditions	Domain user should have entered enough data into the system
Extended use cases	None
Included use cases	Retrain Model
Trigger	Domain user have triggered the model retraining from the UI by clicking on to the “Retrain model” button
Main flow	<ol style="list-style-type: none"> <li>1. Newly found hyperparameters are used to retrain the model.</li> <li>2. Old model is replaced with the new model.</li> </ol>
Alternative flows	None
Expectational flows	None
Post Conditions	None

Table 47: Use case description UC:06 (*Self-Composed*)

Use Case Name	Prepare dataset
Use Case Id	UC:06
Description	Pulling the new data from the database in order to create a new dataset for model retraining
Primary Actor	Domain Specific User
Pre-Conditions	Domain user should have entered enough data into the system
Extended use cases	None
Included use cases	Retrain Model
Trigger	Domain user have triggered the model retraining from the UI by clicking on to the “Retrain model” button
Main flow	<ol style="list-style-type: none"> <li>1. Gets the parameters sent from the request body.</li> <li>2. Fetches data from the database related to the parameters.</li> <li>3. Creating new dataset using the data.</li> </ol>
Alternative flows	None
Expectational flows	None

Post Conditions	None
-----------------	------

Table 48: Use case description UC:08 (*Self-Composed*)

Use Case Name	Generate Summary	
Use Case Id	UC:08	
Description	Generating summary for the input review using the latest model saved.	
Primary Actor	Domain Specific User, General User	
Pre-Conditions	User should have entered a review text from the frontend to generate a summary for.	
Extended use cases	None	
Included use cases	View Summary	
Trigger	User clicked on “Generate summary” after using the review text as input.	
Main flow	Actor	System
	1. User should have entered a text from the frontend in the input field requested. 2. User clicks on “General summary”	3. System uses the input review to perform data preprocessing. 4. System uses the preprocessed text review to generate the summary
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 message displayed.	

Table 49: Use case description UC:09 (*Self-Composed*)

Use Case Name	Store data
Use Case Id	UC:09

Description	Storing the review and summary data along with the sentiment.	
Primary Actor	Domain Specific User	
Pre-Conditions	Domain user should have entered input review and requested	
Extended use cases	None	
Included use cases	View summary	
Trigger	Domain user clicks on ‘Generate summary’ after adding a review text	
Main flow	Actor	System
	1. User should have entered a text from the frontend in the input field requested. 2. User clicks on “General summary”	3. The review data is used to generate the summary. 4. Using the generated summary to get the sentiment and sentiment score. 5. The result of all these will be written into the database
Alternative flows	None	
Expectational flows	None	
Post Conditions	None	

#### C.4. Functional requirements

Table 50: ‘MoSCoW’ technique of requirement prioritization

Priority level	Description
Must have (M)	The demand at this level is the fundamental functional requirement for a prototype, and it must be carried out.
Should have (S)	Although not strictly required for the anticipated prototype to function, important criteria do provide a lot of value.
Could have (C)	Optional, non-essential desirable needs are crucial to the project's scope.
Will not have (W)	Requirements that the system might not meet right now and that are not given first consideration.

## APPENDIX D – DESIGN

---



---

### D.1. Design goals

Table 51: 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 be 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.

## D.2. UI wireframes

Figure 31: Homepage Wireframe (Self-Composed)

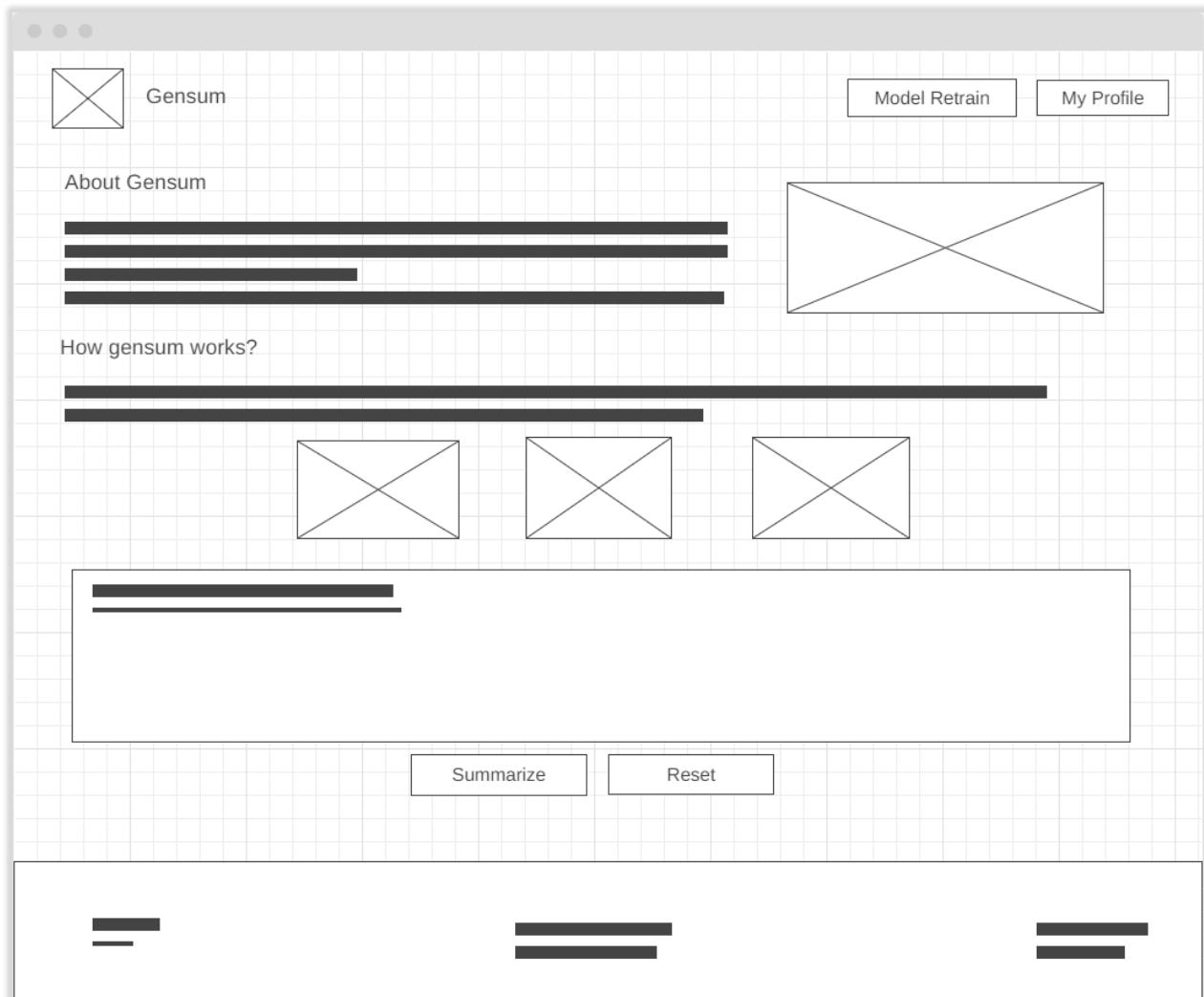
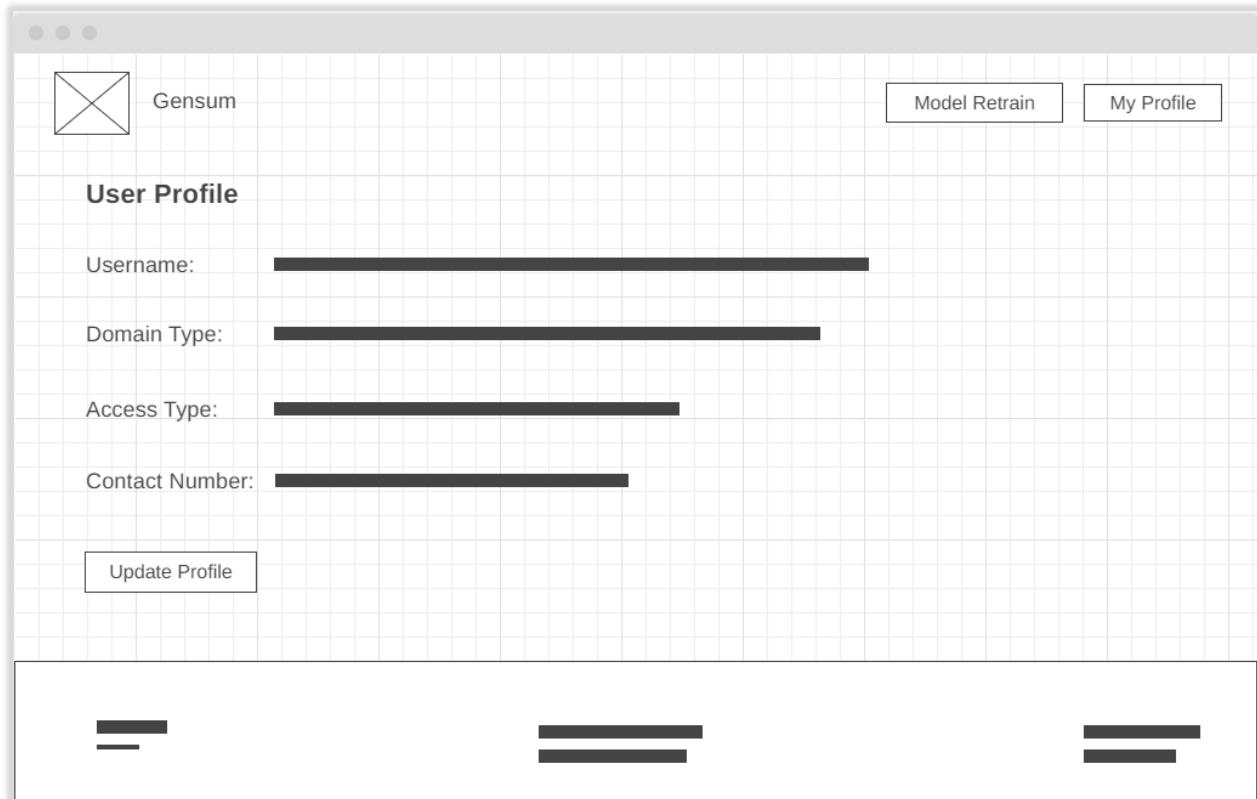


Figure 32: Review History Page (Self-Composed)



Figure 33: User Profile Page (Self-Composed)



## APPENDIX E – IMPLEMENTATION

### E.1. Dataset Resources

- **CNN Dailymail**

Figure 34: CNN Dailymail Dataset (*view*)

The screenshot shows a web page from the TensorFlow Datasets Catalog. At the top, there is a navigation bar with links to 'TensorFlow', 'Resources', 'Datasets', and 'Catalog'. On the right side of the header, there are two small icons: a thumbs-up for 'Was this helpful?' and a thumbs-down. Below the header, the title 'cnn\_dailymail' is displayed next to a dropdown menu icon. A bulleted list titled 'Description:' follows, stating 'CNN/DailyMail non-anonymized summarization dataset.' It notes that there are two features: 'article' (text of news article) and 'highlights' (joined text of highlights with and around each highlight, which is the target summary). Further down, another bulleted list provides details: 'Additional Documentation' (link to 'Explore on Papers With Code'), 'Homepage' (link to <https://github.com/abisee/cnn-dailymail>), 'Source code' (link to `tfds.summarization.CnnDailymail`), 'Versions' (listing versions 1.0.0 through 3.4.0 with their respective changes), 'Download size' (558.32 MiB), and 'Dataset size' (1.29 GiB).

- **Description:**

CNN/DailyMail non-anonymized summarization dataset.  
There are two features: - article: text of news article, used as the document to be summarized - highlights: joined text of highlights with and around each highlight, which is the target summary
- **Additional Documentation:** [Explore on Papers With Code](#)
- **Homepage:** <https://github.com/abisee/cnn-dailymail>
- **Source code:** `tfds.summarization.CnnDailymail`
- **Versions:**
  - 1.0.0 : New split API (<https://tensorflow.org/datasets/splits>)
  - 2.0.0 : Separate target sentences with newline. (Having the model predict newline separators makes it easier to evaluate using summary-level ROUGE.)
  - 3.0.0 : Using cased version.
  - 3.1.0 : Removed BuilderConfig
  - 3.2.0 : Remove extra space before added sentence period. This shouldn't affect ROUGE scores because punctuation is removed.
  - 3.3.0 : Add publisher feature.
  - 3.4.0 (default): Add ID feature.
- **Download size:** 558.32 MiB
- **Dataset size:** 1.29 GiB

- **Gigaword**

Figure 35: Gigaword Dataset (view)

The screenshot shows a web page from the TensorFlow Datasets Catalog. The URL is [https://tfhub.dev/datasets/gigaword/1](#). The page title is "gigaword". The main content includes a "Description" section with a detailed description of the dataset, its features, and download information.

**Description:**

Headline-generation on a corpus of article pairs from Gigaword consisting of around 4 million articles. Use the 'org\_data' provided by <https://github.com/microsoft/unilm/> which is identical to <https://github.com/harvardnlp/sent-summary> but with better format.

There are two features: - document: article. - summary: headline.

- **Homepage:** <https://github.com/harvardnlp/sent-summary>
- **Source code:** [tfds.summarization.Gigaword](#)
- **Versions:**
  - [1.2.0](#) (default): No release notes.
- **Download size:** 551.61 MiB
- **Dataset size:** 1.02 GiB

- **Xsum**

Figure 36: Xsum Dataset (view)

The screenshot shows the TensorFlow Datasets Catalog page for the Xsum dataset. The URL is [TensorFlow > Resources > Datasets > Catalog](#). At the top right, there are "Was this helpful?" buttons for thumbs up and thumbs down. Below the title "XSUM" is a red warning box containing the text: "⚠ Warning: Manual download required. See instructions below." The main content area starts with a bullet point: "• Description:" followed by the text: "Extreme Summarization (XSum) Dataset. There are two features: - document: Input news article. - summary: One sentence summary of the article. This data need to manually downloaded and extracted as described in <https://github.com/EdinburghNLP/XSum/blob/master/XSum-Dataset/README.md> The folder 'xsum-extracts-from-downloads' need to be compressed as 'xsum-extracts-from-downloads.tar.gz' and put in manually downloaded folder." Below this, there is a list of additional documentation links:

- Additional Documentation: [Explore on Papers With Code](#)
- Homepage: <https://github.com/EdinburghNLP/XSum/tree/master/XSum-Dataset>
- Source code: [tfds.summarization.Xsum](#)
- Versions:
  - 1.0.0 : Dataset without cleaning.
  - 1.1.0 (default): Removes web contents.
- Download size: 2.59 MiB
- Dataset size: 512.03 MiB

- **Amazon movie reviews**

Figure 37: Amazon Movie Reviews Dataset (view)

By Jure Leskovec STANFORD UNIVERSITY



**Web data: Amazon movie reviews**

**Dataset information**

This dataset consists of movie reviews from [amazon](#). The data span a period of more than 10 years, including all ~8 million reviews up to October 2012. Reviews include product and user information, ratings, and a plaintext review. We also have reviews from [all other Amazon categories](#).

Dataset statistics	
Number of reviews	7,911,684
Number of users	889,176
Number of products	253,059
Users with > 50 reviews	16,341
Median no. of words per review	101
Timespan	Aug 1997 - Oct 2012

**Source (citation)**

- J. McAuley and J. Leskovec. [From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews](#). WWW, 2013.

**Files**

File	Description
<a href="#">movies.txt.gz</a>	Amazon movie data (~8 million reviews)

**Data format**

```

product/productId: B00006HAXW
review/userId: A1RSDE90N6RSZF
review/profileName: Joseph M. Kotow
review/helpfulness: 9/9
review/score: 5.0
review/time: 1042502400
review/summary: Pittsburgh - Home of the OLDIES
review/text: I have all of the doo wop DVD's and this one is as good or better than the
1st ones. Remember once these performers are gone, we'll never get to see them again.
Rhino did an excellent job and if you like or love doo wop and Rock n Roll you'll LOVE
this DVD !!

```

- **Hotel Reviews**

Figure 38: Hotel Reviews Dataset (view)

The screenshot shows a Kaggle notebook interface. The title is "Sentiment analysis with hotel reviews". The notebook is in Python and has 515K Hotel Reviews Data in Europe. It is a Notebook with 369.8s run time and version 2 of 2. The categories are Data Visualization, Exploratory Data Analysis, Classification, NLP, and Feature Engineering. The introduction section discusses sentiment analysis as part of Natural Language Processing (NLP) and its application in extracting emotions from customer reviews. It mentions libraries like NLTK, Gensim, and Scikit-learn. The notebook then loads data from Kaggle, samples it, and performs clean data operations. It includes sections on feature engineering, exploratory data analysis, and modeling reviewer scores. The conclusion section summarizes the findings.

## E.2. Selection of Programming Language

The following table provides a summary of the evaluation of the programming language selected for the data science segment, in which each alternative was assigned a score ranging from H (High), M (Medium), to L (Low).

Table 52: Selection of Data Science Language

<b>Data science</b>			
<b>Aspect</b>	<b>Relevance</b>	<b>Python</b>	<b>R</b>
Library availability and accessibility.	Essential to have a language that supports multiple libraries for the author to choose from, to gather data and build the model.	H	M

Author experience and implementation ease.	Efforts should be made to simplify the model application process, and it would be helpful if the author has experience with the chosen language.	H	M
Learning curve	Progress should not be hindered by the language's complexity since the goal is to utilize it as a tool to build a system, not to spend time learning it unnecessarily (Virtanen et al., 1970).	L	M
Documentation and Community	Clear documentation and a supportive community are vital since the author cannot afford to fix minor issues themselves.	H	M
<b>Conclusion</b>			
After analyzing the options, the author opted for <b>Python</b> since it was more suitable.			

### E.3. Selection of Deep Learning (DL) framework

Table 53: Selection of DL framework

Framework	Description
TensorFlow	This tool is geared towards production-level applications and can handle large datasets. It is equipped with comprehensive documentation and has a supportive community. Additionally, it offers improved visualization options that simplify the process of debugging and monitoring training (Abadi et al., 1970).
PyTorch	This tool has a higher-level development, making it more lightweight and user-friendly. It has a smaller learning curve, making it easier to start using, and it feels more intuitive since building models is simpler (Paszke et al., 1970).
<b>Conclusion</b>	
The author opted to use <b>PyTorch</b> due to lightweight and user-friendly feature which makes it easier to work with (Paszke et al., 1970).	

## E.4. Selection of User Interface (UI) framework

Table 54: Selection of UI framework

Framework	Description
Angular	This tool is appropriate for large-scale applications and includes dedicated submodules for specific functionalities. However, it may be less performant than other options and can be unnecessarily heavy (Waranashiwar & Ukey, 1970).
Vue	This framework is small and starts up quickly, and its code is straightforward, making it easy to use. Simulations have shown that it outperforms Angular and React. However, it has significantly fewer resources available (Wahyudi et al., 1970).
Svelte	This option is the most lightweight and reactive, offering superior performance compared to others. However, it has a small community of developers and is relatively new.
React	This option allows for customization and promotes code reusability through functions as components. It has a large and active community and is open-source, as well as being SEO-friendly. Furthermore, the React developer tools are very useful (Verma et al., 1970).
<b>Conclusion</b>	After analyzing the options, the author selected <b>React</b> for building the GUI since it will be simple, and there is no need for a tool capable of handling large-scale applications, which is not the main focus (Verma et al., 1970).

## E.5. Selection of Application Programming Interface (API) framework

Table 55: Selection of web framework

Framework	Description
Flask	This framework is extremely lightweight and offers only basic functionality. Nonetheless, it is the preferred option for ML API development due to its lightness (Relan, 1970).

Django	This option is appropriate for larger-scale applications that require a wide range of functionalities. However, it is more rigid and less flexible, making it more demanding and heavier (Srivastava, 1970).
<b>Conclusion</b>	
The author opted for <b>Flask</b> since it provides only the essential features required for exposing an ML model.	

## E.6. Data Preprocessing

Figure 39: Preprocessing: Remove Markdown (Self-Composed)

```
def md_links(text: Text) -> Text:
    markdown_link=re.compile(r'\[.*?\]\(..*?\)')
    return markdown_link.sub(r'',text)

df['text'] = df['text'].parallel_apply(lambda sentence: md_links(sentence))
df['summary'] = df['summary'].parallel_apply(lambda sentence: md_links(sentence))
```

Figure 40: Preprocessing – Remove Hyperlinks (Self-Composed)

```
def scrape_links(text):
    url = re.compile(r'https?://\S+|www\.\S+')
    return url.sub(r'',text)

df['text'] = df['text'].parallel_apply(lambda sentence: scrape_links(sentence))
df['summary'] = df['summary'].parallel_apply(lambda sentence: scrape_links(sentence))
```

Figure 41: Preprocessing: Remove Html Tags (Self-Composed)

```
def remove_html_tags(text: Text) -> Text:  
    html=re.compile(r'<.*?>')  
    return html.sub(r'',text)
```

```
df['text'] = df['text'].parallel_apply(lambda sentence: remove_html_tags(sentence))  
df['summary'] = df['summary'].parallel_apply(lambda sentence: remove_html_tags(sentence))
```

Figure 42: Preprocessing: Char Words Extension (Self-Composed)

```
def chat_words_conversion(text: Text) -> Text:  
    new_text = []  
    for word in text.split():  
        if word.upper() in chat_words_map_dict:  
            new_text.append(chat_words_map_dict[word.upper()])  
        else:  
            new_text.append(word)  
    return " ".join(new_text)
```

```
df['text'] = df['text'].parallel_apply(lambda sentence: chat_words_conversion(sentence))  
df['summary'] = df['summary'].parallel_apply(lambda sentence: chat_words_conversion(sentence))
```

The above code snippets are used to convert the short key words into longer form, such as e.g.: ‘ATM’ is converted into ‘At the moment’

Figure 43: Preprocessing: Handling Common Contractions (Self-Composed)

```
def en_contractions(text: Text) -> Text:  
    return ' '.join([contractions.fix(word)  
                    if word in contractions.contractions_dict else word  
                    for word in text.split()])  
  
df['text'] = df['text'].parallel_apply(lambda sentence: en_contractions(sentence))  
df['summary'] = df['summary'].parallel_apply(lambda sentence: en_contractions(sentence))
```

The above code snippets are used to handle/extend common contractions such as e.g.: ‘They’re’ into ‘They are’

Figure 44: Preprocessing: Removing Special Characters (Self-Composed)

s\_chars = 'ÝÞÍÜÜÐÝæÞÛÑAéÃ"â'Gâ'ZöeríiyýësÞáraÛðAözíçSñueÓgðç'íTíAðeÛðEáÍtíCðâhÝç"zðÙëcùëæðEðjúñAñ,âAûñiðeðþiðððÛðÛñU' PUNC = '+@#\$\_-!\$%^&\*&(`E<>?/\|\})\[\{\,;\,\:,\'\''

```
def special_char(text: Text) -> Text:
    # first, let's remove any unicode strings
    text = text.encode('ascii', 'ignore').decode()
    # remove printable backslashes
    text = re.sub(r'[\t\s\\n\\r\\b\\a]', ' ', text)
    # Special letters
    text = re.sub(r'[\{\}]'.format(s_chars), '', text)
    # Punctuation [remove punctuation between spaces only which represent noises]
    text = re.sub(r'\s[\{\}]\s'.format(PUNC), ' ', text)
    # space at the start or the end of the context
    text = re.sub(r'(^\\s)|(\\s$)', ' ', text)
    # Single character
    text = re.sub(r'(^\\s[^iIaA]\\s)', ' ', text)
return text
```

```
df['text'] = df['text'].parallel_apply(lambda sentence: special_char(sentence))
df['summary'] = df['summary'].parallel_apply(lambda sentence: special_char(sentence))
```

```
df.head(3)
```

Figure 45: Preprocessing: Resolving Spelling Mistakes (Self-Composed)

```
from textblob import TextBlob

def spell_correction(df):
    # creating a new column for the corrected text
    df['corrected_text'] = df['text']
    # creating a new column for the corrected summary
    df['corrected_summary'] = df['summary']
    # creating a for loop for the entire dataset
    for i in range(len(df)):
        # Records
        print('Counter: ' + str(i+1) + '/' + str(len(df)+1))
        # creating a variable for the text of the current row
        text = df['corrected_text'][i]
        # creating a variable for the summary of the current row
        summary = df['corrected_summary'][i]
        # creating a variable for the corrected text of the current row
        corrected_text = TextBlob(text).correct()
        # creating a variable for the corrected summary of the current row
        corrected_summary = TextBlob(summary).correct()
        # updating the corrected text column with the corrected text
        df['corrected_text'][i] = str(corrected_text)
        # updating the corrected summary column with the corrected summary
        df['corrected_summary'][i] = str(corrected_summary)
    # returning the dataset with the new columns
    return df

spell_correction(df)
df_copy_correction = df.copy()
```

Figure 46: Preprocessing: Removing Duplicates (Self-Composed)

```

def rm_duplicates(text: Text) -> Text:
    return re.sub(r'\b(\w+\s*)\b{1}', '\\1', text)

df_copy_correction['corrected_text'] = df_copy_correction['corrected_text'].parallel_apply(lambda sentence: rm_duplicates(sentence))
df_copy_correction['corrected_summary'] = df_copy_correction['corrected_summary'].parallel_apply(lambda sentence: rm_duplicates(sentence))

```

Figure 47: Preprocessing: Restoring Missing Punctuations (Self-Composed)

```

args = InferenceArguments(
    model_name_or_path="Qishuai/distilbert_punctuator_en",
    tokenizer_name="Qishuai/distilbert_punctuator_en",
    tag2punctuator=DEFAULT_ENGLISH_TAG_PUNCTUATOR_MAP,
)
inference = Inference(inference_args=args, verbose=False)

def punct_restoration(list_of_text: List[Text], name: Text) -> List[Text]:
    list_of_texts = []
    for text in tqdm(list_of_text, desc=f"Auto Punctuation for {name}"):
        list_of_texts.append(
            inference.punctuation([text])[0][0]
        )
    return list_of_texts

df_copy_correction['punc_corrected_text'] = punct_restoration(df_copy_correction['corrected_text'].values.tolist(), "text")
df_copy_correction['punc_corrected_summary'] = punct_restoration(df_copy_correction['corrected_summary'].values.tolist(), 'summary')

```

Figure 48: Preprocessing: Grammarly Correction (Self-Composed)

```

def grammely_correction(list_text: List[Text], name: Text) -> List[Text]:
    list_of_correction = []
    for text in tqdm(list_text, desc=f'Grammely Correction for {name}'):
        if len(text.split()) < 50:
            list_of_correction.append(list(gf.correct(text, max_candidates=1))[0])
        else:
            list_of_correction.append(" ".join([list(gf.correct(sentence, max_candidates=1))[0]
                                                for sentence in tokenizer.tokenize(text)]))
    return list_of_correction

df_copy_correction['gram_corrected_text'] = grammely_correction(df_copy_correction['punc_corrected_text'].values.tolist(), "text")
df_copy_correction['gram_corrected_summary'] = grammely_correction(df_copy_correction['punc_corrected_summary'].values.tolist(), 'summary')

```

## E.7. Generalized model training

The following code snippets will be related to the generalized model training for the bart-base model since, it is the finalized model after the experimentation.

Figure 49: Mounting to google drive & handling imports

```
# Connecting to Gogole Colab
from google.colab import drive
drive.mount('/content/gdrive')
```

Mounted at /content/gdrive

```
# Installing the required libraries
!pip install sentencepiece optuna
!pip install torch huggingface_hub
!pip install datasets transformers
!pip install rouge_score nltk py7zr
```

Figure 50: Handling library imports for generalized model

```
# Importing the necessary libraries
import torch
import numpy as np
import pandas as pd
import datasets
import optuna
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM, DataCollatorForSeq2Seq
import tensorflow as tf
from datasets import load_metric
import nltk
from huggingface_hub import notebook_login
from transformers.keras_callbacks import KerasMetricCallback

import transformers
from datasets import load_dataset, load_metric, load_from_disk
import numpy as np
import pandas as pd
import nltk

nltk.download('punkt')
```

Figure 51: Variable initialization

```

# Create function for printing
def print_custom(text):
    print('\n')
    print(text)
    print('-'*100)

# Specify our parameter range and project variables
LR_MIN = 4e-5
LR_CEIL = 0.01
WD_MIN = 4e-5
WD_CEIL = 0.01
MIN_EPOCHS = 8
MAX_EPOCHS = 15
PER_DEVICE_EVAL_BATCH = 4
PER_DEVICE_TRAIN_BATCH = 4
MIN_BATCH_SIZE = 4
MAX_BATCH_SIZE = 6
NUM_TRIALS = 1
SAVE_DIR = 'opt-test'
MODEL_NAME = 'facebook/bart-base'
MAX_INPUT = 512
MAX_TARGET = 128

# Selecting 1000 rows from the dataset
dataset_path = 'gdrive/My Drive/fyp/xsum/'
data = pd.read_csv(dataset_path + 'xsum.csv', encoding='latin-1')
data = data[0:1000]

metric = load_metric('rouge')
data

```

Figure 52: Handle preprocess data and train test split

```

prefix = "summarize: "
def preprocess_data(data_to_process):
    #get the document text
    if 'ts' in MODEL_NAME:
        inputs = [prefix + doc for doc in data_to_process["document"]]
    else:
        inputs = [document for document in data_to_process['document']]

    #tokenize text
    model_inputs = tokenizer(inputs, max_length=MAX_INPUT, padding='max_length', truncation=True)

    #tokenize labels
    with tokenizer.as_target_tokenizer():
        targets = tokenizer(data_to_process['summary'], max_length=MAX_TARGET, padding='max_length', truncation=True)

    model_inputs['labels'] = targets['input_ids']
    #returns input_ids, attention_masks, labels
    return model_inputs

# Perform a train test split of 80:20 ratio on the dataset
train_dataset = data[:int(len(data)*0.7)]
test_dataset = data[int(len(data)*0.7):int(len(data)*0.85)]
validation_dataset = data[int(len(data)*0.85):]

data = datasets.DatasetDict({ 'train': datasets.Dataset.from_pandas(train_dataset),
                             'test': datasets.Dataset.from_pandas(test_dataset),
                             'validation': datasets.Dataset.from_pandas(validation_dataset)})

tokenize_data = data.map(preprocess_data, batched = True, remove_columns=['document', 'summary'])

```

Figure 53: Creating rouge function for evaluation

```
#####
# metrics
# compute rouge for evaluation
#####

def compute_rouge(pred):
    predictions, labels = pred
    #decode the predictions
    decode_predictions = tokenizer.batch_decode(predictions, skip_special_tokens=True)
    #decode labels
    decode_labels = tokenizer.batch_decode(labels, skip_special_tokens=True)

    #compute results
    res = metric.compute(predictions=decode_predictions, references=decode_labels, use_stemmer=True)
    #get %
    res = {key: value.mid.fmeasure * 100 for key, value in res.items()}

    pred_lens = [np.count_nonzero(pred != tokenizer.pad_token_id) for pred in predictions]
    res['gen_len'] = np.mean(pred_lens)

    return {k: round(v, 4) for k, v in res.items()}
```

Figure 54: Model parameter &amp; training parameter tuning

```
from transformers import BartConfig, BartForConditionalGeneration

print_custom('Performing search space for hyperparameters....')

def objective(trial: optuna.Trial):
    config = BartConfig.from_pretrained('facebook/bart-base')

    config.decoder_attention_heads = trial.suggest_categorical('decoder_attention_heads', [2, 3, 4, 6, 8, 12, 16])
    config.decoder_ffn_dim = trial.suggest_int('decoder_ffn_dim', 1024, 4096)
    config.decoder_layerdrop = trial.suggest_uniform('decoder_layerdrop', 0.0, 0.3)
    config.decoder_layers = trial.suggest_int('decoder_layers', 4, 12)
    config.decoder_start_token_id = 1

    if config.d_model % config.decoder_attention_heads != 0:
        config.decoder_attention_heads = config.d_model // 64

    model = BartForConditionalGeneration(config=config)

    data_collator = transformers.DataCollatorForSeq2Seq(tokenizer, model=model)

    training_args = Seq2SeqTrainingArguments(
        output_dir=SAVE_DIR,
        save_strategy="epoch",
        evaluation_strategy="epoch",
        learning_rate=trial.suggest_float("learning_rate", LR_MIN, LR_CEIL, log=True),
        weight_decay=trial.suggest_float("weight_decay", WD_MIN, WD_CEIL, log=True),
        num_train_epochs=trial.suggest_int("num_train_epochs", MIN_EPOCHS, MAX_EPOCHS),
        warmup_ratio=trial.suggest_float("warmup_ratio", 0.0, 1.0),
        per_device_train_batch_size=trial.suggest_int("per_device_train_batch_size", MIN_BATCH_SIZE, MAX_BATCH_SIZE),
        per_device_eval_batch_size=trial.suggest_int("per_device_eval_batch_size", MIN_BATCH_SIZE, MAX_BATCH_SIZE),
        save_total_limit=1,
        load_best_model_at_end=True,
        greater_is_better=True,
        predict_with_generate=True,
        run_name=MODEL_NAME,
        report_to="none",
    )
```

## E.8. User interface

Figure 55: GUI – Homepage (After Signup) (Self-Composed)

**About Gensum**

Gensum is a tool for abstractive text summarization of English review texts, utilizing advanced NLP techniques and optimized deep learning algorithms (Transformers) built with Python, Pytorch, Huggingface Transformers library, React, and Typescript. Its backend is created using Flask while its frontend is built with React.

Initially, the model is designed to be adaptable to any domain and will improve its performance as it is used. Users can also retrain the model with their own data and automated hyperparameter tuning will be conducted during the retraining process. This enables the model to adapt to new domains and improve its performance.

In addition to the main function, the tool also displays the sentiment of the summarized review, including the sentiment score. For domain users, they can view and delete the review text they input, allowing them to decide which data to use when retraining the model. This helps prevent retraining the model with faulty data, which could result in a loss of performance.

Domain users have the additional ability to generate a CSV file of the results fetched from the database, as well as manage their profile metadata. They will receive push notifications to inform them when the model retraining completes, as well as updates throughout the retraining progress.

**How gensum works?**

User has the option to sign-in to the application as a domain user, or to also use the application as a general user without signing in. Domain users are allowed to retrain the model with their own data, and also manage their profile metadata.

General users are not allowed to retrain the model with their own data, and also not allowed to manage their profile metadata. However, both of the users will be able to perform the core functionality of the application which is to summarize the review text.

Review summary: 4.0

Review summary: 5.0

Review summary: 4.6

Enter review here...

Summarize Reset

@nozhimkalam Terms of Service | Privacy Policy Facebook | Twitter

Figure 56: GUI – Review History Page (Self-Composed)

The screenshot shows the Gensum application interface. At the top, there is a navigation bar with the Gensum logo, a "Model Retrain" button, and a "My Profile" dropdown. Below the navigation bar, there is a "REVIWS" section with a "Download Records" button. The main content area displays two reviews in separate cards:

- REVIEW 1**
  - Review:** Gensum is a tool for abstractive text summarization of English review texts, utilizing advanced NLP techniques and optimized deep learning algorithms (Transformers) built with Python, Pytorch, Huggingface Transformers library, React, and Typescript. Its backend is created using Flask while its frontend is built with React.
  - Summary:** Gensum is a tool for deep learning, designed to speed up translation and improve quality of data produced by machine learning. It has been developed by Pytorch and Londonderry, and is based on Deep Learning.
  - Sentiment:** Positive
  - Sentiment score:** 0.9884
  - Created At:** Mon, 24 Apr 2023 04:30:55 GMT
  - DELETE** button
- REVIEW 2**
  - Review:** I am a creative Full-Stack Web Developer who has experience in technologies such as Data Science & ML and Cloud Computing. I am a highly coordinated, committed and diplomatic software engineer with a defined capacity to operate and execute any specific role on schedule. I am able to communicate with a vast variety of individuals easily, with outstanding organizational skills. I see that I will bring my skills and expertise into practice in a full-time role in the industry, which will directly support the activities of the businesses I am involved in. I have the potential to build original conceptions and insights and solve a great many problems, guided by my intuitive and optimistic approach to problem solving. In algorithms as in business scenarios, I am able to apply my problems solving skills. Furthermore, I can easily and effectively understand the intensifying principles and help others to develop with great self encouragement. Therefore, I guess I am able to handle a lot of teams.
  - Summary:** I am a full-time software engineer, who has extensive experience in the development and implementation of software, including for projects that require large amounts of hardware and software. I am a highly coordinated, committed and diplomatic software engineer with a well-developed understanding of problem solving. I have demonstrated the ability to develop and execute complex software projects on schedule and on schedule.
  - Sentiment:** Positive
  - Sentiment score:** 0.9997
  - Created At:** Thu, 30 Mar 2023 15:25:01 GMT
  - DELETE** button

At the bottom of the page, there is a footer bar with links: @nazhimkalam, Terms of Service | Privacy Policy, and Facebook | Twitter.

Figure 57: GUI – User Profile (Self-Composed)

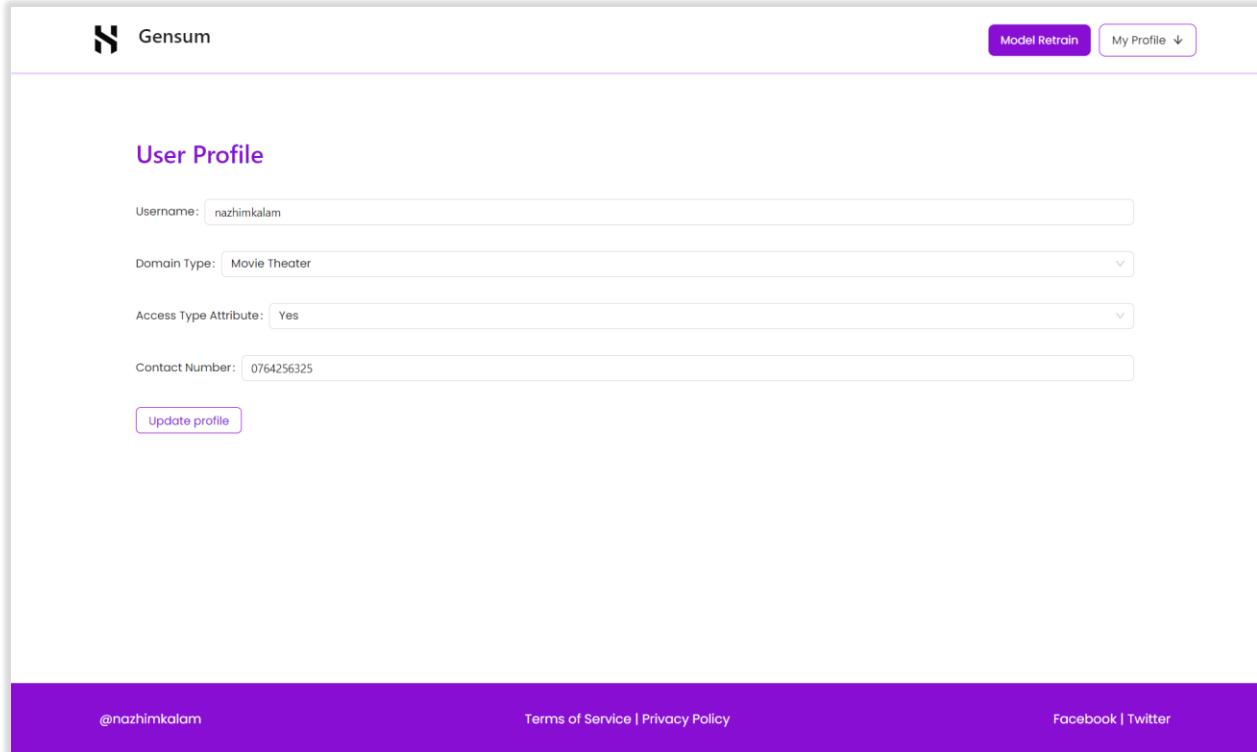


Figure 58: GUI – Model Retraining Modal (Self-Composed)

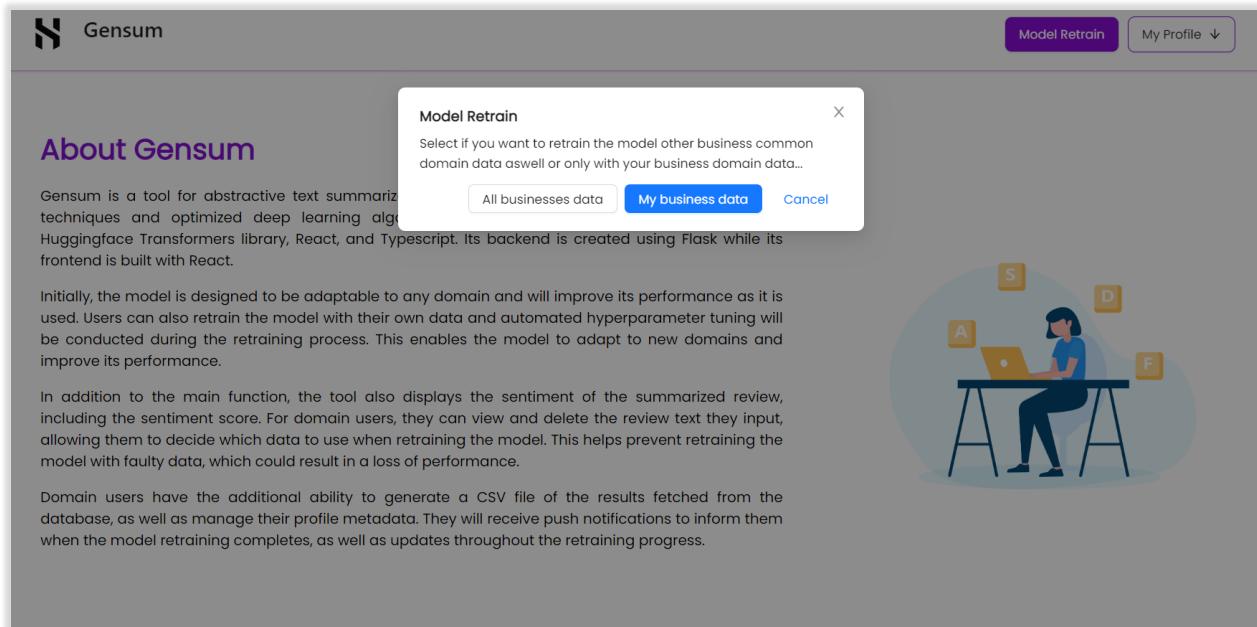


Figure 59: GUI – Homepage (Before Signup) (Self-Composed)

The screenshot shows the Gensum homepage. At the top left is the Gensum logo (a stylized 'G') and the word 'Gensum'. At the top right is a purple 'Sign In' button. Below the header, there's a section titled 'About Gensum' with a purple background. The text describes Gensum as a tool for abstractive text summarization of English review texts, using advanced NLP techniques and deep learning algorithms (Transformers) built with Python, Pytorch, Huggingface Transformers library, React, and Typescript. Its backend is created using Flask while its frontend is built with React. It mentions that the model is adaptable to any domain, can be retrained with user data, and includes automated hyperparameter tuning. The text also notes that domain users can generate CSV files and manage profile metadata, receiving push notifications for retraining updates. To the right of the text is a circular illustration of a person sitting at a desk with a laptop, surrounded by four orange boxes labeled 'A', 'B', 'C', and 'D'.

Gensum is a tool for abstractive text summarization of English review texts, utilizing advanced NLP techniques and optimized deep learning algorithms (Transformers) built with Python, Pytorch, Huggingface Transformers library, React, and Typescript. Its backend is created using Flask while its frontend is built with React.

Initially, the model is designed to be adaptable to any domain and will improve its performance as it is used. Users can also retrain the model with their own data and automated hyperparameter tuning will be conducted during the retraining process. This enables the model to adapt to new domains and improve its performance.

In addition to the main function, the tool also displays the sentiment of the summarized review, including the sentiment score. For domain users, they can view and delete the review text they input, allowing them to decide which data to use when retraining the model. This helps prevent retraining the model with faulty data, which could result in a loss of performance.

Domain users have the additional ability to generate a CSV file of the results fetched from the database, as well as manage their profile metadata. They will receive push notifications to inform them when the model retraining completes, as well as updates throughout the retraining progress.

## APPENDIX F – TESTING

---



---

### F.1. Functional Testing

Table 56: Functional Testing

Test case	FR ID	User Action	Expected Result	Actual Result	Result Status
1	FR1	Users, both general and specific to the domain, can input review text to produce a summary.	Users can input review text in a provided field and generate a summary by clicking a button.	Users can input review text in a provided field and generate a summary by clicking a button.	Passed
2	FR10, FR11	Users will click "signup" to create an account, then go to their profile to update it.	After signing up, domain users will see their personal profile and can access their pages.	After signing up, domain users will see their personal profile and can access their pages.	Passed
3	FR2	If there is enough data in the database, domain users can click the "retrain model" button to initiate model retraining.	Users will receive a prompt to confirm model retraining, and email notifications will be sent to them to update them on the progress.	Users will receive a prompt to confirm model retraining, and email notifications will be sent to them to update them on the progress.	Passed
4	FR12	User will be able to trigger model retraining during off peak hours.	Model retraining happens	Model retraining happens	Passed

5	FR3, FR4, FR5, FR15, FR17	None	The system will find the new best set of (model architecture + model training) hyperparameters by fetching the domain user data/groups and perform necessary preprocessing and trigger model retraining.	The system will find the new best set of (model architecture + model training) hyperparameters by fetching the domain user data/groups and perform necessary preprocessing and trigger model retraining.	Passed
6	FR6	None	The summary output will be displayed on the UI	The summary output will be displayed on the UI	Passed
7	FR7	None	The system uses the latest retrained model for summary generation	The system uses the latest retrained model for summary generation	Passed
8	FR13	Users can input review text to generate a summary and receive the sentiment and sentiment score of the review.	Displays the summary text along with the sentiment score and the sentiment.	Displays the summary text along with the sentiment score and the sentiment.	Passed
9	FR8	None	The system encrypts the data and stores it in	The system encrypts the data and stores it in	Passed

			the database for model retraining purposes	the database for model retraining purposes	
10	FR16	Domain user clicks on “delete” review from the review list	The review gets deleted from the database and is updated on the GUI	The review gets deleted from the database and is updated on the GUI	Passed
11	FR9	When domain user submits a review for summarization.	The review and result is encrypted and stored into the database	The review and result is encrypted and stored into the database	Passed

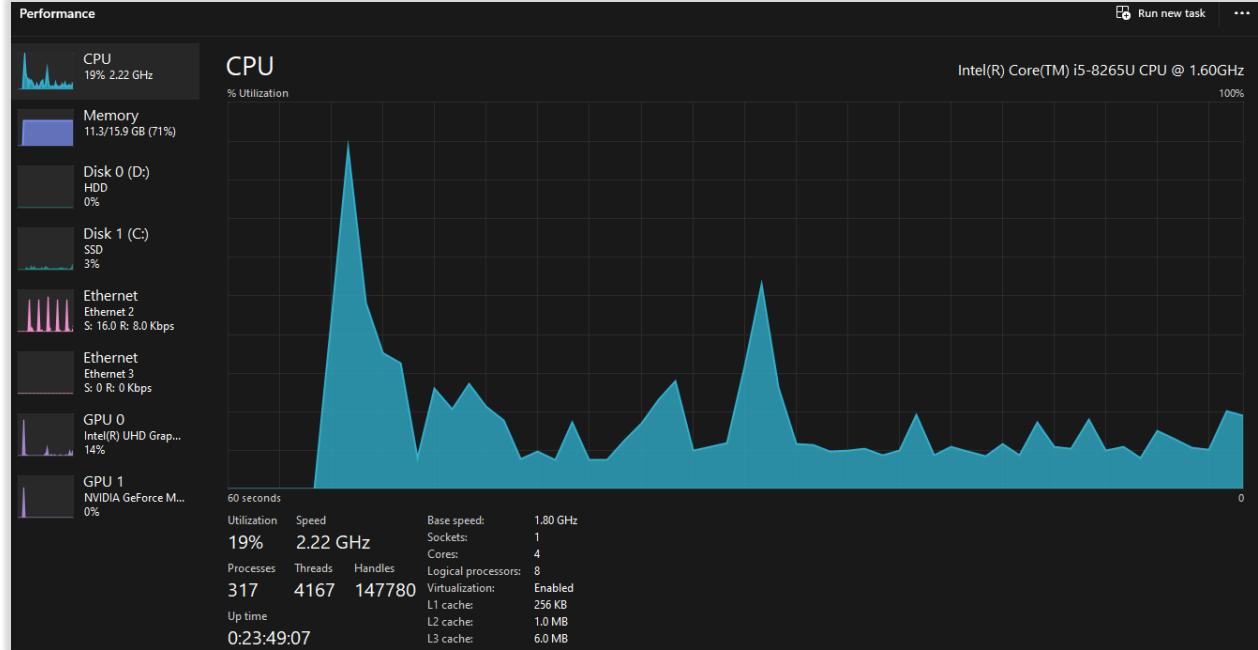
## F.2. Non-functional testing

In order to evaluate whether the system satisfies the non-functional requirements and design objectives, the writer utilized testing methodologies such as performance testing, GUI testing, maintainability testing, and a limited number of test cases.

### Performance testing

At present, the author has configured the system to operate in a local environment, and the system resource manager graph below depicts the utilization of resources when the application is deployed and operational in this environment.

Figure 60: System Resource Manager Graph (Self-Composed)



## GUI testing

During the requirement gathering phase, it was established that developing a straightforward and efficient GUI was crucial. To assess its performance and accessibility, the GUI was evaluated using Google Lighthouse. The results of this evaluation are illustrated in the diagram below.



Figure 61: Lighthouse Landing Page



Figure 62: Lighthouse Records Page



Figure 63: Lighthouse User Profile Page

## Maintainability testing

Ensuring maintainability is essential to enable smooth future research on the system, particularly the developed algorithm. To achieve this, **CodeFactor** and **CodeQL** were employed to verify that the repositories are well-documented and maintained, and that there are no vulnerabilities present.

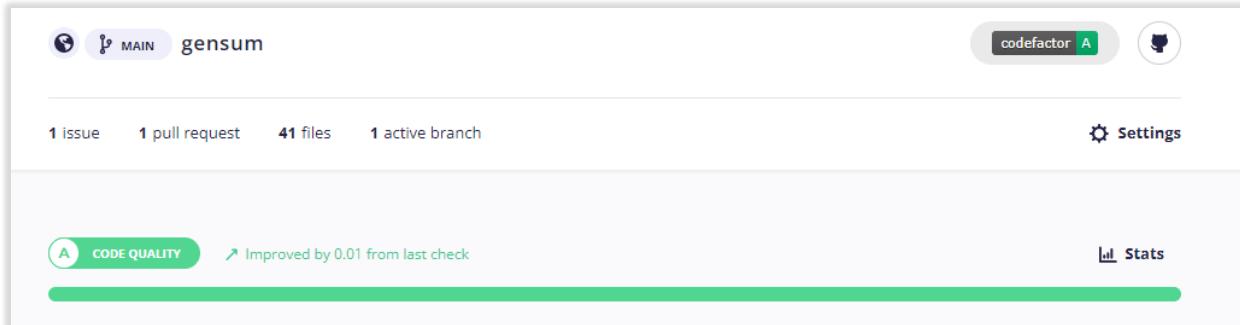


Figure 64: CodeFactor - Gensum Repository

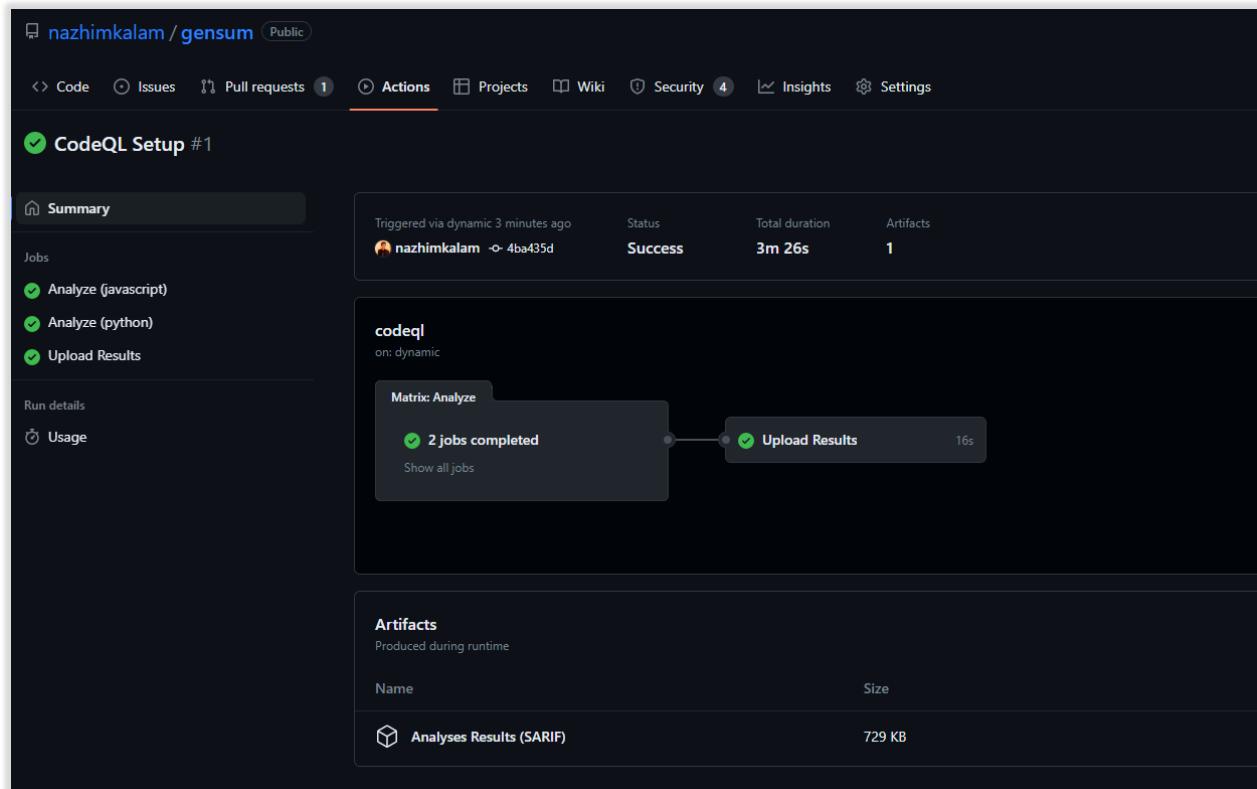


Figure 65: CodeQL - Gensum Repository

## Test cases

Table 57: Non-functional testing

Non-functional requirements		
Test case	ID	Result
1	NFR1	Created a minimalist UI design making it user friendly to all type of users.
2	NFR2	Meaningful error messages are displayed on the GUI, if anything faulty happens.
3	NFR3	The summary generation on the GUI takes an average of 5000ms
4	NFR4	CodeQL and CodeFactor is used to maintain the coding standards up to the best quality.
5	NFR5	The idea of model generalization with respect on the domain users works as expected. A domain specific model is created whenever a new domain user signup into the application.
6	NFR6	Data encryption is applied for the data which are domain specific and stored in the database, to ensure that the meaning of the data is not retrained even if the data is lost.

## APPENDIX G – EVALUATION

---

### G.1. Expert evaluators

Table 58: Details of the expert evaluator(s) selected.

ID	Affiliation	Expertise related to the research
EV1	PhD Research Student NLP	NLP
EV2	NLP Researcher	NLP
EV3	Software Architect & ML Researchers	ML & Algorithms
EV4	Software Engineer	ML & Neural Networks
EV5	Lecturer	NLP

### G.2. Evaluation of functional requirements

Table 59: Evaluation of the implementation of functional requirements

ID	FRID	Description	Priority	Use Case	Evaluation
EV1	FR1	Both general and domain-specific users can input review text from the GUI for summary generation	M	UC:01	Implemented
EV2	FR2	Retrain system with new data for specific domains after obtaining user consent via GUI.	M	UC:03	Implemented
EV3	FR3	The system must be able to find the new set of best hyperparameters with the usage of the new data.	M	UC:04	Implemented
EV4	FR4	The system must be able to retrain the model with the new best hyperparameters and create the model	M	UC:05	Implemented
EV5	FR5	The system must be able to pull the new data from the database to recreate the new dataset for retraining	M	UC:06	Implemented

EV6	FR6	The system must be able to process the review text and display the summary output on the GUI	M	UC:07	Implemented
EV7	FR7	The system must be able to use the latest trained model to generate the summary for the review text	M	UC:08	Implemented
EV8	FR8	User review and generated summary should be stored in the database for retraining	M	UC:09	Implemented
EV9	FR9	The system should encrypt the data when saving into the database (both the review and summary)	S	UC:09	Implemented
EV10	FR10	Only domain-specific users can create an account after providing required details.	S	UC:02	Implemented
EV11	FR11	The system could allow the ability to update the account details of the domain user after creating the account	C	UC:02	Implemented
EV12	FR12	The system could be able to perform model retraining automatically during off peak hours every day.	C	UC:03	Implemented
EV13	FR13	The system could also find the sentiment of the generated summary if its positive or negative and return the result	C	UC:08	Implemented
EV14	FR14	The system could make use of a hybrid model for the text summarization.	C	UC:08	Not-Implemented
EV15	FR15	Combine data from common domain groups for dataset creation only with approved consent.	C	UC:06	Implemented

EV16	FR16	The system could allow the domain users to delete the reviews from the database.	C	UC:10	Implemented
EV17	FR17	The system could automate and update the model architecture to increase performance	C	UC:04	Implemented
The percentage of functional requirements that have been fulfilled = $\frac{16}{17} * 100 = 94.11\%$					

### G.3. Evaluation of non-functional requirements

Table 60: Evaluation of the implementation of non-functional requirements

ID	Specification	Description	Priority	Evaluation
NFR1	Usability	The system needs to be simple enough for non-technical individuals to utilize without much effort.	M	Implemented
NFR2	Performance	Summary generation should be done within 3000ms	M	Implemented
NFR3	Maintainability	Following coding standards and best practices	M	Implemented
NFR4	Generalization	Application can be used by any domain users and model adapts to the domain.	M	Implemented
NFR5	Usability	Meaningful error messages should be displayed if anything goes wrong	C	Implemented
NFR6	Security	The system should protect against data corruption by attackers, and testing can ensure this.	C	Implemented
NFR7	Scalability	The prototype must support many concurrent user-requests from multiple businesses under a single domain.	C	Halfway - Implemented (Database is set to be scaled)
The percentage of non-functional requirements that have been fulfilled = $\frac{6.5}{7} * 100 = 92.9\%$				

Table 61: Evaluation of the achievement of design goals

<b>ID</b>	<b>Goal</b>	<b>Evaluation</b>
DG1	Performance	Achieved
DG2	Correctness	Achieved
DG3	Usability	Achieved
DG4	Adaptability	Achieved
DG5	Scalability	Halfway achieved (database)
The percentage of achievement of design objectives. $\frac{3.5}{4} * 100 = 87.5\%$		

## APPENDIX H – CONCLUSION

---



---

### H.1. Status of Research Objectives

Table 62: Status of research objectives

Objective	Description	Status
Problem Identification	<p>Comprehend and document the identified issue.</p> <p><b>RO1:</b> Perform research in a domain of interest and identify a sufficiently comprehensive problem that needs to be addressed.</p> <p><b>RO2:</b> Thoroughly explore and analyze potential solutions for addressing the problem.</p> <p><b>RO3:</b> Explore methods for developing an adaptive and generalized approach.</p> <p><b>RO4:</b> Create a schedule, determine associated deliverables, and develop a Gantt chart for the project.</p>	Completed
Literature Review	<p>Complete a thorough critical review of earlier related work.</p> <p><b>RO1:</b> Make a preliminary investigation on existing abstractive text summarization using deep learning approaches.</p> <p><b>RO2:</b> Make a preliminary investigation on why transformers architecture was the chosen deep learning choice for this research.</p> <p><b>RO3:</b> Analyze the top tier transformer architectures widely used.</p> <p><b>RO4:</b> Analyzing how the models can be fine-tuned via hyperparameter optimization.</p> <p><b>RO5:</b> Analyzing the different approaches used for model evaluation.</p>	Completed

	<b>RO6:</b> Analyze how the model can be generalized for every other domain.	
Requirement Elicitation	<p>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.</p> <p><b>RO1:</b> Gathering information related to the expected metadata required for the dataset to contain for the model training.</p> <p><b>RO2:</b> Gathering the requirements of transformer architectures for fine-tuning and understand the end to end user expectations.</p> <p><b>RO3:</b> Getting insights from domain experts to build a suitable system.</p> <p><b>RO4:</b> Gathering the requirements for handling generalization.</p>	Completed
Design	<p>Considering the following when developing the suggested system:</p> <p><b>RO1:</b> Design a component to preprocess the dataset for the respective model inputs.</p> <p><b>RO2:</b> Design a component to store the top tier transformer models with their respective metadata, to use throughout.</p> <p><b>RO3:</b> Design a hyperparameter tuning component that can improve accuracy of the transformer model.</p> <p><b>RO4:</b> Design high-level architecture for the system.</p>	Completed
Implementation	<p>Setting up a mechanism capable of addressing the gaps that were intended to be covered.</p> <p><b>RO1:</b> To develop data preprocessing component.</p> <p><b>RO2:</b> To develop a component that handles and stores the top tier transformer architectures for fine-tuning.</p> <p><b>RO3:</b> To develop the automated hyperparameter search component that handles all the top tier architectures assigned.</p> <p><b>RO4:</b> To develop a component for the model evaluations for the measured hyperparameters</p>	Completed

	<b>RO5:</b> Consider legal, social, ethical, and professional issues during implementation.	
Evaluation	<p>Effectively test the algorithm implemented, the system, and the respective data science model using recommended techniques.</p> <p><b>RO27:</b> Evaluate the developed algorithm and the respective model against the evaluation metrics researched in the literature review.</p> <p><b>RO28:</b> Create a test plan &amp; test cases and perform unit, performance, and integration testing.</p>	Completed
Documentation	<p>Record the progress of the research project and report any encountered challenges.</p> <p><b>RO29:</b> Produce a cohesive report that details newly acquired skills, evaluations, contributions, etc., and verify that all previously stated objectives have been accomplished.</p>	Completed

## H.2. Achievement of Learning Outcomes

Table 63: Achievement of learning outcomes

Description	LO(s)
The project was divided into smaller problems, and each problem was further broken down into manageable parts. Each part was addressed separately by using relevant techniques obtained from analyzing literature and project requirements.	LO1
The project plan was designed by identifying the components and setting them as milestones to be accomplished within a specific timeframe, ensuring the project's timely completion.	LO2
The author collected and analyzed project requirements from two groups: academic researchers and end-users, which provided valuable insights for developing the system and determining which features to prioritize.	LO3
The author critically reviewed the literature to comprehend the concepts pertinent to the domain.	LO4

Upon obtaining the requirements, insights, and knowledge, the author proceeded to address the two subproblems incrementally, acquiring new skills where necessary. The supervisor provided regular guidance to ensure that the project was progressing as planned, with milestone deliverables being produced accordingly. Any SLEP considerations were considered and documented as well.	LO5, LO6, LO7
The author documented the progress of the research by presenting each chapter to the supervisor as a milestone and incorporating feedback from both the supervisor and module leader. Prior to completing the dissertation, two document artifacts - the project proposal and PSDP - were submitted. Additionally, the author presented papers at conferences to validate their proposed solution.	LO8

### H.3. Project Plan

Figure 66: Gantt Chart: Initial Plan (Self-Composed)

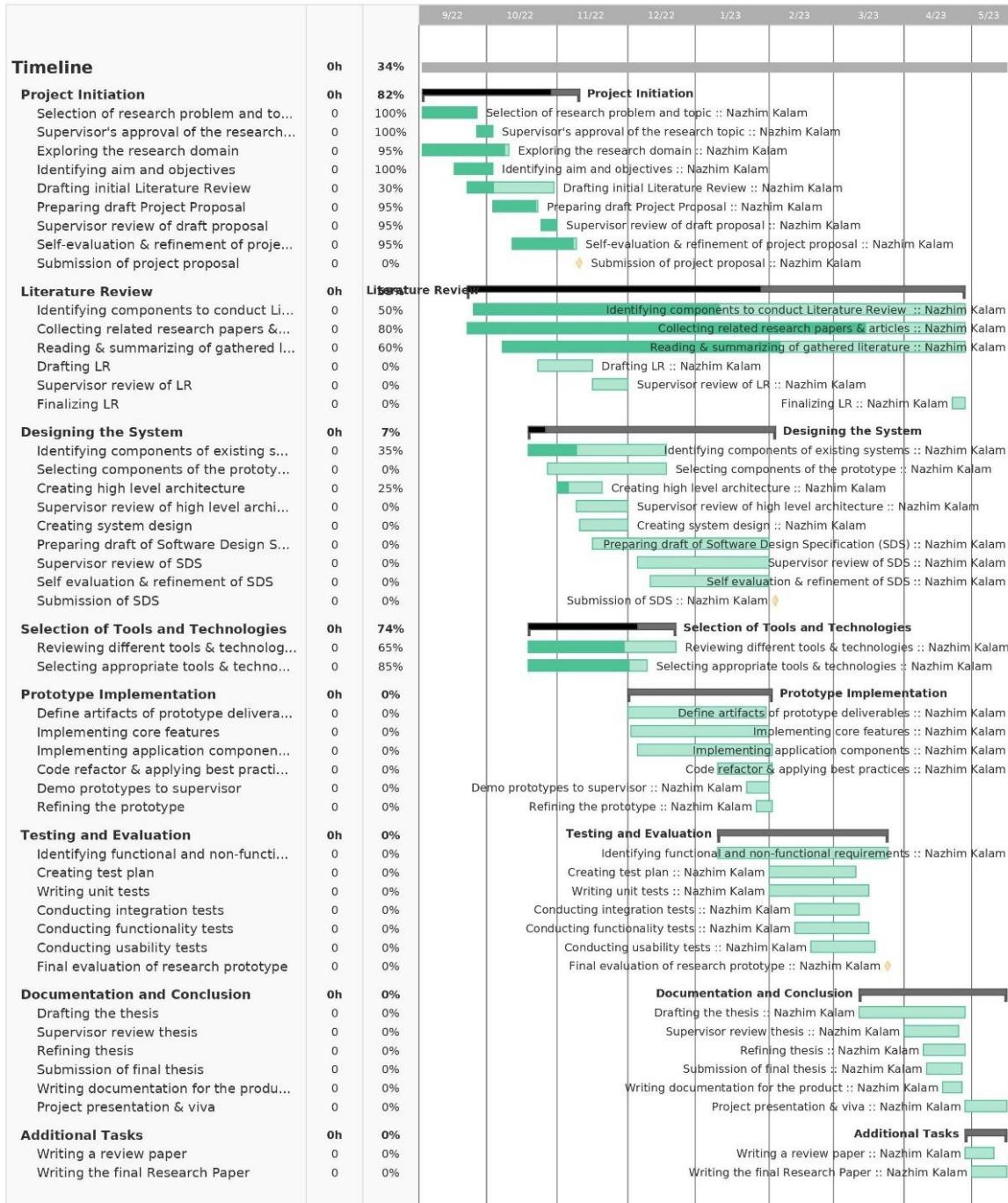
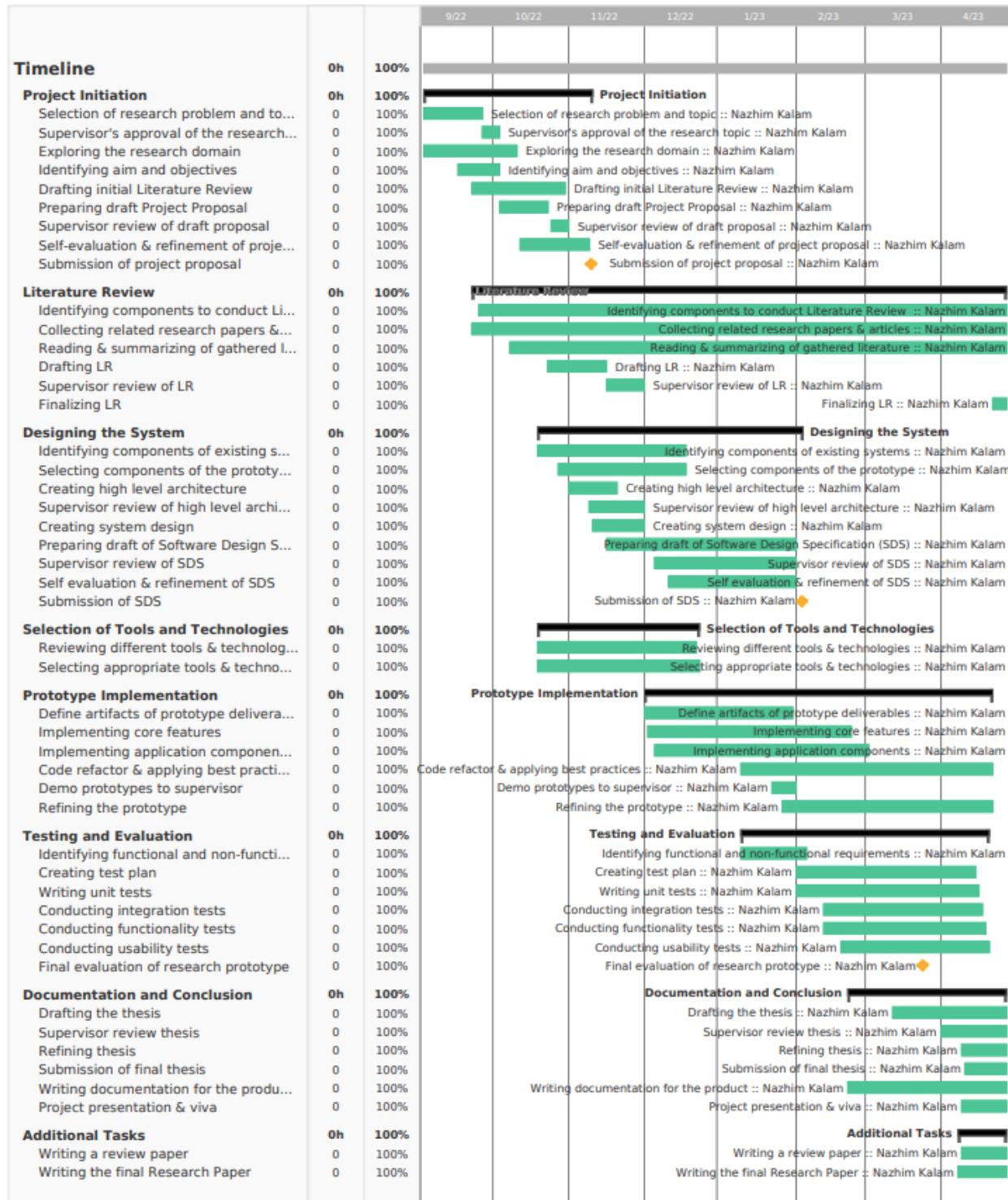


Figure 67: Gantt Chart: Final Plan (Self-Composed)

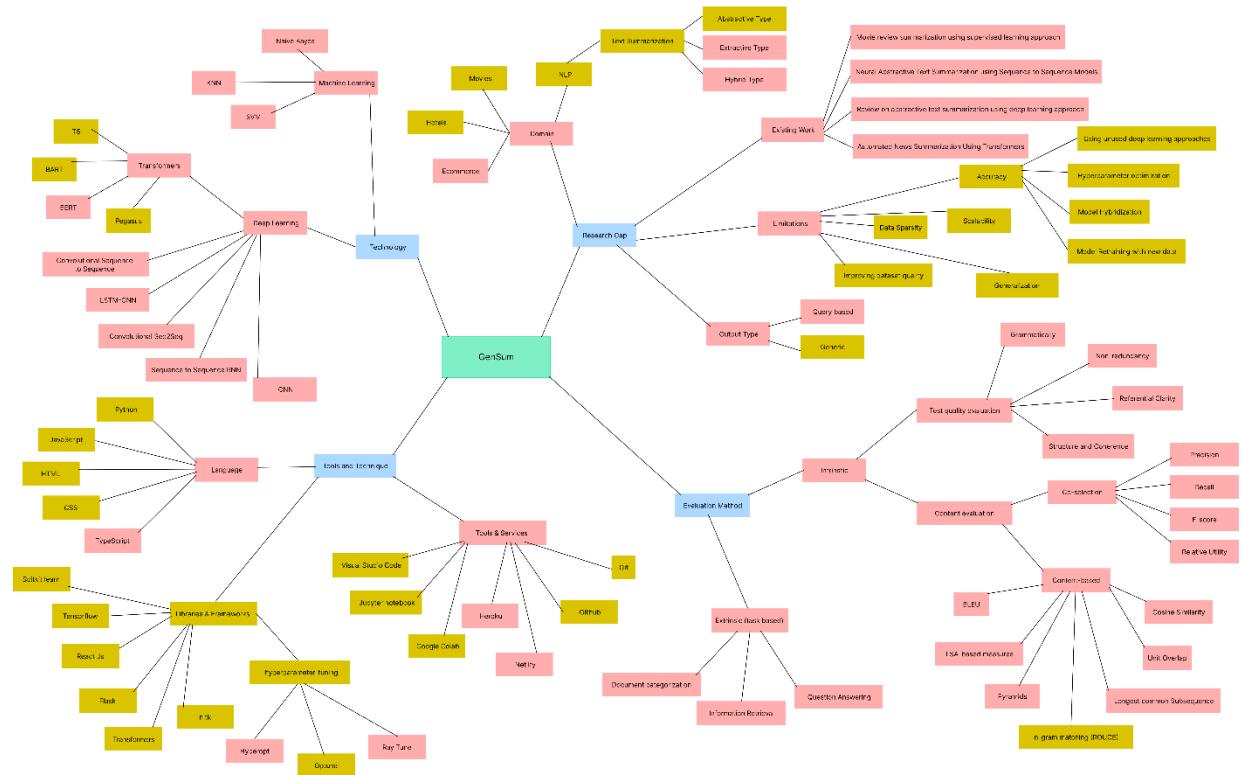


## APPENDIX I – CONCEPT MAP

---



---



An enhanced version of the concept map can be found [here](#).

## APPENDIX J – REVIEW PAPER

---

---

### A Review on Creating a Performance Adaptive Generalized Abstractive text Summarization using Optimized Transformers

NAZHIM KALAM, University of Westminster, UK

TORIN WIRASINGHA, Informatics Institute of Technology, Sri Lanka

Abstractive text summarization systems have been integrated with various applications in the world to perform text summarization, and it's nothing new to the field. However, with prior research it found that in the domain of movies the need for performance improvement is required using the 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 researchers from Google Brain introduced NLP Transformers, which is the latest approach to solving NLP problems, and it's increasingly been known and used nowadays over traditional ML DL approaches like using basic LSTM, and 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 that can be adapted with respect to any domain (be it hotels, movies, restaurants) and increase its performance as the system gets used over time.

This paper is a review of the approach taken to explore and construct a performance-adaptive generalized abstractive text summarizer using optimized transformers. The approach taken to optimize transformers will be discussed in this review.

**CCS Concepts:** • Computing methodologies; • Artificial intelligence; • Natural language processing; • Natural language generation;

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

**ACM Reference Format:**

Nazhim Kalam and Torin Wirasingha. 2022. A Review on Creating a Performance Adaptive Generalized Abstractive text Summarization using Optimized Transformers. 1, 1 (April 2022), 8 pages. <https://doi.org/10.1145/nnnnnnn.nnnnnnn>

#### 1 INTRODUCTION

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. [13]

Online movie reviews are evolving into an important information source for users, with the continuous increase in data on the web [19]. 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. [13].

---

Authors' addresses: Nazhim Kalam, nazhimkalam@gmail.com, University of Westminster, 309 Regent St., London, UK; Torin Wirasingha, Informatics Institute of Technology, 57 Ramakrishna Rd, Colombo 06, Sri Lanka, torin.w@iit.ac.lk.

---

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2022 Association for Computing Machinery.

Manuscript submitted to ACM

Manuscript submitted to ACM

1

Text summary assist users or business decision-makers by compiling and analyzing a significant number of online reviews. [3]

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 [7]

## 2 BUSINESS 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 [22]

On the other hand, companies like Netflix or Amazon Prime can use movie summaries to help users understand their watching pattern or their interest. Likewise, movie-related industries need to allow customers to quickly scan the summary and quickly decide whether they should be watching it or not [13]

## 3 TEXT SUMMARIZATION AND TECHNIQUES

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 lengthy textual content [23] 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 [10]

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 [13]

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 [3]. The abstractive text summarization technique aims to produce 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 [8]. Additionally, it is well-recognized that a strong abstractive summary encompasses the input's key details and is linguistically fluent [27].

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 [11] 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 [14]

#### 4 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 [18].

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 [24].

#### 5 NLP TRANSFORMER MODELS

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 [26].

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 [6].

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 [26].

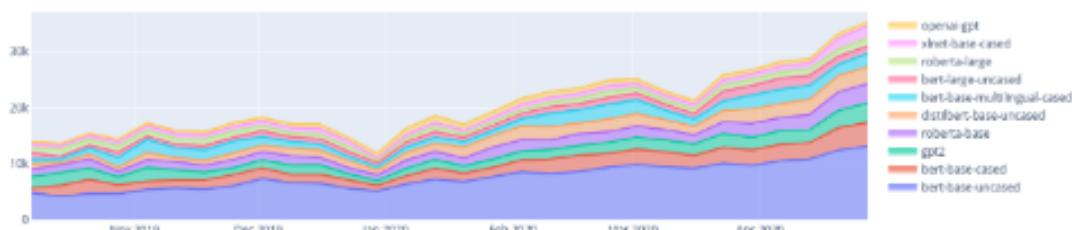


Fig. 1. Transformer Architecture Downloads Rate [26]

[10] 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.

#### 6 AUTOMATED HYPERPARAMETER MODEL 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 [17]. 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 [12]

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 warm up ratio. As a result, giving critical hyperparameters a higher priority is crucial [9]

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 [12]

## 7 MODEL 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 [2]

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 [1]

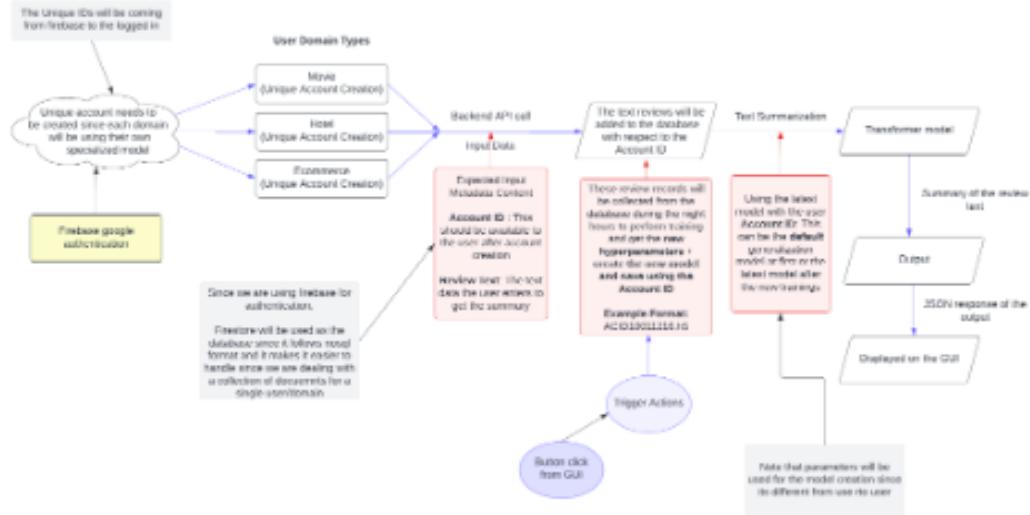


Fig. 2. Proposed Generalized Abstractive Summarization System Process Flow (*self-composed*)

## 8 ALGORITHMIC APPROACHES FOR TEXT SUMMARIZATION

The study of [13] starts by first focusing on feature extraction, then transforming reviews into vector spaces, and applying the Naïve 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.

[6] 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 [21] 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 [10] 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

## 9 USAGE OF TRANSFORMERS

[11] 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.

[10] 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.

## 10 MACHINE LEARNING TEXT SUMMARIZATION TECHNIQUES

[5] 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.

Manuscript submitted to ACM

[13] 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.

## 11 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 [10]

[23] 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 [10]

## 12 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 [20]

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 [15]

## 13 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  
Manuscript submitted to ACM

A Review on Creating a Performance Adaptive Generalized Abstractive text Summarization using Optimized Transformers

7

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 [13]

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 [4]

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 [10]

#### 14 EVALUATION TECHNIQUES

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 1. Evaluation techniques for abstractive text summarization

Measure	Description	Objective Orientation
ROUGE	Measures are made by comparison between an automatically generated summary/translation against a group of reference summaries (generally human created summaries)	Positively oriented. Higher, the better.
BLEU	measures the precision (as to how much words in the generated summaries appeared in the human generated summaries)	

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) [16]. ROUGE measures the recall, (according to how frequently the terms from the summaries created by humans appeared in those computers - generated.)

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. [25]. BLEU measures the precision (as to how much words in the generated summaries appeared in the human generated summaries)

#### 15 CONCLUSION

In this review, the author has pointed out the reasons for the need for the need of performance increase for abstractive review summarization and how NLP transformers can be used as the solution.

Moreover the author also discusses how and why the solution is generalized along with the ways in making using automated hyperparameter tools and model retraining, evaluations metrics is also included by the author along with their objected orientations.

## REFERENCES

- [1] [n. d.] [pdf] domain generalization with mixstyle. () . <https://www.semanticscholar.org/reader/4f6cafafc9563a5b904535078df7c74afe39ef59>.
- [2] [n. d.] [pdf] exploring generalization in deep learning. () . <https://www.semanticscholar.org/reader/d53fb3fcecab07a0d70bf466dd473ec6052cc07>.
- [3] Alaa F. Alsauer and Seela Sasi. 2017. Movie review summarization and sentiment analysis using rapidminer. *2017 International Conference on Networks and Advances in Computational Technologies (NetACT)*. doi: 10.1109/nectact.2017.8076790.
- [4] Nasid Habib Barma and Hasnain Heickal. 2022. An automatic abstractive text summarization system. *Dhaka University Journal of Applied Science and Engineering*, 6, 2, 39–48. doi: 10.3329/dujase.v6i2.59217.
- [5] Ravali Boorugu and G. Ramesh. 2020. A survey on nlp based text summarization for summarizing product reviews. *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*. doi: 10.1109/icirca48905.2020.9183355.
- [6] Adrian M. Brătovăanu and Razvan Andonie. 2020. Visualizing transformers for nlp: a brief survey. *2020 24th International Conference Information Visualisation (IV)*. doi: 10.1109/iv51561.2020.00051.
- [7] Kia Dashtipour, Mandar Gogate, Ahsan Adeel, Hadi Larijani, and Amir Hussain. 2021. Sentiment analysis of persian movie reviews using deep learning. *Entropy*, 23, 5, 596. doi: 10.3390/e23050596.
- [8] Shivam Duseja. 2020. Text summarization using deep neural networks. (Aug. 2020). <https://towardsdatascience.com/text-summarization-using-deep-neural-networks-e7ee7521d804>.
- [9] Sylvia Engdahl. 2008. Blogs. (2008). <https://aws.amazon.com/blogs/machine-learning/bring-your-own-hyperparameter-optimization-algorithm-on-amazon-sagemaker/>.
- [10] Abdul Ghaffor Etemad, Ali Imam Abidi, and Meigha Chhabra. 2021. A review on abstractive text summarization using deep learning. *2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*. doi: 10.1109/icrito51393.2021.9596500.
- [11] Vishal Gupta and Gurpreet Singh Lehal. 2010. A survey of text summarization extractive techniques. *Journal of Emerging Technologies in Web Intelligence*, 2, 3. doi: 10.4304/jetwi.2.3.258\_268.
- [12] Johnsymol Joy and Mercy Paul Selvan. 2022. A comprehensive study on the performance of different multi class classification algorithms and hyperparameter tuning techniques using optuna. *2022 International Conference on Computing, Communication, Security and Intelligent Systems (IC3SIS)*. doi: 10.1109/ic3sis54991.2022.9885695.
- [13] Atif Khan, Muhammad Adnan Gul, Mahdi Zareci, R. R. Biswal, Asim Zeh, Muhammad Naeem, Yousaf Saeed, and Naomic Salim. 2020. Movie review summarization using supervised learning and graph based ranking algorithm. *Computational Intelligence and Neuroscience*, 2020, 1–14. doi: 10.1155/2020/7526580.
- [14] Mahira Kirmani, Nida Manzoor Hakak, Mudasir Mohd, and Mohsin Mohd. 1970. Hybrid text summarization: a survey. (Jan. 1970). [http://link.springer.com/10.1007/978-981-13-0589-4\\_7](http://link.springer.com/10.1007/978-981-13-0589-4_7).
- [15] Panagiotis Kouris, Georgios Alexiadidis, and Andreas Stafyllopatis. 2019. Abstractive text summarization based on deep learning and semantic content generalization. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. doi: 10.18653/v1/p19-1501.
- [16] Chin Yew Lin. [n. d.] Rouge: a package for automatic evaluation of summaries. () . <https://aclanthology.org/W04-1013>.
- [17] Xueqing Liu and Chi Wang. 2021. An empirical study on hyperparameter optimization for fine tuning pre-trained language models. (June 2021). <http://arxiv.org/abs/2106.09204>.
- [18] Marc Moreno Lopez and Jugal Kalita. 2017. Deep learning applied to nlp. (Mar. 2017). <http://arxiv.org/abs/1703.03091>.
- [19] Mamtesh M and Seema Mehla. 2019. Sentiment analysis of movie reviews using machine learning classifiers. *International Journal of Computer Applications*, 182, 50, 25–28. doi: 10.5120/ijca201918756.
- [20] Julian John McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs. *Proceedings of the 22nd international conference on World Wide Web*. doi: 10.1145/2483388.2483466.
- [21] Rajdeep Mukherjee, Hari Chandana Peruri, Uppada Vishnu, Pawan Goyal, Sourangshu Bhattacharya, and Niloy Ganguly. 2020. Read what you need: controllable aspect based opinion summarization of tourist reviews. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. doi: 10.1145/3397271.3401269.
- [22] Abraham Pizam and Taylor Ellis. 1999. Customer satisfaction and its measurement in hospitality enterprises. *International Journal of Contemporary Hospitality Management*, 11, 7, 326–339. doi: 10.1108/09596119910293231.
- [23] Tian Shi, Yaser Keneshloo, Naray Ramakrishnan, and Chandan K. Reddy. 2020. Neural abstractive text summarization with sequence to sequence models. (Sept. 2020). <https://arxiv.org/abs/1812.02303>.
- [24] Richard Socher, Yoshua Bengio, and Christopher D. Manning. [n. d.] Deep learning for nlp (without magic). () . <https://aclanthology.org/P12-4005>.
- [25] Josef Steinberger and Karel Ježek. [n. d.] Evaluation measures for text summarization. () . <http://www.cai.sk/ojs/index.php/cai/article/view/37>.
- [26] Thomas Wolf 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*. doi: 10.18653/v1/2020.emnlp-demos.6.
- [27] Haoyu Zhang, Jianjun Xu, and Ji Wang. 2019. Pretraining based natural language generation for text summarization. (Apr. 2019). <http://arxiv.org/abs/1902.09243>.

## APPENDIX K – RESEARCH PAPER

---



---

# An Adaptive Approach for Generalized Text Summarization using Optimized Transformers

Nazhim Kalam

*Computer Science and Engineering  
University of Westminster  
London, UK  
nazhimkalam@gmail.com*

Torin Wirasingha

*Department of Computing  
Informatics Institute of Technology  
Colombo, Sri Lanka  
torin.w@iit.ac.lk*

**Abstract**—This research explores the ways in which to get an optimal solution using Transformer for abstractive text summarization and yet making a generalized solution that can be adapted with respect to any domain (be it hotels, movies, restaurants) and increase its performance as the system gets used over time.

**Index Terms**—Natural Language Processing, Machine Learning, Deep Learning, Recall-Oriented Understudy for Gisting Evaluation, Inductive logic programming

### I. INTRODUCTION

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. [1]. 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 [2].

#### A. 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 [3].

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 [4].

#### B. 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 [5].

On the other hand, companies like Netflix or Amazon Prime can use movie summaries to help users understand their watching pattern or their interest. Likewise, movie-related industries need to allow customers to quickly scan the summary and quickly decide whether they should be watching it or not [6].

#### C. Text Summarization

With the massive accumulation of information/data on the internet nowadays, it is extremely difficult to extract relevant information from numerous textual documents. The goal of text summarizing is to provide a condensed yet meaningful version of lengthy textual content [7].

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 [1].

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 to watch a certain movie [6].

#### D. 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 [8]. The abstractive text summarization technique aims to produce 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 [1]. Additionally, it is well-recognized that a strong abstractive summary encompasses the input's key details and is linguistically fluent [9].

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 [10]. 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 [11]

#### *E. 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 [12].

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 [13].

#### *F. Transformers*

The open-source library Transformers contain 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 [2].

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 [14]. 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 pre-trained transformer model architectures between Oct 2019 to May 2020 [2].

[1] the research compares various other researchers' approaches taken in order to perform abstractive text summarization, these techniques include the use of transformers and other neural networks approach such as CNN and LSTM RNN networks. The research comparison table below only includes the approaches of transformers used, taken in abstractive text summarization.

#### *G. 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 [15]. 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 [16].

There are several hyperparameters that play a significant role in performance enhancement; however, not all the parameters do so; just a select handful do, for example, learning rate, weight decay, number of epochs, batch size, and warm-up ratio. As a result, giving critical hyperparameters a higher priority is crucial [17].

Automated framework tools, such as Optuna, an open-source framework for hyperparameter optimization built on the Python programming language, do 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 [16].

#### *H. 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 a generalization [1].

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 solve a common problem in multiple domains [18].

#### *I. 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 use of new data from the user end by saving as the model is used is another way of exposing new data for model retraining [19]. 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.

## II. MOTIVATION TO ENHANCE ABSTRACTIVE TEXT SUMMARIZATION PERFORMANCE

The identified problem can also be applied to several other domains which require improving the quality of 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 [20].

As mentioned in the work of [1], syntactic and semantic issues with text summarization were the main issues that researchers were concerned with 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.

In the domain of movie review summarization, currently, there is no research done using the latest deep learning approaches (such as Transformers) to solve this problem, standard machine & deep learning algorithms such as Naïve Bayes, and RNN has been used, and 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 [1].

### III. EXISTING WORK

#### A. Text Summarization Systems

There were multiple studies done previously in the area of text summarization, regarding both abstractive and extractive text summarization. [6] 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 a feature where the users are allowed to quickly recognize the positive and negative aspects of the movie with respect to the review process with. The text summarization approach taken by the author is an extractive approach, where sentence score ranking plays a major role in creating the summary.

The study of [21] is towards the domain of e-commerce but yet related to text summarization for customer reviews on the products they sell, so the purpose is that allow other customers to 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, the abstractive approach is considered to create the summary, which is a better choice of approach.

The research of [4] is another extractive approach for text summarization, where the author develops a solution for generating personalized aspect-based opinion summaries using a dataset that 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 their own words rather than with any sentence ranking technique.

[10] research is a comprehensive comparison study with benchmarking results of various pre-trained 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 used to explore each model, with the 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 [1] is also an abstractive approach to text summarization with the addition of proper grammar and no repeated words used a deep learning approach with RNN and likewise [1] 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.

#### B. Algorithmic approaches for Text Summarization

The study of [6] starts by first focusing on feature extraction, then transforming reviews into vector spaces, and applying the Naïve 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.

[21] research made use of the 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 to the 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 study of [1] focuses on the author's study 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.

#### C. Usage of Transformers

[10] The research employed pre-trained models such as 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 defaulted and might be tuned for better performance. The constraints consist of concentrating on developing more reliable models that can further expand the method to produce summaries of varying lengths and applicable for multi-document summarization.

[1] The author explores 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.

#### IV. PROPOSED SOLUTION APPROACH

The diagram below shows how the approach for creating a generalized model is considered and executed using the optimized transformer models.

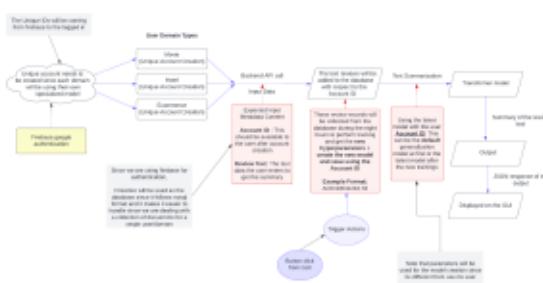


Fig. 1. Proposed Generalized Abstractive Summarization System Architecture

## V. EVALUATION

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.

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) [22]. ROUGE measures the recall, (according to how frequently the terms from the summaries created by humans appeared in those computers - generated.)

$$ROUGE-1 = \frac{\sum_{\text{Reference Summary}} \sum_{\text{unigram}} \text{Count}_{\text{match}}(\text{unigram})}{\sum_{\text{Reference Summary}} \sum_{\text{unigram}} \text{Count}(\text{unigram})}$$

Fig. 2. Rouge-1 Score Equation

$$ROUGE-2 = \frac{\sum_{S \in [RefSummarize]} \sum_{\text{bigram}_i \in S} \min(count(i, X), count(i, S))}{\sum_{S \in [RefSummaries]} \sum_{\text{bigram}_i \in S} count(i, S)}$$

Fig. 3. Rouge-2 Score Equation

$$ROUGE-L_{(candidate, references)} = \max_k \{ ROUGE-L_{(candidate, references_k)} \}$$

Fig. 4. Rouge-L Score Equation

BLEU also known as Bilingual Evaluation Understudy is a metric used for the evaluation of the quality of machine-generated text by comparing it with a reference text that is supposed to be generated. [23]. BLEU measures the precision (as to how many words in the generated summaries appeared in the human-generated summaries)

$$\text{BLEU} = \frac{\text{Number of words in the summary which are in gold standard}}{\text{Total number of words in the summary}}$$

Fig. 5. BLEU Score Equation

The ROUGE score was used as the final evaluation metric for this research since the weightage of it is the best metric for this research as proven by previous work.

BART, T5 & Pegasus transformer architecture was experimented with for this research, out of which Bart gave the optimal results whereas T5 came close however Pegasus failed, there can be several reasons why Pegasus failed one of which can be due to the difference in the model architecture and the type of problem it can solve with respect to the dataset.

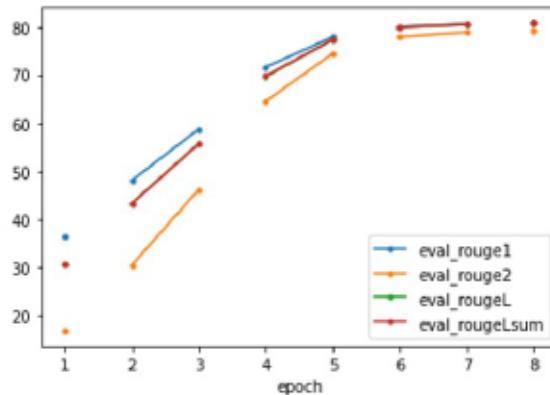


Fig. 6. Validation accuracy by the number of epochs - BART Model

ROUGE1 of 80.8, ROUGE2 of 79.42, ROUGEL L of 80.8, and ROUGELSUM of 80.8 was the optimal evaluation metric result achieved from the BART model giving the best result.

## VI. FUTURE ENHANCEMENTS

Future Enhancements for text summarization models can be achieved through the use of transformer hybridization. This approach could lead to improved performance, enabling the models to better handle complex natural language processing tasks. Another potential improvement is the inclusion of text paraphrase models for user reviews.

This would help to address the potential issue of inaccuracies in user-generated content. Additionally, applying keyword extraction for sentiment classification in review summaries could provide valuable insights for domain users to improve their services.

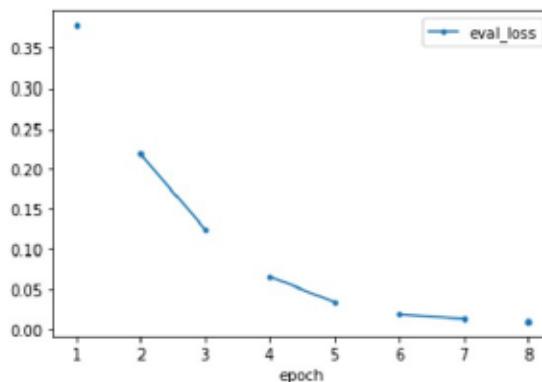


Fig. 7. Validation loss by the number of epochs - BART Model

By identifying which keywords contributed to the sentiment classification, businesses can understand their customers better and make informed decisions to enhance their services. These advancements can significantly benefit the field of natural language processing and improve the accuracy and efficiency of text summarization models.

## VII. CONCLUSION

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 with respect to any domain.

## REFERENCES

- [1] A. G. Etemad, A. I. Abidi, and M. Chhabra, "A review on abstractive text summarization using deep learning," *2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 2021. DOI: 10.1109/icrito51393.2021.9596500.
- [2] T. Wolf, L. Debut, V. Sanh, et al., "Transformers: State-of-the-art natural language processing," *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020. DOI: 10.18653/v1/2020.emnlp-demos.6.
- [3] J. J. McAuley and J. Leskovec, "From amateurs to connoisseurs," *Proceedings of the 22nd international conference on World Wide Web*, 2013. DOI: 10.1145/2488388.2488466.
- [4] R. Mukherjee, H. C. Peruri, U. Vishnu, P. Goyal, S. Bhattacharya, and N. Ganguly, "Read what you need: Controllable aspect-based opinion summarization of tourist reviews," *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020. DOI: 10.1145/3397271.3401269.
- [5] A. Pizam and T. Ellis, "Customer satisfaction and its measurement in hospitality enterprises," *International Journal of Contemporary Hospitality Management*, vol. 11, no. 7, pp. 326–339, 1999. DOI: 10.1108/09596119910293231.
- [6] A. Khan, M. A. Gul, M. Zareei, et al., "Movie review summarization using supervised learning and graph-based ranking algorithm," *Computational Intelligence and Neuroscience*, vol. 2020, pp. 1–14, 2020. DOI: 10.1155/2020/7526580.
- [7] T. Shi, Y. Keneshloo, N. Ramakrishnan, and C. K. Reddy, *Neural abstractive text summarization with sequence-to-sequence models*, Sep. 2020. [Online]. Available: <https://arxiv.org/abs/1812.02303>.
- [8] A. F. Alsager and S. Sasi, "Movie review summarization and sentiment analysis using rapidminer," *2017 International Conference on Networks amp; Advances in Computational Technologies (NetACT)*, 2017. DOI: 10.1109/netact.2017.8076790.
- [9] H. Zhang, J. Xu, and J. Wang, *Pretraining-based natural language generation for text summarization*, Apr. 2019. [Online]. Available: <https://arxiv.org/abs/1902.09243>.
- [10] V. Gupta and G. S. Lehal, "A survey of text summarization extractive techniques," *Journal of Emerging Technologies in Web Intelligence*, vol. 2, no. 3, 2010. DOI: 10.4304/jetwi.2.3.258-268.
- [11] M. Kirmani, N. Manzoor Hakak, M. Mohd, and M. Mohd, *Hybrid text summarization: A survey*, Jan. 1970. [Online]. Available: [http://link.springer.com/10.1007/978-981-13-0589-4\\_7](http://link.springer.com/10.1007/978-981-13-0589-4_7).
- [12] M. M. Lopez and J. Kalita, *Deep learning applied to nlp*, Mar. 2017. [Online]. Available: <http://arxiv.org/abs/1703.03091>.
- [13] R. Socher, Y. Bengio, and C. D. Manning, *Deep learning for nlp (without magic)*. [Online]. Available: <https://aclanthology.org/P12-4005>.
- [14] A. M. Brasoveanu and R. Andonie, "Visualizing transformers for nlp: A brief survey," *2020 24th International Conference Information Visualisation (IV)*, 2020. DOI: 10.1109/iv51561.2020.00051.
- [15] X. Liu and C. Wang, *An empirical study on hyperparameter optimization for fine-tuning pre-trained language models*, Jun. 2021. [Online]. Available: <http://arxiv.org/abs/2106.09204>.
- [16] J. Joy and M. P. Selvan, "A comprehensive study on the performance of different multi-class classification algorithms and hyperparameter tuning techniques using optuna," *2022 International Conference on Computing, Communication, Security and Intelligent Systems (IC3SIS)*, 2022. DOI: 10.1109/ic3sis54991.2022.9885695.
- [17] S. Engdahl, *Blogs*, 2008. [Online]. Available: <https://aws.amazon.com/blogs/machine-learning/bring-your-own-hyperparameter-optimization-algorithm-on-amazon-sagemaker/>.

- [18] *[pdf] domain generalization with mixstyle*. [Online]. Available: <https://www.semanticscholar.org/reader/4f6eafafc9563a5b904535078df7e74afe39ef59>.
- [19] N. H. Barna and H. Heickal, "An automatic abstractive text summarization system," *Dhaka University Journal of Applied Science and Engineering*, vol. 6, no. 2, pp. 39–48, 2022. DOI: 10.3329/dujase.v6i2.59217.
- [20] P. Kouris, G. Alexandridis, and A. Stafylopatis, "Abstractive text summarization based on deep learning and semantic content generalization," *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019. DOI: 10.18653/v1/p19-1501.
- [21] R. Boorugu and G. Ramesh, "A survey on nlp based text summarization for summarizing product reviews," *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, 2020. DOI: 10.1109/icirca48905.2020.9183355.
- [22] C.-Y. Lin, *Rouge: A package for automatic evaluation of summaries*. [Online]. Available: <https://aclanthology.org/W04-1013>.
- [23] J. Steinberger and K. Ježek, *Evaluation measures for text summarization*. [Online]. Available: <http://www.cai.sk/ojs/index.php/cai/article/view/37>.

## APPENDIX L – PROOF OF SUBMISSION

Figure 68: Research & Review Paper Submission

The screenshot shows the SmartNets 2023 EDAS Conference and Journal Management System interface. At the top, there is a navigation bar with links for Home, Register, Travel grants, My..., and Help. Below the navigation bar, the title "EDAS Conference and Journal Management System" is displayed. There are four informational boxes with icons and text: 1. You can register and submit your paper! 2. Please indicate whether you want to receive occasional email updates about new EDAS features. 3. Papers in which languages can you review? 4. Your conflicts-of-interest have not been updated in the last three months. (Persons with conflicts-of-interest are those who should not review papers of yours, e.g., because they are close friends, collaborators, former PhD students or work in the same institution.) Under the heading "My pending, active and accepted papers", it says "Only papers for upcoming and recently-concluded conferences and journal issues are shown." A table lists two papers:

Conference	Paper title (details)	Status	Edit	Add and delete authors	Withdraw	Registration	Review manuscript
SmartNets 2023 - BDA-ML	<i>A Review on Creating a Performance Adaptive Generalized Abstractive Text Summarization Using Optimized Transformers</i>	Active (has manuscript)					until May 2, 2023 05:29 Asia/Colombo
SmartNets 2023 - BDA-ML	<i>An Adaptive Approach for Generalized Text Summarization Using Optimized Transformers</i>	Active (has manuscript)					until May 2, 2023 05:29 Asia/Colombo

The above figure shows the proof of the review and research paper submission at SmartNETs 2023 under EDAS Conference and Journal Management System.

