

Estimating Story Points in Scrum: Balancing Accuracy and Interpretability with Explainable AI

I.N. Walpita Gamage
2024



Estimating Story Points in Scrum: Balancing Accuracy and Interpretability with Explainable AI

I.N. Walpita Gamage
Index No : 19001827

Supervisor: Dr. H.K.T.C. Halloluwa
Co-Supervisor: Mr. R.M.U.A. Rathnayake

May 2024

Submitted in partial fulfilment of the requirements of the
B.Sc. (Honours) in Computer Science Final Year Project



Declaration

I certify that this dissertation does not incorporate, without acknowledgement, any material previously submitted for a degree or diploma in any university and to the best of my knowledge and belief, it does not contain any material previously published or written by another person or myself except where due reference is made in the text. I also hereby give consent for my dissertation, if accepted, be made available for photocopying and for interlibrary loans, and for the title and abstract to be made available to outside organizations.

Candidate Name: I.N. Walpita Gamage



.....

Signature of Candidate

Date: 28/09/2024

This is to certify that this dissertation is based on the work of Mr. I.N. Walpita Gamage under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Supervisor's Name: Dr. H.K.T.C. Halloluwa



.....

Signature of Supervisor

Date: 28/09/2024

Abstract

This study addresses the challenge of accurate Story Points (SP) estimation in agile software development. SP, a unit for measuring development effort, are crucial for project planning and resource allocation. However, manual SP estimation is critical yet a tedious and error-prone process, often causing delays and exceeding project budgets. This highlights the pressing need for automated and accurate SP prediction in the software development industry.

This study addresses this need by proposing a novel data preprocessing approach for story point estimation. It involves removing similar user stories, data augmentation and description segmentation. Furthermore, the study contributes 3 new datasets to the public domain specifically designed for story point estimation research. This enhanced data richness and diversity are shown to significantly improve model performance. The study leverages these 3 datasets and the Choetkiertikul dataset to train various traditional and transformer models, including Support Vector Machines (SVM), Random Forest, Recurrent Neural Network (RNN), Bidirectional Encoder Representations from Transformers (BERT), DistilBERT and RoBERTa. Among the traditional approaches, SVM achieved the highest accuracy (46.12%). BERT outperformed other transformer models (44.58%) but fell slightly short of SVM's performance.

To enhance model transparency and interpretability, the study employed Local Interpretable Model-agnostic Explanations (LIME), Shapley Additive explanations (SHAP), and Transformer Interpret libraries. These techniques offer explanations by highlighting keywords influential in the model's predictions. Additionally, a human-based evaluation involving 7 industry professionals was conducted to assess both model performance and the reliability of the Explainable Artificial Intelligence (XAI) methods.

Acknowledgement

This research would not have been possible without the invaluable support and guidance of several individuals. First, I would like to express my sincere gratitude to the University of Colombo School of Computing (UCSC) for providing me with the opportunity to carry out individual research and develop my academic and research skills. My sincere gratitude goes to my supervisor, Dr. H.K.T.C. Halloluwa and my co-supervisor, Mr. R.M.U.A. Rathnayake, for their continuous support and insightful guidance throughout the past 12 months. I would also like to extend my thanks to my colleagues at UCSC for their valuable advice and support. Their input significantly contributed to the refinement of my research ideas. Finally, I am immensely grateful to my parents and brother for motivating and encouraging me. Their constant support throughout this journey has been invaluable.

Table of Contents

Acronyms	1
1 Introduction	2
1.1 Background and Motivation	2
1.2 Problem Statement and Significance	3
1.3 Research Aim	4
1.4 Research Gap	4
1.5 Research Questions and Objectives	4
1.5.1 Research Questions	4
1.5.2 Research Objectives	5
1.6 Research Scope	5
2 Literature Review	6
2.1 Regression Based	6
2.1.1 Deep-SE	6
2.1.2 RNN-CNN model	7
2.2 Classification Based	8
2.2.1 TF-IDF	8
2.2.2 Developer features	10
2.2.3 Graph Based - Text Level GNN-StoryPoint Estimator	11
2.3 Clustering Algorithms	12
2.3.1 K Means and K-medoids clustering	12
2.3.2 Hierarchical clustering	13
2.4 GPT2SP	14
2.5 Explainable AI	15
2.5.1 Approaches of XAI in other research areas	15
3 Design	19
3.1 Research Approach and Methodology	19
3.2 High Level Design	20
3.2.1 Data Collection	20

3.2.2	Pre-processing	24
3.2.3	Design and Implementation of Models for SP Estimation	26
3.2.4	Implementation of XAI methods	27
4	Implementation	34
4.1	Data Collection	34
4.2	Pre-processing	34
4.2.1	Data Filtering	34
4.2.2	Data Cleaning	35
4.2.3	Removing Similar User Stories	36
4.2.4	Label Encoding	37
4.2.5	Tokenization and Vectorization	37
4.2.6	Handling Class Imbalance Using Data Augmentation and Class Weights	38
4.3	Model Training and Experiments	39
4.3.1	Environment	39
4.3.2	Model Evaluation Metrics	40
4.3.3	Model Implimentation	41
4.4	XAI Apporaches	43
4.4.1	LIME	44
4.4.2	SHAP	44
4.4.3	Transformer-Interpret	45
5	Results and Evaluation	47
5.1	First Phase of Experiments Using the Traditional Approaches	47
5.1.1	Experiment I - Using Random Forest	47
5.1.2	Experiment II - Using RNN with Long Short Term Memory (LSTM)	47
5.1.3	Experiment III - Using SVM	48
5.1.4	Conclusion on the First Phase of Experiments Using the Tra- ditional Approaches	48
5.2	Second Phase of Experiments Using the Traditional Approaches	48
5.2.1	Hyper Parameter Tuning	49

5.2.2	Experiment I - Using Various Embedding Techniques	49
5.2.3	Experiment II - Description Segmentation	50
5.2.4	Experiment III - Augmented Dataset	51
5.2.5	Conclusion on Second Phase of Experiments Using the Traditional Approaches	51
5.3	First Phase of Experiments Using the Transformer Models	51
5.3.1	Experiment I - Using RoBERTa	53
5.3.2	Experiment II - Using DilstilBERT	53
5.3.3	Experiment III - Using BERT	53
5.3.4	Conclusion on First Phase of Experiments Using the Transformer Models	54
5.4	Second Phase of Experiments Using the Transformer Models	55
5.4.1	Hyper Parameter Tuning	56
5.4.2	Experiment I - Description Segmentation	56
5.4.3	Experiment II - Augmented Dataset	57
5.4.4	Conclusion on Second Phase of Experiments Using the Transformer Models	57
5.5	Human-Based Evaluation	57
6	Conclusions	64
	Appendix	70

List of Figures

2.1	The Deep learning model	7
2.2	Evaluation results of Deep-SE over the 3 baselines	8
2.3	RNN-CNN model	9
2.4	Performance metrics for the industrial project	9
2.5	Performance metrics for the SVM model in all projects	10
2.6	Evaluation results for the 3 cases	11
2.7	Architecture of TextGraph GNN model	11
2.8	TextLevelGNN model evaluation with random forest	12
2.9	Steps for proposed study	13
2.10	The architecture of the GPT2SP model	14
2.11	Machine learning model’s explainability and accuracy	15
2.12	Survey results for the need of XAI	16
2.13	XAI approach: Predicted output score with a confidence score and a similar answer	17
2.14	XAI approach: Predicted output score with highlighted words in the input answer	17
2.15	Results of LIME, SHAP and CAM on DHIS and IOU	18
3.1	Flow of deductive Research	19
3.2	Design science research steps	20
3.3	High-Level Research Design	21
3.4	The overview of dataset A	22
3.5	The overview of dataset B	22
3.6	Data distributed of the TAWOS dataset	23
3.7	SP distributed of the TAWOS dataset	23
3.8	SP distribution of the 39 projects (first set)	28
3.9	SP distribution of the 39 projects (second set)	29
3.10	SP distribution of the 39 projects (third set)	30
3.11	SP distribution of the 39 projects (fourth set)	31
3.12	SP distribution of the 39 projects (fifth set)	32

3.13	SP distribution of the 39 projects (sixth set)	33
3.14	How BERT tokenizer works for unknown words	33
4.1	Tables in the TAWOS dataset	34
4.2	The structure of the issue table	35
4.3	Construction of Dataset A and Dataset B	35
4.4	Confusion Matrix experiment	40
4.5	Class-wise accuracy experiment	41
4.6	Explainability output from LIME	44
4.7	Explainability output from SHAP	45
4.8	Explainability output from Transformer-Interpret	46
5.1	The architecture of the second experiment	52
5.2	The answers provided by a participant	58

List of Tables

4.1	Confusion Matrix	40
5.1	A summary of the performance of traditional approaches	49
5.2	Performance of the SVM model on various n-gram ranges	50
5.3	Performance of the SVM model on various word chunk lengths	50
5.4	A summary of the performance of transformer models on the 3 datasets	55
5.5	A summary of the performance of transformer models on the 4 clas- sifiers	55
5.6	Performance of the BERT model on various word chunk lengths	57
5.7	Comparison of the SP predictions approaches	59
5.8	The keywords that each of the XAI approaches used to determine the SP value	63

Acronyms

ML Machine Learning

NLP Natural Language Processing

AI Artificial Intelligence

XAI Explainable Artificial Intelligence

GPT Generative Pre-trained Transformer

BERT Bidirectional Encoder Representations from Transformers

DL Deep Learning

RNN Recurrent Neural Network

DNN Deep Neural Network

CNN Convolutional Neural Network

GNN Graph Neural Networks

LSTM Long Short Term Memory

BiLSTM Bi-directional Long Short Term Memory

SP Story Points

MAE Mean Absolute Error

SA Standardized Accuracy

MdAE Median Absolute Error

MMRE Mean Magnitude of Relative Error

SVM Support Vector Machines

NB Naive Bayes

KNN K-Nearest Neighbor

DT Decision Tree

NN Neural Networks

TF-IDF Term Frequency-Inverse Document Frequency

RHN Recurrent Highway Network

LIME Local Interpretable Model-agnostic Explanations

SHAP Shapley Additive explanations

SQL Structured Query Language

RBF Radial Basis Function

Chapter 1 - Introduction

1.1 Background and Motivation

Scrum is a widely used Agile methodology for software development which emphasizes an iterative development process [1]. This is used when the requirements are ambiguous and it focuses on client-user engagement. The Agile development process starts with the customer representatives providing the product owners with a list of work items or in other words user stories to be completed in order to develop the software. Then the team evaluates and refines the user stories in the product backlog as the initial phase of the product backlog refinement process. Then the team does work breakdowns for larger items (i.e. epics) by creating a collection of smaller work items and prioritizing the items by estimating the effort needed to complete each task. In order to deliver a working product, the team completes a sprint planning process that includes defining the sprint goal, assessing team capacity, selecting tasks for a sprint backlog based on team capacity and completing the sprint in a certain number of iterations [2]. Overall Agile development involves developing software in repeated iterations and in small incremental parts simultaneously. The development team designs, implements, tests and delivers a working release in each iteration (sprint) of the project.

The problem arises when refining the product backlog which is also one of the most critical aspects of the Agile process that involves estimating each user story. A formal, comprehensive explanation of a software feature written from the client's or end-user perspective is known as a "user story" and the development team has multiple user stories to complete in each sprint. Consequently, it is vital to estimate the work necessary to fulfil each user story rather than concentrating on the whole project. An effort estimation technique is used for this calculation. Using effort estimation, Agile teams evaluate the size that a product backlog item should be in relation to how much work it will take to complete. Since it enables teams to efficiently schedule projects and distribute resources, it is an essential element in

Agile software development. Story Points (SP), the use case point approach and FPA are a few techniques for effort estimation. Among them, SP has become more popular in Agile projects because of how easy it is to use. Furthermore, SP was the most commonly used sizing metric for sprints according to [3].

SP are one of the approaches to estimating user stories in Scrum. SP are a unit of measure used to convey an estimation of the total effort needed to complete a task or a user story in the product backlog. There are several SP estimation techniques such as Planning Poker which uses either the Fibonacci sequence or numeric sequence from 1 to 10 and T-shirt size estimation which uses values such as XS, S, M and L. SP are often estimated by the team based on team consensus utilizing a variety of factors such as analogy and expert opinion while taking into account the amount of effort, complexity, risk and uncertainty [4]. This might lead to inconsistent and inaccurate estimates. [3] draws attention to the possibility of bias in the subjective estimation based on the knowledge of domain experts. Therefore, such inaccurate SP estimates (overestimates or underestimates) might result in unproductive sprint planning, which would waste developer time and result in lost productivity, increased costs, project failure, unhappy customers, and lost revenue [2]. Various Machine Learning (ML) and Artificial Intelligence (AI) approaches have been used to address this problem however a more efficient, consistent and accurate task-level effort-measuring approach which explains the decision-making process is essential.

1.2 Problem Statement and Significance

SP estimation of user stories in Agile is a critical aspect of the development process [5]. Therefore the accuracy of the SP estimation models is essential. Even though there were several models built using different approaches, there's still room to improve accuracy. However to improve the accuracy of the existing models, we'll have to adopt much more complex ML models such as transformer models. However, these models are usually called "black boxes" models as it is difficult for users to identify the factors which resulted in the prediction [6]. The user's acceptance and confidence in SP estimations are impacted by the lack of transparency of the

models. Thus, to build a more reliable SP estimation model, it needs to give explanations as to why it generated the SP estimation. Therefore, applying XAI to the model should be done.

1.3 Research Aim

To create an approach to estimate SP values accurately and generate explanations for each estimation by using ML and XAI methods.

1.4 Research Gap

There are many ML models built for SP estimation and for cross-project evaluation the state-of-the-art accuracy is 47.2%. Most of the previous literature had used the Choetkiertikul dataset which consists of 23,313 user stories and the remaining literature had used datasets which was provided by a client which is not publicly available. Almost all of the previous literature had focused on improving the model's accuracy rather than the model's interpretability and only one research had put the foundation for the model's interpretability. I intend to find an approach to estimate SP values by maintaining both accuracy and interpretability. Also, I intend to construct a new and richer dataset than the Choetkiertikul dataset.

1.5 Research Questions and Objectives

1.5.1 Research Questions

- RQ 1** What are other projects that can be combined with the existing dataset to make it richer?
- RQ 2** What are the preprocessing steps needed for the dataset to make it more optimized?
- RQ 3** Can transformer models generate SP values for given user stories more accurately than the traditional approaches?
- RQ 4** How to apply XAI techniques to generate explanations for each SP prediction?

1.5.2 Research Objectives

- RO 1.1** To analyze the currently used datasets and projects for SP estimation.
- RO 1.2** To build a richer dataset than the existing datasets for SP estimation.
- RO 2.1** To explore transformer models and determine which transformer model is best suited and their accuracy.
- RO 2.2** To identify the key challenges when it comes to adapting transformer models for SP estimation.
- RO 2.3** To propose solutions to overcome the identified challenges while adapting transformer models for SP estimation.
- RO 2.4** To evaluate the best-suited transformer model.
- RO 3.1** To explore XAI techniques and determine which technique is more suited.
- RO 3.2** To identify the key challenges when it comes to adapting XAI techniques for generating explanations for each SP prediction.
- RO 3.3** To propose solutions to overcome the identified challenges while adapting XAI techniques for generating explanations.

1.6 Research Scope

This study covers the following aspects,

1. Exploring various projects that can be used for SP estimation.
2. Exploring traditional approaches that can be used for SP estimation.
3. Exploring transformer-based approaches that can be used for SP estimation.
4. Exploring XAI approaches that can be used with ML models.

The project only aimed to determine the SP value by considering the title, description and issue type of a given user story. This did not consider the attributes/features of the developer who is assigned to the user story.

Chapter 2 - Literature Review

There have been several studies done regarding accuracy however only a handful of studies were done regarding the XAI aspect. The studies have focused on regression-based, classification-based, graph-based, clustering and transformer-based models.

2.1 Regression Based

2.1.1 Deep-SE

A new approach was proposed to estimate SP based on combining Recurrent Highway Network Recurrent Highway Network (RHN) and LSTM which are 2 Deep Learning Deep Learning (DL) architectures [7]. The proposed model was an end-to-end prediction system known as Deep-SE. There are many layers in the model and the description is given as the input for the model which then goes through and processed by the layers before estimating the value of the SP. The architecture of the Deep-SE model is given in the figure 2.1.

Word embedding, document representation using LSTM, deep representation using RHN and differentiable regression are the 4 main components of the Deep-SE model. The final layer of the model consists of a regressor which estimates the SP value in the numerical sequence. Their dataset consists of 23,313 user stories from 16 open-source projects. Furthermore, the evaluation showed that the Deep-SE model outperforms Mean, Random Guessing and Median methods which are the 3 widely used baselines with relation to Mean Absolute Error (MAE), Median Absolute Error (MdAE) and Standardized Accuracy (SA). The evaluation results are given in the figure 2.2.

A system was proposed which is a close adaptation of the Deep-SE model over a commercial product dataset [8]. New elements were introduced to see the correlation between the model performance and the level of detail in user stories.

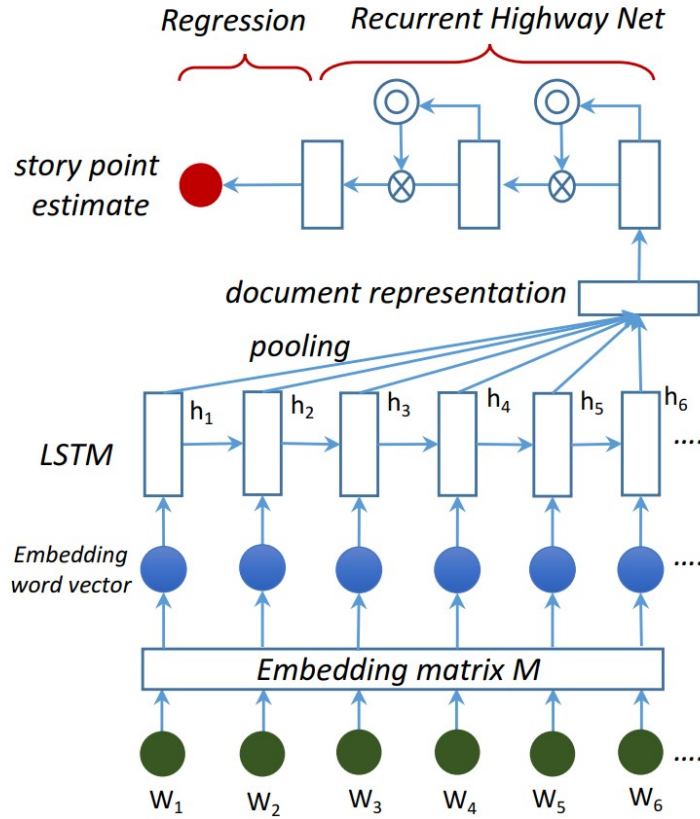


Figure 2.1: The Deep learning model

Also, they introduced an analysis of the model’s performance with respect to the user story types such as bugs and new features. This study shows the Deep-SE’s applicability outside of the open-source community.

2.1.2 RNN-CNN model

[9] proposed a deep learning model which is a combination of both the RNN and Convolutional Neural Network (CNN). When developing the RNN-CNN model, they have used the Bi-directional Long Short Term Memory (BiLSTM) as the RNN to capture the contextual information on both the forward and backward directions. The BiLSTM learns the user story as a vector. And then using that learned vector, CNN extracts the features to predict the SP value. Lastly, a dense network is used to predict the SP values based on the descriptions which were from the learned user stories. The proposed model is given in the figure 2.3. They have used the same dataset which was used to train the Deep-SE model which consists of 23,313 user stories from 16 open-source projects. The performance of the model

Proj	Method	MAE	MdAE	SA	Proj	Method	MAE	MdAE	SA
ME	Deep-SE	1.02	0.73	59.84	JI	Deep-SE	1.38	1.09	59.52
	mean	1.64	1.78	35.61		mean	2.48	2.15	27.06
	median	1.73	2.00	32.01		median	2.93	2.00	13.88
UG	Deep-SE	1.03	0.80	52.66	MD	Deep-SE	5.97	4.93	50.29
	mean	1.48	1.23	32.13		mean	10.90	12.11	9.16
	median	1.60	1.00	26.29		median	7.18	6.00	40.16
AS	Deep-SE	1.36	0.58	60.26	DM	Deep-SE	3.77	2.22	47.87
	mean	2.08	1.52	39.02		mean	5.29	4.55	26.85
	median	1.84	1.00	46.17		median	4.82	3.00	33.38
AP	Deep-SE	2.71	2.52	42.58	MU	Deep-SE	2.18	1.96	40.09
	mean	3.15	3.46	33.30		mean	2.59	2.22	28.82
	median	3.71	4.00	21.54		median	2.69	2.00	26.07
TI	Deep-SE	1.97	1.34	55.92	MS	Deep-SE	3.23	1.99	17.17
	mean	3.05	1.97	31.59		mean	3.34	2.68	14.21
	median	2.47	2.00	44.65		median	3.30	2.00	15.42
DC	Deep-SE	0.68	0.53	69.92	XD	Deep-SE	1.63	1.31	46.82
	mean	1.30	1.14	42.88		mean	2.27	2.53	26.00
	median	0.73	1.00	68.08		median	2.07	2.00	32.55
BB	Deep-SE	0.74	0.61	71.24	TD	Deep-SE	2.97	2.92	48.28
	mean	1.75	1.31	32.11		mean	4.81	5.08	16.18
	median	1.32	1.00	48.72		median	3.87	4.00	32.43
CV	Deep-SE	2.11	0.80	50.45	TE	Deep-SE	0.64	0.59	69.67
	mean	3.49	3.06	17.84		mean	1.14	0.91	45.86
	median	2.84	2.00	33.33		median	1.16	1.00	44.44

Figure 2.2: Evaluation results of Deep-SE over the 3 baselines

was evaluated using the metrics R2 Score and MAE value.

2.2 Classification Based

2.2.1 TF-IDF

An approach was proposed for classifying user stories into several SP classes [10]. Their dataset consists of open-source and commercial issue reports which they gathered from Jira repositories. They used Term Frequency-Inverse Document Frequency (TF-IDF) to extract features from those issue reports in the dataset and each row in the dataset is an issue report while the extracted features are in columns. They used several classifier models to predict the SP for each issue report by feeding the title and description of those issue reports. For the classifier models they used SVM, Naive Bayes (NB), K-Nearest Neighbor (KNN) and Decision Tree

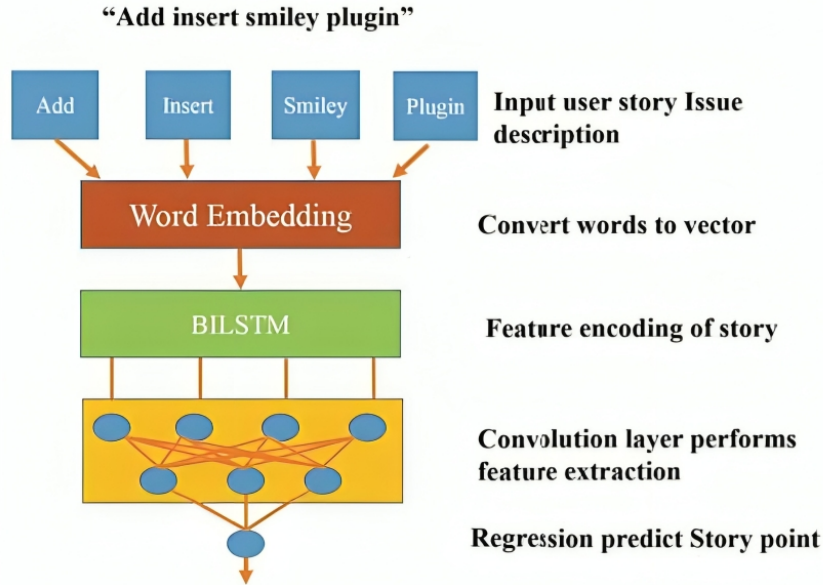


Figure 2.3: RNN-CNN model

(DT) models. They used accuracy and Mean Magnitude of Relative Error (MMRE) to evaluate the performance of each of the classifiers. The performance metrics for the classification models are given in figure 2.4.

Estimator	Correct estimates	Accuracy	MMRE
SVM	413	0.59	0.50
NB	309	0.44	0.85
KNN	249	0.36	0.70
DT	158	0.23	0.98

Figure 2.4: Performance metrics for the industrial project

According to the results, the SVM model achieved the highest accuracy and the best MMRE value out of all the classifiers for the industrial project. Therefore they have also checked the performance of the SVM model for the open source projects as well. The performance metrics for the SVM model for the open source projects are given in figure 2.5.

Furthermore, they confirmed the results from the study of Pekka Abrahamsson [11],

Project	Total est.	Correct est.	Accuracy	MMRE
APSTUD	228	143	0.63	0.61
XD	360	223	0.62	0.42
MESOS	387	223	0.58	0.39
NEXUS	421	341	0.81	0.16
TIMOB	634	380	0.6	0.6
SCR	699	413	0.59	0.5
MULE	805	416	0.52	1.07
DNN	858	669	0.78	0.16
TISTUD	1215	810	0.67	0.35
<i>mean</i>	<i>623</i>	<i>402</i>	<i>0.64</i>	<i>0.47</i>

Figure 2.5: Performance metrics for the SVM model in all projects

which suggested the length of the user story is a useful predictor for SP estimation. They suggested that in order to train the classifier to a level of accuracy, more than 200 issues should be used.

2.2.2 Developer features

A different approach was proposed than other studies by using developer-related features on a SVM model to estimate the SP [12]. The study combined developer-related features with the attributes extracted from 4142 user stories of 8 open-source projects. The developer-related features they considered are the current developer's workload, reputation, total work capacity (SP), total work capacity (number of issues) and number of developer comments. The SVM model was evaluated for the below 3 cases and the results are given in figure 2.6. Also, they used accuracy, SA and MAE to evaluate the performance of the model.

1. using only developer-related features (Dev)
2. using only features extracted from the text (Text)
3. using a combination of both the developer-related features and features extracted from the text (Text + Dev)

They have concluded that using only developer features is best suited for estimating the SP because it has the highest accuracy and MAE compared to the other 3 approaches.

	Accuracy					MAE						SA				
	Dev	Text	Text+Dev	Median	Random	Dev	Text	Text+Dev	Mean	Median	Random	Dev	Text	Text+Dev	Mean	Median
APSTUD	0.288	0.297	0.290	0.276	0.089	3.074	3.483	3.374	3.484	3.453	6.495	52.673	46.378	48.047	46.364	46.834
DNN	0.437	0.410	0.421	0.438	0.142	0.704	0.771	0.754	0.753	0.700	5.760	87.776	86.618	86.908	86.920	87.848
MESOS	0.338	0.362	0.359	0.255	0.130	1.375	1.234	1.215	1.254	1.177	4.823	71.493	74.423	74.817	74.006	75.606
MULE	0.336	0.256	0.306	0.241	0.120	2.540	2.979	2.814	2.600	2.594	5.551	54.234	46.334	49.308	53.159	53.272
NEXUS	0.591	0.553	0.581	0.462	0.120	0.495	0.526	0.511	0.373	0.366	5.350	90.738	90.177	90.457	93.020	93.152
TIMOB	0.344	0.283	0.305	0.325	0.116	2.154	2.672	2.550	2.278	2.264	5.759	62.590	53.605	55.719	60.445	60.683
TISTUD	0.412	0.350	0.344	0.412	0.139	1.753	2.035	2.064	1.800	1.744	5.262	66.691	61.328	60.778	65.792	66.853
XD	0.329	0.356	0.355	0.226	0.115	1.502	1.509	1.533	1.412	1.334	5.118	70.660	70.524	70.048	72.417	73.928
Average	0.384	0.358	0.370	0.329	0.121	1.700	1.901	1.852	1.744	1.704	5.515	69.179	65.531	66.421	68.371	69.100

Figure 2.6: Evaluation results for the 3 cases

2.2.3 Graph Based - Text Level GNN-StoryPoint Estimator

A novel classification-based Graph Neural Networks (GNN) method was proposed for estimating SP [13]. It is called TextLevelGNN. This study was focused on solving some of the issues found in the traditional GNN models such as challenges while testing the model online and excessive memory usage. They implemented a text-level graph that connects word nodes for every input text, rather than making an individual graph for the whole corpus. The dataset used consists of 23,313 user stories from 16 open-source projects and they have divided 80% of the dataset for training the remaining for validation. They classified the SP values into 4 namely small (1–5), medium (6–15), large (16–40) and huge (≥ 40). Their method involves creating a graph that shows the relationship between each word and its neighboring terms by combining the title and description of the user story. Then the last step involves training the GNN model to predict the SP values. The model architecture is given in figure 2.7.

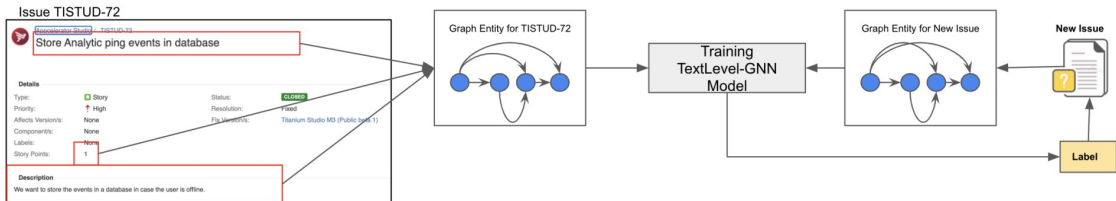


Figure 2.7: Architecture of TextGraph GNN model

It was further stated by the authors that GNN have better representations than traditional vectorizing methods because of their capacity to learn data using graph data structures. When it comes to the evaluation, they have done testing by

comparing the random forest classifier with the TF-IDF and the TextLevelGNN model. The results are given in the figure 2.8.

No	Metric Software	Classify Acc		Reg MAE	
		TFIDF RF	GNN	TFIDF RFR	GNN
1	appceleratorstudio	78.08%	77.78%	1.66	2.67
2	aptanastudio	46.99%	41.41%	3.50	5.16
3	bamboo	100.00%	100.00%	0.88	0.00
4	clover	83.12%	81.25%	4.04	1.22
5	datamanagement	61.67%	56.36%	6.90	13.40
6	duracloud	99.25%	99.22%	0.91	0.07
7	jirasoftware	91.55%	89.06%	1.84	1.31
8	mesos	97.02%	96.88%	1.30	0.28
9	moodle	59.83%	56.25%	11.17	25.16
10	mule	74.16%	66.41%	2.46	4.03
11	mulestudio	54.42%	52.34%	3.92	5.84
12	springxd	87.96%	88.07%	1.90	1.08
13	talenddataquality	80.14%	83.20%	3.99	1.50
14	talendesb	98.28%	98.44%	0.87	0.14
15	titanium	75.61%	74.55%	2.75	2.30
16	usergrid	95.88%	96.88%	1.25	0.03
	Average	80.25%	78.63%	3.08	3.09

Figure 2.8: TextLevelGNN model evaluation with random forest

However, the accuracy came with the cost of training time. The authors observed that it took longer time to train the TextLevelGNN model than the Random Forest model. It took 25 minutes for the training step of the TextLevelGNN model while the Random Forest model took 4 minutes for the training step.

2.3 Clustering Algorithms

2.3.1 K Means and K-medoids clustering

An approach was proposed that used k-means clustering [14]. The dataset used is from an industrial project which consists of 98 user stories. Preprocessing, clustering and validation are the 3 phases of the proposed model. The preprocessing step consists of removing stop-words and tokenization. The clustering step is used to determine the required clusters by implementing the k-means algorithm to both TF-IDF vectorizers with cosine similarity and count vectorizer. The validation

step consists of finding the cluster out of all the obtained results. The steps for the proposed approach are given in the figure 2.9.

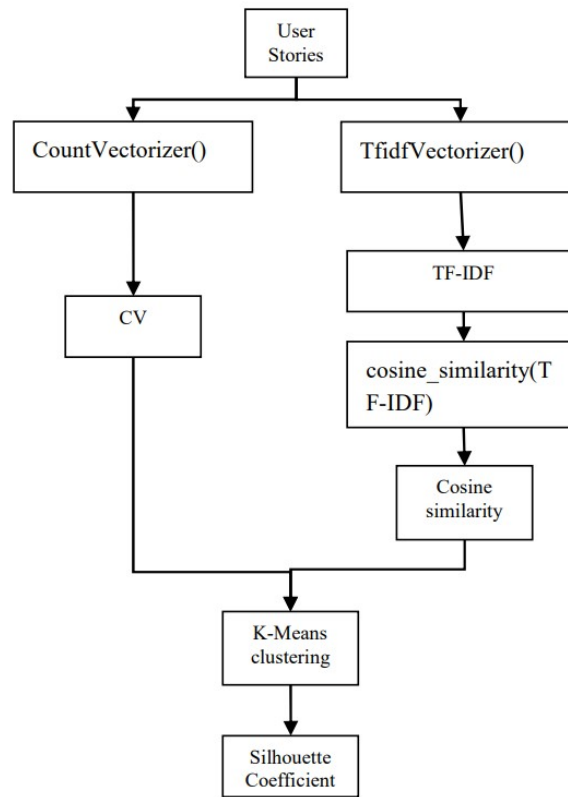


Figure 2.9: Steps for proposed study

Furthermore, To evaluate how well the data points are grouped in a cluster, they used the silhouette coefficient value. Also, the k-means algorithm they used consists of 4 steps.

1. Choose k data points at random to be used as centroids
2. It is calculated how similar the data points are to each cluster centroid
3. K centroids are modified by recently allocated data points.
4. Repeat (a) and (b) steps until all data points have been covered.

2.3.2 Hierarchical clustering

A novel clustering-based approach was proposed named LHC-SE to estimate SP in agile development [15]. The model estimates the SP value of an issue based on previous similar problems using agglomerative hierarchical clustering and topic space of issue descriptions provided by LDA. The theory is that grouping com-

parable data points together can decrease variation and improve the precision of the model based on those data points. The dataset used for this study consists of 31,960 issues in 26 open-source projects.

2.4 GPT2SP

The Generative Pre-trained Transformer (GPT)-2 model was developed by OpenAI which is a transformer-based language model that is used for a wide range of Natural Language Processing (NLP) tasks including text translation and summarization. There are different GPT-2 variants such as small, medium, large and extra large. A novel approach was proposed using the GPT model [2]. The model that they have developed is called GPT2SP and it utilizes the GPT-2 small variant which consists of 177 million parameters. The model architecture is given in the figure 2.10.

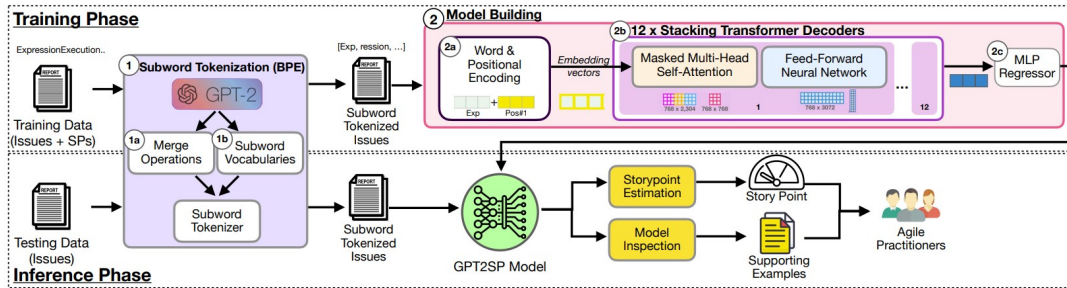


Figure 2.10: The architecture of the GPT2SP model

Their dataset consists of 23,313 user stories from 16 open-source projects. The approach authors proposed provides interpretability and enables the transferability of GPT2SP across projects. For interpretability, they have proposed a proof-of-concept. The authors have further mentioned that the GPT2SP model achieved 1.16 for the MAE which results in within-project evaluations to outperform the baselines by 34% to 57% and in cross-project evaluations to outperform the baselines by 39% to 49%. They also mentioned that the Deep-SE model was significantly improved by the GPT2SP model.

2.5 Explainable AI

XAI is divided into 2 parts namely explainability and interpretability. Explainability is the process of comprehending and clarifying a ML model's underlying workings. Interpretability refers to the ability to comprehend and provide an explanation for the predictions or decisions made by a ML model [16]. XAI has emerged as a novel area in research in the DL context [17]. They further stated that today's Deep Neural Network (DNN)s provide outputs that can't be explained without entirely new explanation methods and all the currently available DNN architectures such as RNN, CNN and LSTM are considered as black-box models because the observer cannot understand or interpret the underlying decision methods. Furthermore, a machine learning model's explainability is typically inversely correlated with accuracy. This means the higher the accuracy of the predictions, the less explainable the model is. This is illustrated in the figure 2.11 where decision trees have better explainability while having comparatively lower prediction accuracy however DL models have better prediction accuracy while having comparatively lower explainability.

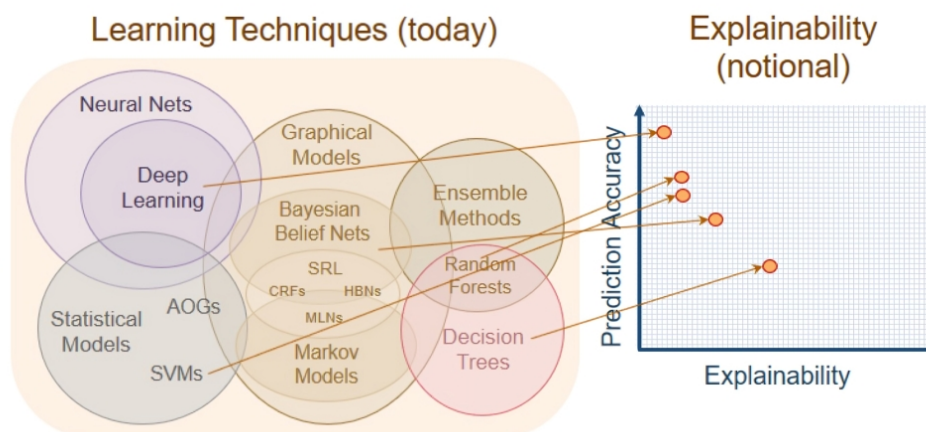


Figure 2.11: Machine learning model's explainability and accuracy

2.5.1 Approaches of XAI in other research areas

In a previous literature on the research area of Automated Short Answer Grading, the authors have done a survey on 71 individuals including professors, lecturers and

teachers to check if it is essential that they comprehend how an AI arrives at its predicted score. The results of that survey are given in the figure 2.12. According to the results, the majority of the individuals confirm that it is essential for them to comprehend how an AI determines its predicted score and the individuals feel more confident on the models if the models explain their decisions [18].

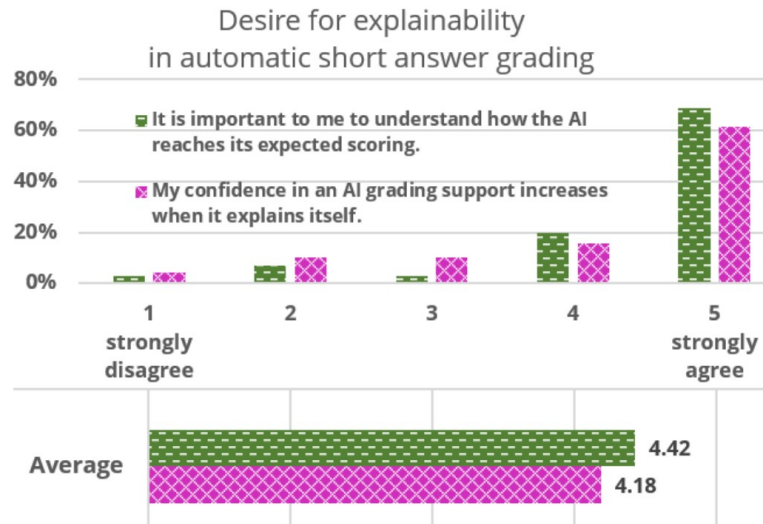


Figure 2.12: Survey results for the need of XAI

Another previous literature also mentioned several approaches of applying XAI in the research area of Automated Short Answer Grading. One of them is to output the predicted score with a confidence score and a similar answer to the given input. They defined the confidence score as the interpretable and observable level of prediction certainty [18]. The output of this XAI approach is given in figure 2.13.

The next approach gives the predicted score with the highlighted words in the input answer. The words are highlighted depending on their relevance to the contribution to the score. The highlighted opacity also depends on the relevance. That means if the opacity is high then that word is highly relevant to the score and if the opacity is low then that word is not that relevant to the score [18]. The output of this XAI approach is given in the figure 2.14.

An approach was proposed for XAI by using a separate dataset [19]. In this dataset

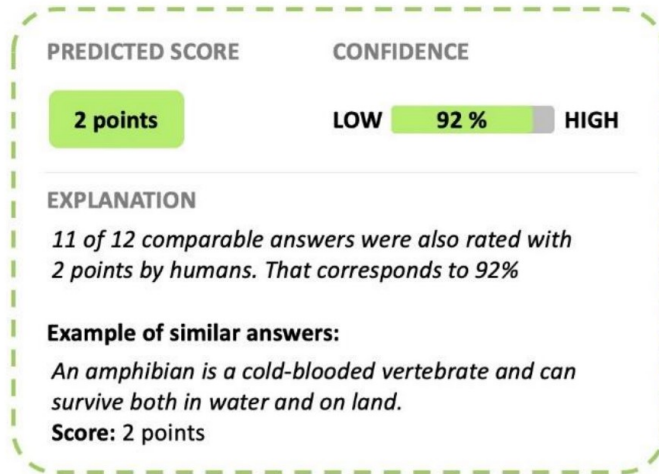


Figure 2.13: XAI approach: Predicted output score with a confidence score and a similar answer

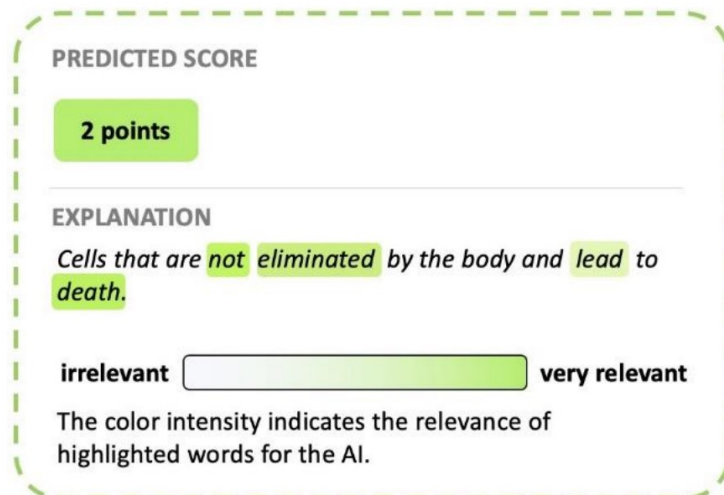


Figure 2.14: XAI approach: Predicted output score with highlighted words in the input answer

apart from the general features such as question, answer and the relevant score, there is also an additional feature called the elaborated feedback that explains the relevant score. The authors have constructed this feature manually by using domain experts.

An analysis and a review were conducted on various XAI methods such as LIME, SHAP, Anchors, etc [20]. According to the authors, the most thorough and widely used techniques for visualizing feature importance and interactions in the literature

are LIME and SHAP. Furthermore LIME and SHAP methods are independent of models and also they have been shown to work with any kind of data. LIME gives information about the relative contributions of each feature to the final result of a ML model for a certain prediction [16].

Several tests were conducted by a study to find the best XAI method for interpretation [21]. They selected LIME, SHAP and Class Activation Mapping (CAM) as the XAI methods. The authors further state that those 3 XAI methods are widely used. They have proposed 2 metrics to evaluate XAI methods namely Determining the Highest-Impacted Segments (DHIS) which determine the regions that have the greatest influence on the predictions and Intersection Over Union (IOU) which takes a value from 0 to 1 where the most accurate XAI approach is the one with the largest IOU value. The results of the evaluation are given in the figure 2.15. By selecting the 2 highest-value areas, LIME outperforms the other two XAI approaches, according to results on the DHIS method. Also when comparing the IOU values, out of the 3 XAI methods LIME has the highest value. Therefore, by analysing the results the authors mentioned that the explanations provided by LIME and SHAP are more precise and human-like.

XAI methods	Examples	Regions	IOU	Running time (s)
LIME	200	26, 33	0.341	49.967
	500	12, 21	0.434	124.822
	1000	12, 20	0.566	235.761
SHAP	200	12, 13	0.368	44.626
	500	12, 13	0.368	94.331
	1000	12, 11	0.415	167.524
	Threshold			
CAM	20%	12, 20	0.459	0.012
	50%	12, 20	0.658	0.016
	80%	12, 20	0.553	0.007

Figure 2.15: Results of LIME, SHAP and CAM on DHIS and IOU

Chapter 3 - Design

3.1 Research Approach and Methodology

This study explored traditional approaches, transformer models [22] and XAI techniques to find an approach to estimate SP more accurately and consistently utilizing the existing theoretical foundation. Therefore this research falls under the category of deductive research using an experimental research approach in Saunders' Research Onion Framework [23]. The main steps of the approach are given in figure 3.1. The approach starts with an existing theory and then using that theory the hypotheses are formulated. Then comes the data collection and analysis. The final step is to decide whether the hypothesis is rejected or not. The hypotheses of this study are mentioned below.

1. The null hypothesis (H0): Using the selected transformer models we cannot get better accuracy than the state-of-the-art traditional models and using XAI methods we cannot generate justifications for each estimation.
2. Alternative Hypotheses (H1): Using the selected transformer models we can get better accuracy than the state-of-the-art traditional models and using XAI methods we can generate justifications for each estimation.

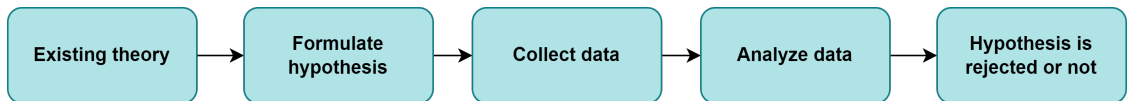


Figure 3.1: Flow of deductive Research

This study used the design science methodology to solve the limitations of existing story point estimating approaches and provide an accurate and reliable approach for this area of research. This rigorous methodology, shown in figure 3.2, ensured a novel estimation approach was developed that successfully contributed to the area of research [24].

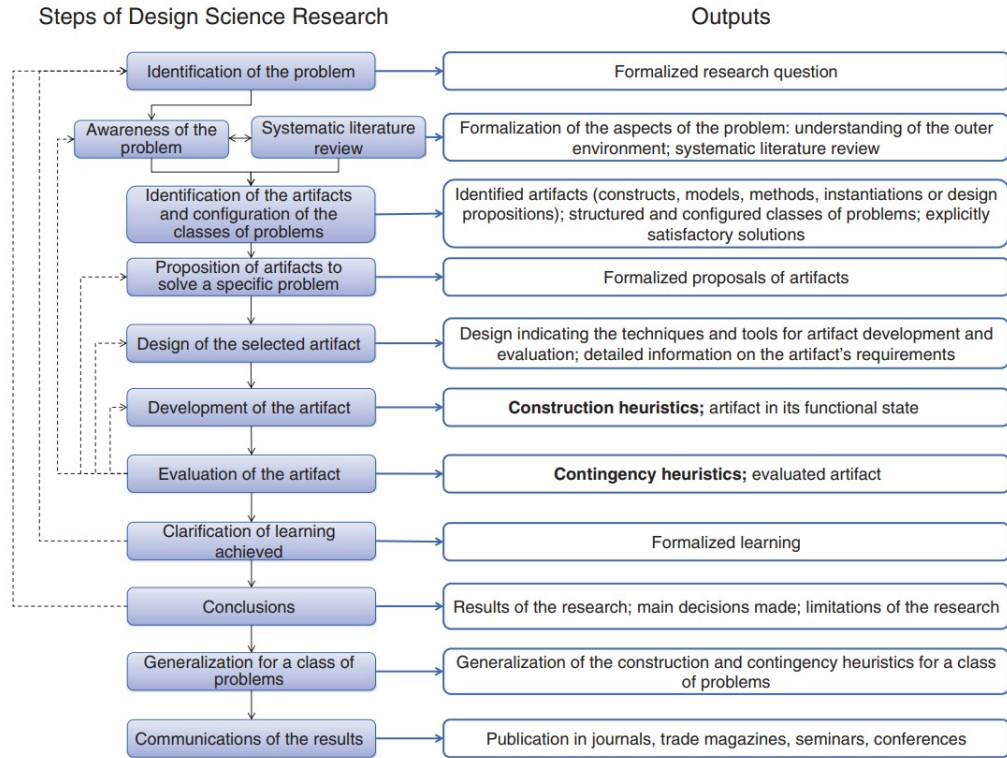


Figure 3.2: Design science research steps

3.2 High Level Design

This study primarily consists of two phases. The first phase involves the development of SP prediction model. The second phase involves the integration of XAI methods to generate explanations of the model's predictions. A high-level illustration of the steps followed to build the traditional approaches and transformer models are given in figure 3.3 which includes all the steps of data preprocessing. A comprehensive elaboration of each phase is provided in the following sections.

3.2.1 Data Collection

The model's accuracy depends on the dataset's quality and size. There are few publicly available datasets for this domain. The Choetkiertikul dataset is commonly used in most of the previous literature which consists of 16 projects with a total of 23,313 user stories with the features title, description and SP [25]. To address the first research question, the construction of a richer dataset was needed. For that, I used the TAWOS (Tawosi Agile Web-based Open-Source) dataset which was

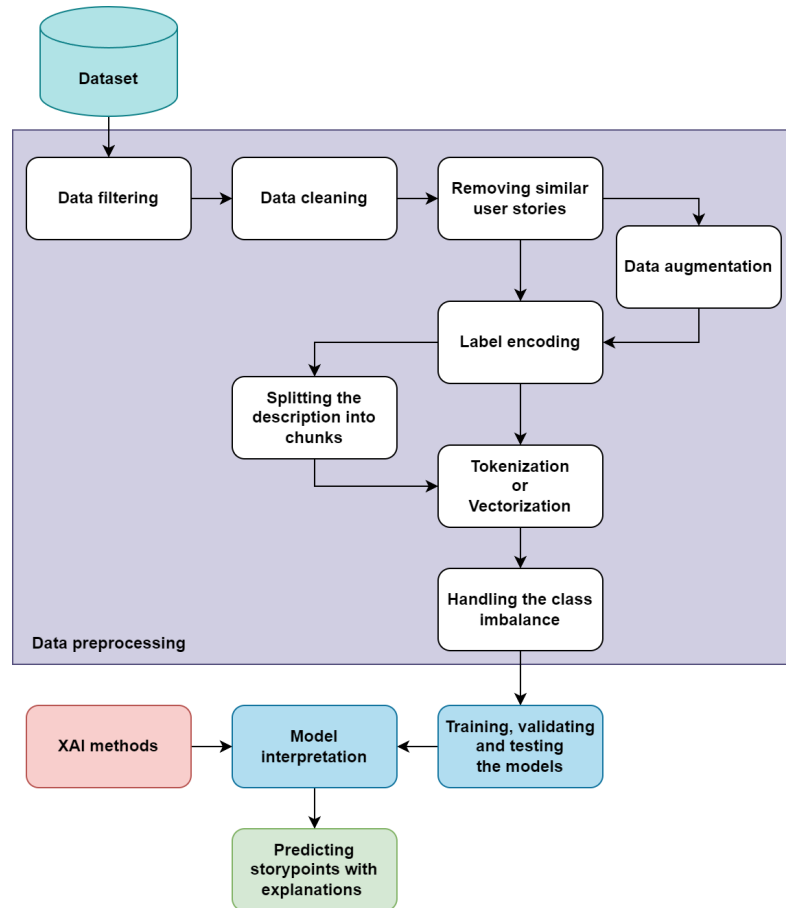


Figure 3.3: High-Level Research Design

provided by Vali Tawosi. The TAWOS dataset is a comprehensive dataset from 44 open-source Agile projects that includes a variety of data from over 500,000 issues [26]. Also, it is better than the Choetkiertikul dataset in terms of size, quality and features. Furthermore, the 16 projects in the Choetkiertikul dataset are also included in the TAWOS dataset. Based on the features, the TAWOS dataset was used to construct 2 types of datasets for this study, which are mentioned below.

1. Dataset A - with the features issue key, title, description and SP
2. Dataset B - with the features issue key, title, description and SP, type of issue and priority of the issue

The overview of both datasets is given in figure 3.4 and figure 3.5.

The TAWOS dataset consists of 44 open-source projects. However, for this study only 39 projects were useful. The data distributed of all the 39 projects including the project names and their respective number of data items are given in the figure 3.6.

	issuekey		title	description	storypoint
0	XD-3768	"How do I make a job restartable in spring xd"	"The jobs that appear under Executions section...		1
1	XD-3767	"admin config timezone command does not work"	"Working with Spring-XD version 1.3.2.RELEASE ..."		1
2	XD-3766	"Module Upload command not pushing jar to all ..."	"My project 7 node cluster and in that 2 node ..."		10
3	XD-3765	"Fix stream failover "	"See https://github.com/spring-projects/spring... "		8
4	XD-3764	"SpringXD Job is still executing even after fo..."	"I'm trying to run a Job on SpringXD and the j..."		5
...
65421	SERVER-20056	"Log a startup warning if wiredTigerCacheSizeG..."	"Currently you can set wiredTigerCacheSizeGB t..."		3
65422	SERVER-19895	"resmoke failures should self-document"	"When resmoke fails, it should print out steps..."		2
65423	SERVER-18840	"resmoke should indicate status of test in abb..."	"When resmoke is running a batch of tests and ..."		1
65425	SERVER-16612	"Implicitly zeroed files in WiredTiger"	"There is a problem with the implicit zeroing ..."		5
65426	SERVER-3748	"Add init scripts for mongos and configsvr to ..."	"Although it's not possible to automatically..."		13

61842 rows × 4 columns

Figure 3.4: The overview of dataset A

	issuekey		title	description	type	priority	storypoint
0	XD-3768	"How do I make a job restartable in spring xd"	"The jobs that appear under Executions section..."		Bug	Major	1
1	XD-3767	"admin config timezone command does not work"	"Working with Spring-XD version 1.3.2.RELEASE ..."		Bug	Trivial	1
2	XD-3766	"Module Upload command not pushing jar to all ..."	"My project 7 node cluster and in that 2 node ..."		Bug	Major	10
3	XD-3765	"Fix stream failover "	"See https://github.com/spring-projects/spring... "		Story	Minor	8
4	XD-3764	"SpringXD Job is still executing even after fo..."	"I'm trying to run a Job on SpringXD and the j..."		Bug	Major	5
...
65421	SERVER-20056	"Log a startup warning if wiredTigerCacheSizeG..."	"Currently you can set wiredTigerCacheSizeGB t..."		Improvement	Minor - P4	3
65422	SERVER-19895	"resmoke failures should self-document"	"When resmoke fails, it should print out steps..."		Improvement	Major - P3	2
65423	SERVER-18840	"resmoke should indicate status of test in abb..."	"When resmoke is running a batch of tests and ..."		Improvement	Major - P3	1
65425	SERVER-16612	"Implicitly zeroed files in WiredTiger"	"There is a problem with the implicit zeroing ..."		Task	Major - P3	5
65426	SERVER-3748	"Add init scripts for mongos and configsvr to ..."	"Although it's not possible to automatically..."		Improvement	Major - P3	13

61842 rows × 6 columns

Figure 3.5: The overview of dataset B

Also, the SP distribution of the TAWOS dataset is given in figure 3.7. Furthermore, the SP distribution of each of the 39 projects is given in the figures 3.8, 3.9, 3.10, 3.11, 3.12 and 3.13.

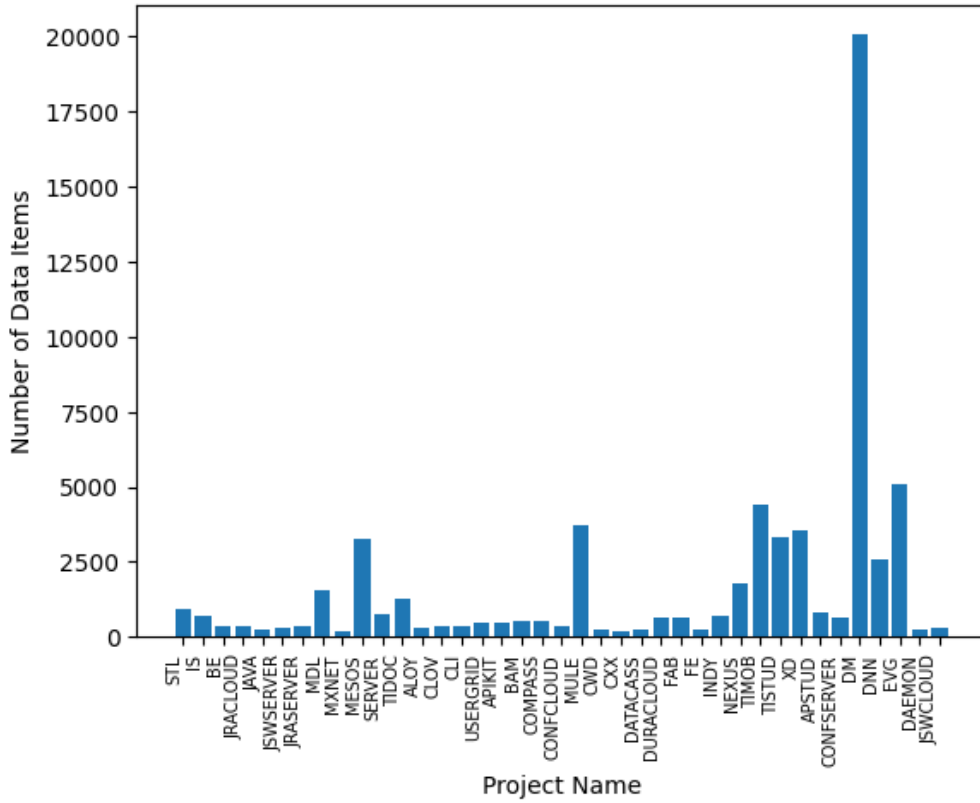


Figure 3.6: Data distributed of the TAWOS dataset

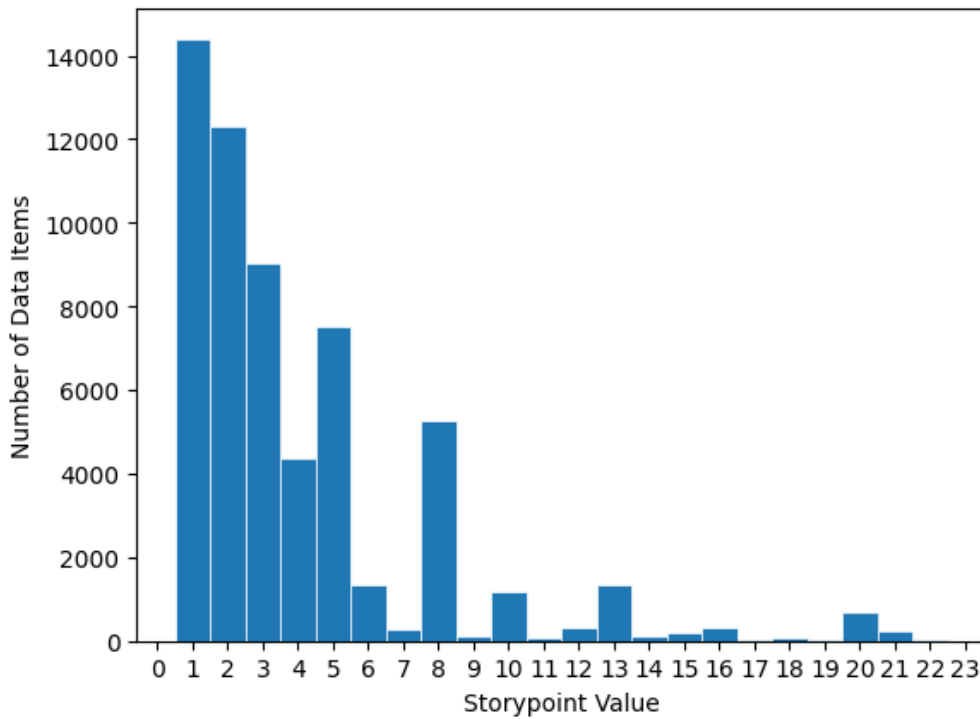


Figure 3.7: SP distributed of the TAWOS dataset

3.2.2 Pre-processing

After analysing the previous literature, it was evident that planning poker was the most widely used approach for SP estimation. Then the initial decision of the data preprocessing was to decide whether to use the Fibonacci sequence (1, 2, 3, 5, 8, 13, 21) or the numerical sequence (1, 2, 3, 4, 5, 6, 7). The Fibonacci sequence was adapted to this study since it was commonly used in this area of research. Based on the initial analysis of the SP distribution of the TAWOS dataset which is given in figure 3.7, it was observed that there aren't many user stories for the SP above 8. Also, a previous study has mentioned that to get a model with adequate accuracy, the classifier must be trained on more than 200 problems per category [10]. Therefore the first step of data preprocessing is to filter out the user stories which has a SP value of greater than 8. Also since this study adapts the Fibonacci sequence, the user stories of SP value 4, 6 and 7 were filtered out.

Then the second step of data preprocessing is to clean the dataset to increase the accuracy because clean data enhances the capability of the model to identify underlying patterns and relationships, resulting in better predictions and results, and enhanced efficiency. After all, the model can train more quickly and make better use of its resources by eliminating data that is inaccurate or useless. Also, less biased and more reliable results can be obtained by cleaning the dataset because unclean data may have biases and inconsistencies that the model may pick up and propagate.

The third step of data preprocessing is to reduce the noise of the dataset by filtering out similar user stories with different SP values. This is done because similar descriptions with different SP values show estimation inconsistencies, which can add noise to the dataset. By eliminating these noises we can achieve a more reliable dataset resulting in analysis and conclusions that are more reliable. Also, we can get a better understanding of what factors are impacting the SP estimation by focusing on the user stories that differ in effort or complexity by eliminating similar ones.

The fourth step in data preprocessing involves encoding the SP values which is essential as they represent categorical data. Since many ML models work with numerical data, this conversion is crucial for the model's comprehension. By encoding SP, we convert them into a format that ML models can comprehend.

The next step involves creating a representation using either tokenization or vectorization depending on the traditional approach or the transformer model. Some ML models have a maximum sequence length that can be tokenized such as the BERT model has a maximum sequence length of 512 tokens. Therefore to prevent information loss depending on the model, the description was split into chunks and then tokenization was done.

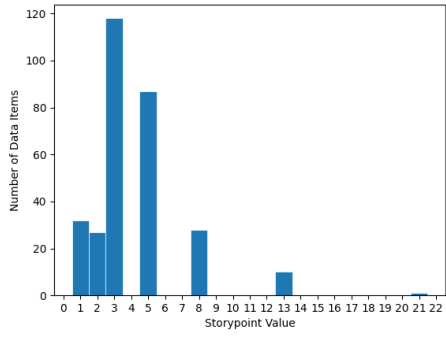
In a classification problem, a class imbalance arises when one class has substantially fewer samples than another [27]. This can result in inaccurate predictions and biased models. As a result of class imbalance, the models might prioritize the dominant class which would negatively impact the minority class's performance. One approach for addressing class imbalance is the use of class weights that are used in model training to provide lower weights to majority classes while giving greater weights to minority classes. Furthermore, according to the SP distribution of the TAWOS dataset given in figure 3.7, there is a class imbalance across all 5 classes. Therefore class weighting was used to handle this issue. With the objective of overcoming the class imbalance in the dataset, this study looked into data augmentation. Although class weighting is a standard strategy to address class imbalance, data augmentation was also tested because it can increase the size and variety of the minority class which is SP value 5 and 8, without creating inaccurate biases. The goal of this method is to enhance the dataset's quality so that the models can learn from and generalize to new cases more effectively, especially in the underrepresented class.

3.2.3 Design and Implementation of Models for SP Estimation

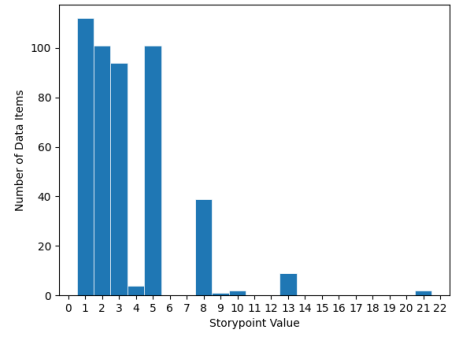
An analysis of previous literature revealed that there were multiple ML models created for SP estimation. I analyzed how well those ML models performed and after a thorough consideration of the previous literature, Random Forest, SVM, LSTM and BiLSTM were selected as the traditional approaches. The BERT, DistilBERT and RoBERTa models were selected as the transformer model because of several compelling reasons. One of the reasons why BERT was selected is its state-of-the-art performance. BERT has continuously shown outstanding performance on a range of NLP tasks such as question-answering, text summarization and sentiment analysis [28]. Although it hasn't been used specifically for the domain of story point estimation, its effectiveness in evaluating textual data shows a real potential. BERT also have the capability to comprehend complex text. This is needed because user stories can contain complex textual data. Unlike other transformer models, BERT can analyze text bi-directionally which allows it to completely understand the relationships and context between words. This allows it to capture the complex details of user stories and potentially improve the accuracy of SP predictions. Also, BERT can generate contextual embeddings which is essential to capture how a word's meaning varies based on the surrounding content. This type of dynamic comprehension of word meaning was significant in my domain. Furthermore, BERT has pre-trained models such as BERT base or large and it has a variation of models such as RoBERTa and DistilBERT. Also, it has the capability to fine-tune the pre-trained models for a specific task such as story point estimation [29]. Lastly is the BERT tokenizer which plays a significant role when it comes to textual data. When dealing with unknown words the BERT tokenizer efficiently divides them into smaller, more recognizable units. This produces understandable tokens for the model, enabling it to derive meaning from even highly complicated terms that it hasn't come across before. The figure 3.14 illustrates how the BERT tokenizer handles the word 'machinelearning'. Various experiments were done with the traditional and transformer models. The details and results of those experiments are given in the next sections.

3.2.4 Implementation of XAI methods

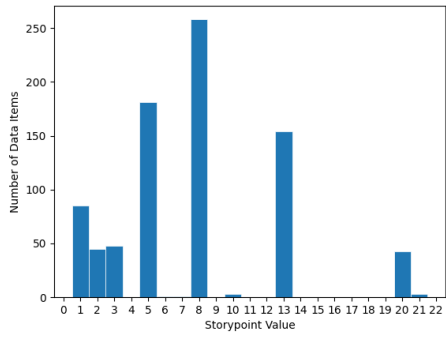
An analysis of previous literature revealed that various XAI approaches have been applied to diverse domains including automated short answer grading research areas. However, the use of XAI in the field of story point estimation is still relatively new. By analysing the previous literature it was evident that LIME was widely used because of its significant efficiency. Therefore, in this study LIME was primarily employed for generating explanations for both traditional approaches and transformer models. However, other than LIME, 2 other approaches namely SHAP and transformers-interpret were used for the transformer model.



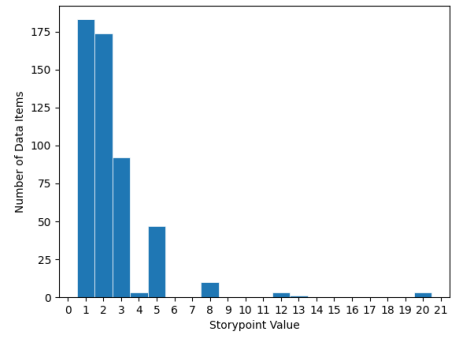
(a) ALOY



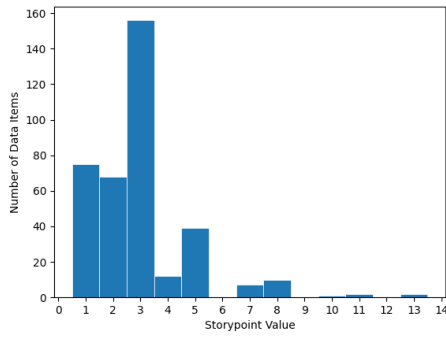
(b) APIKIT



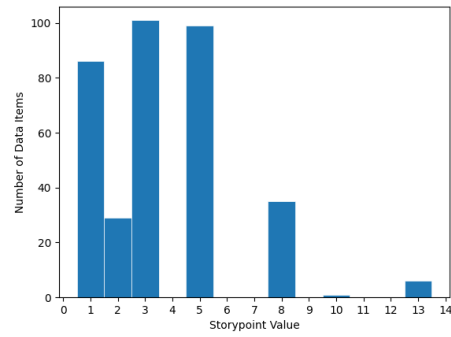
(c) APSTUD



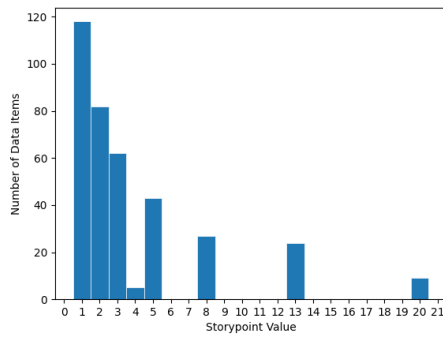
(d) BAM



(e) BE

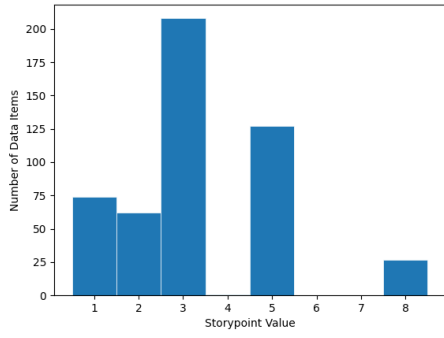


(f) CLI

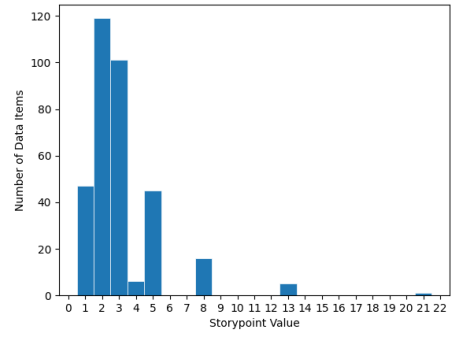


(g) CLOV

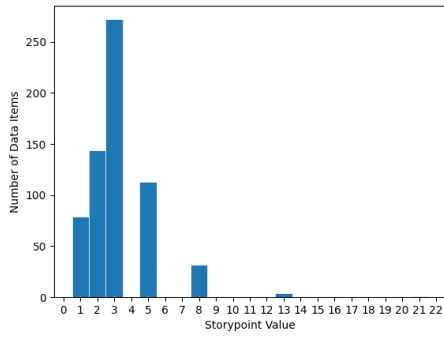
Figure 3.8: SP distribution of the 39 projects (first set)



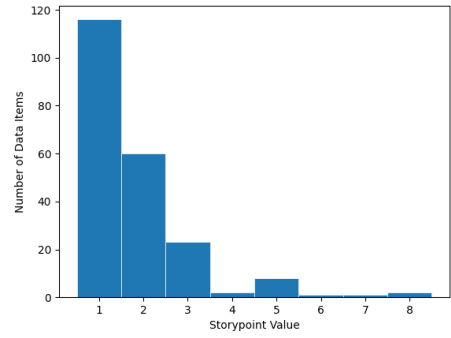
(a) COMPASS



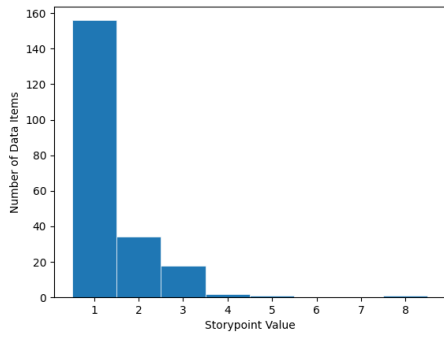
(b) CONF CLOUD



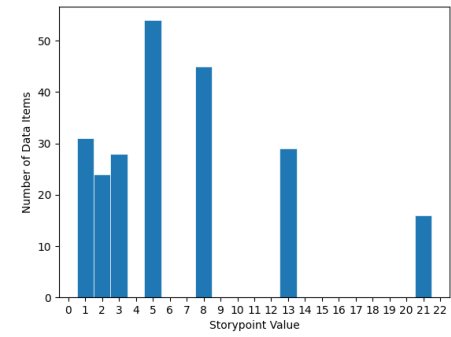
(c) CONF SERVER



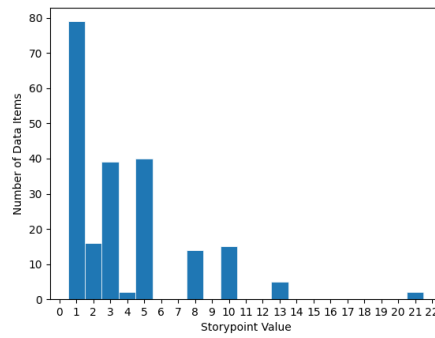
(d) CWD



(e) CXX

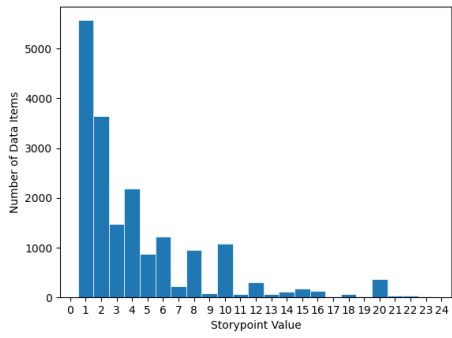


(f) DAEMON

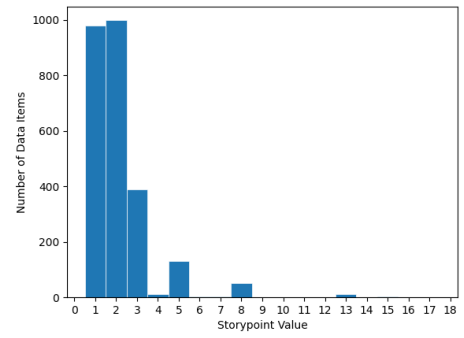


(g) DATA CASS

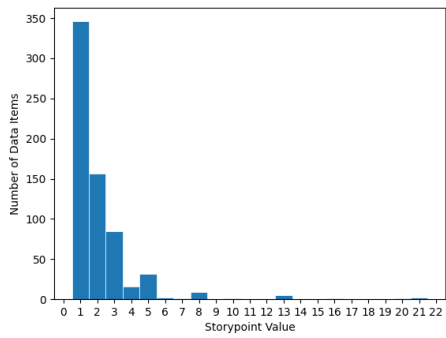
Figure 3.9: SP distribution of the 39 projects (second set)



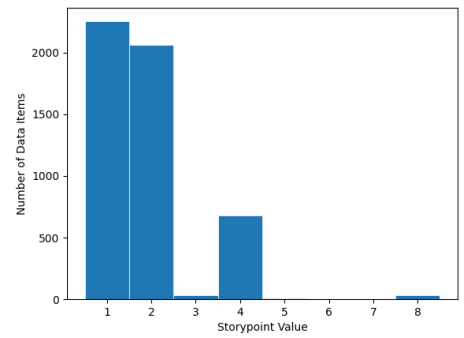
(a) DM



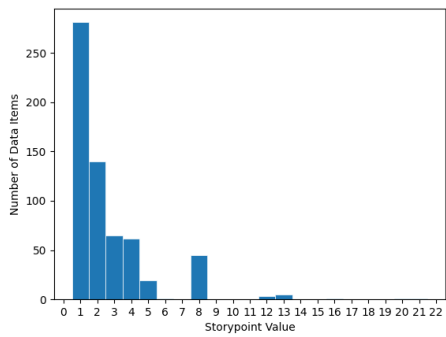
(b) DNN



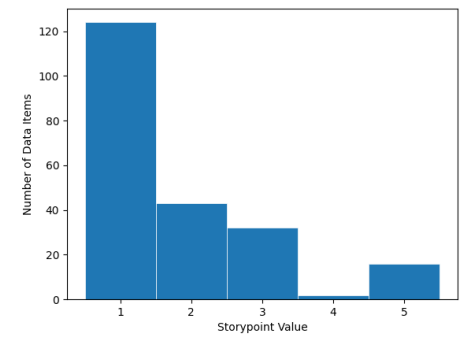
(c) DURACLOUD



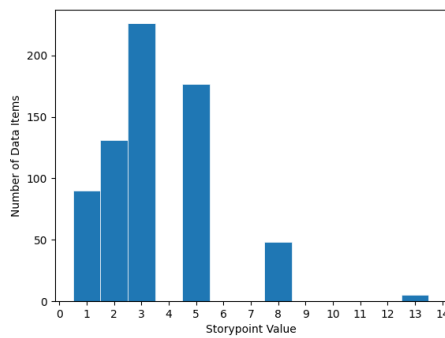
(d) EVG



(e) FAB

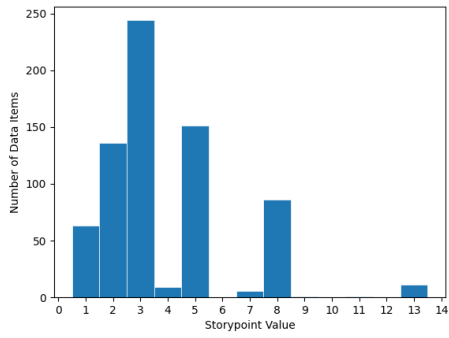


(f) FE

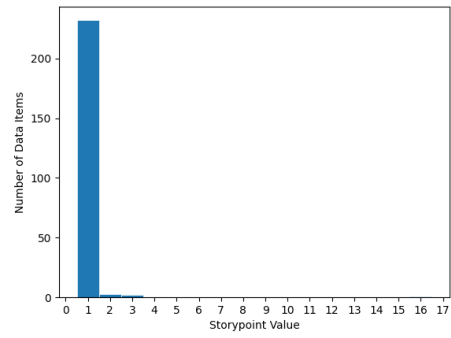


(g) INDY

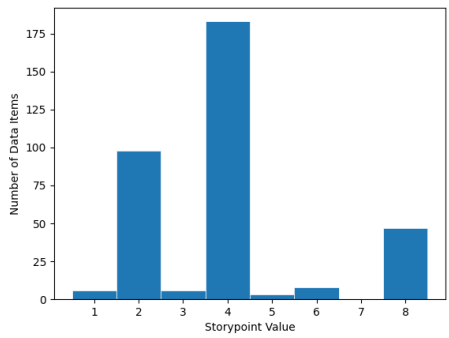
Figure 3.10: SP distribution of the 39 projects (third set)



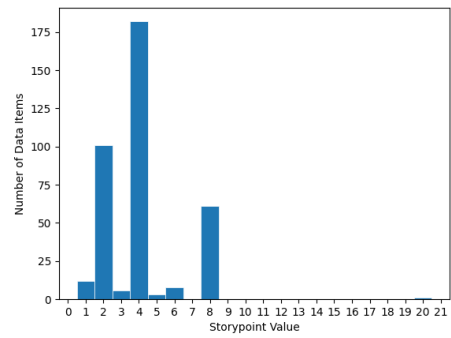
(a) IS



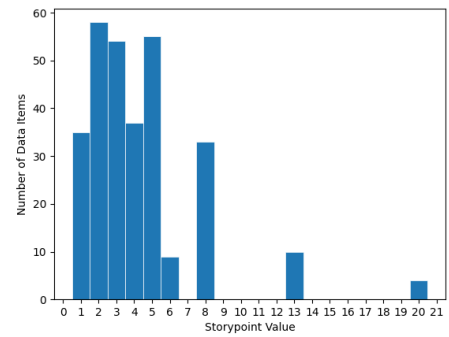
(b) JAVA



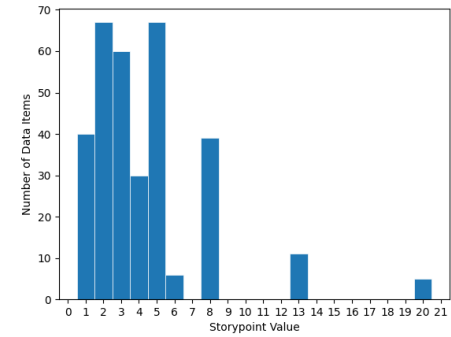
(c) JRACLOUD



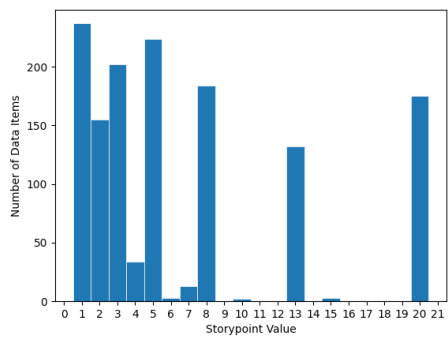
(d) JRASERVER



(e) JSWCLOUD

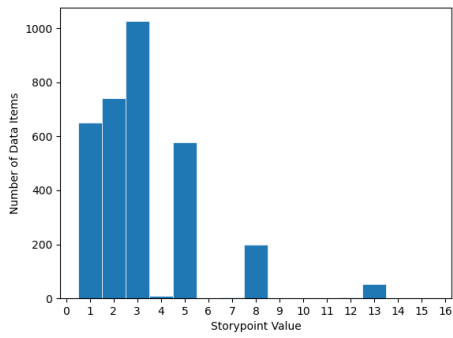


(f) JSWSERVER

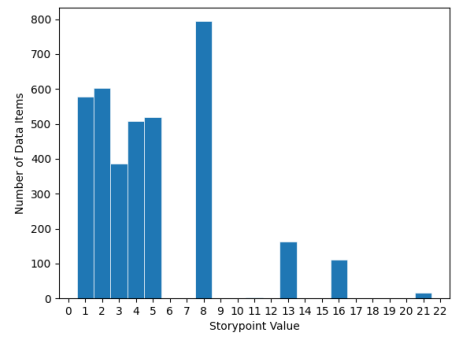


(g) MDL

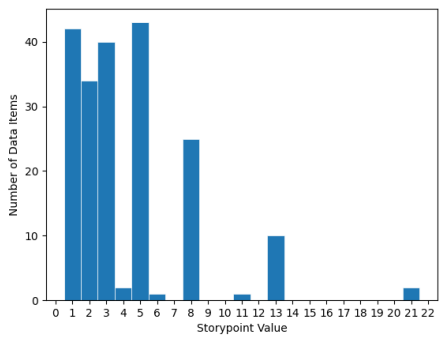
Figure 3.11: SP distribution of the 39 projects (fourth set)



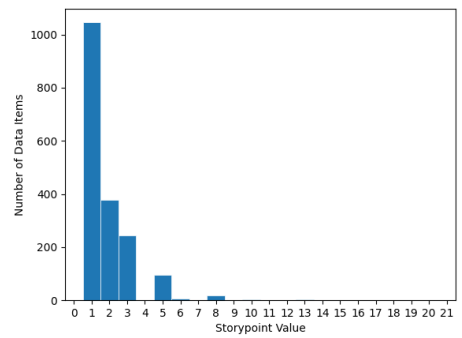
(a) MESOS



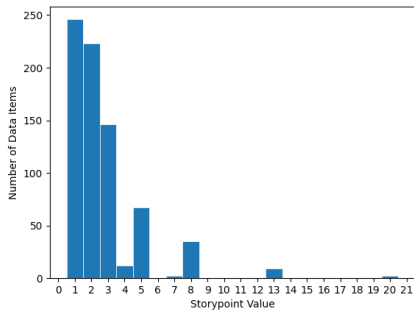
(b) MULE



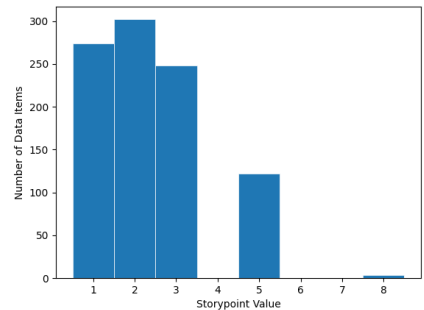
(c) MXNET



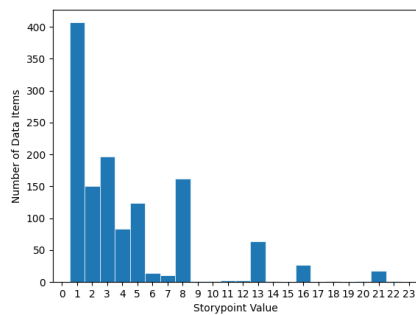
(d) NEXUS



(e) SERVER

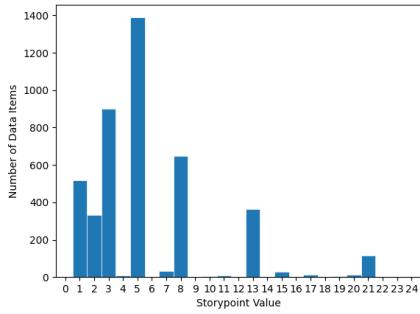


(f) STL

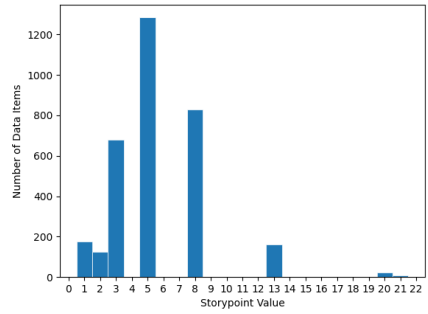


(g) TIDOC

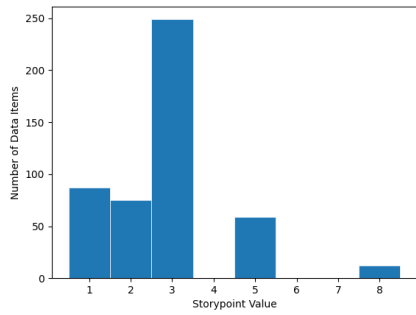
Figure 3.12: SP distribution of the 39 projects (fifth set)



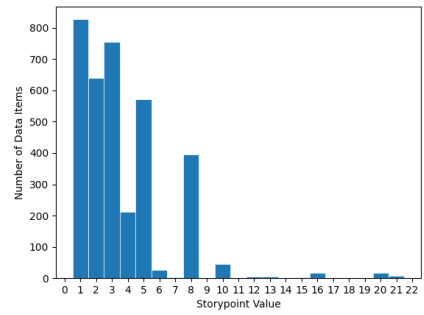
(a) TIMOB



(b) TISTUD



(c) USERGRID



(d) XD

Figure 3.13: SP distribution of the 39 projects (sixth set)

```

▶ tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")
  tokens = tokenizer.tokenize("machinelearning")
  print(tokens)

['machine', '##lea', '##rn', '##ing']

```

Figure 3.14: How BERT tokenizer works for unknown words

Chapter 4 - Implementation

4.1 Data Collection

The TAWOS dataset was publically available and it was provided as an Structured Query Language (SQL) file which served as the dataset for this study. I effectively extracted the necessary data needed for this study by utilizing the command line interface. The TAWOS dataset consists of a set of tables which is illustrated in figure 4.1, that relate to several areas of user stories in software development.

```
mysql> SHOW TABLES;
+-----+
| Tables_in_rdb |
+-----+
| affected_version |
| change_log      |
| comment         |
| component       |
| fix_version     |
| issue           |
| issue_component |
| issue_link      |
| project         |
| repository      |
| sprint          |
| user            |
| version         |
+-----+
```

Figure 4.1: Tables in the TAWOS dataset

Furthermore, out of all the tables, only the "issue" table contains the data that is essential for this study. The structure of the issue table is illustrated in the figure 4.2 which contains the essential data such as issue key, title, description, SP, etc. Also, the 2 datasets namely Dataset A and Dataset B were successfully created by processing the "issue" table. The necessary SQL queries are given in the figure 4.3.

4.2 Pre-processing

4.2.1 Data Filtering

The inclusion and deletion of data items according to SP values were thoroughly considered during the data filtration phase. In order to keep the analysis focused

```
mysql> DESCRIBE issue;
```

Field	Type	Null	Key	Default	Extra
ID	int	NO	PRI	NULL	auto_increment
Jira_ID	int	YES		NULL	
Issue_Key	varchar(512)	YES		NULL	
URL	varchar(512)	YES		NULL	
Title	varchar(512)	YES		NULL	
Description	mediumtext	YES		NULL	
Description_Text	mediumtext	YES		NULL	
Description_Code	mediumtext	YES		NULL	
Type	varchar(128)	YES		NULL	
Priority	varchar(128)	YES		NULL	
Status	varchar(128)	YES		NULL	
Resolution	varchar(128)	YES		NULL	
Creation_Date	datetime	YES		NULL	
Estimation_Date	datetime	YES		NULL	
Resolution_Date	datetime	YES		NULL	
Last_Updated	datetime	YES		NULL	
Story_Point	double	YES		NULL	
Timespent	double	YES		NULL	
In_Progress_Minutes	double	YES		NULL	
Total_Effort_Minutes	double	YES		NULL	
Resolution_Time_Minutes	double	YES		NULL	
Title_Changed_After_Estimation	tinyint	YES		NULL	
Description_Changed_After_Estimation	tinyint	YES		NULL	
Story_Point_Changed_After_Estimation	tinyint	YES		NULL	
Pull_Request_URL	varchar(512)	YES		NULL	
Creator_ID	int	YES	MUL	NULL	
Reporter_ID	int	YES	MUL	NULL	
Assignee_ID	int	YES	MUL	NULL	
Project_ID	int	YES	MUL	NULL	
Sprint_ID	int	YES	MUL	NULL	

Figure 4.2: The structure of the issue table

```
mysql> SELECT issue_key, title, description, story_point FROM issue WHERE story_point IS NOT null INTO OUTFILE 'C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\datasetA.csv' FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n';
Query OK, 65427 rows affected (7.90 sec)

mysql> SELECT issue_key, title, description, type, priority, story_point FROM issue WHERE story_point IS NOT null INTO OUTFILE 'C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\datasetB.csv' FIELDS TERMINATED BY ',' LINES TERMINATED BY '\n';
Query OK, 65427 rows affected (7.81 sec)
```

Figure 4.3: Construction of Dataset A and Dataset B

on the typical range found in the area of study, the SP values greater than 8 were considered outliers and were removed from the analysis. Furthermore, since SP values 4, 6 and 7 deviate from the Fibonacci sequence used in this study, they were also eliminated. Assigning effort levels between consecutive Fibonacci values (eg - between 3 and 5) increases ambiguity and makes it more difficult to analyze the data accurately. Eliminating these data items results in an overall smaller sample size. However, it also guarantees that the selected estimation approach is applied consistently and unambiguously.

4.2.2 Data Cleaning

The accuracy of the SP estimation model is directly impacted by the quality of the data. The data is thoroughly processed and cleaned in order to provide reliable and accurate predictions. This phase includes:

- Removing incomplete or missing data: Empty rows and rows with undefined values are eliminated as they lead to bias and the quality of predictions can

be impacted negatively

- Removing irrelevant data: Rows with URLs that are not directly related to the estimation context are eliminated to reduce noise and keep attention to important features
- Removing noise: Removing stop words, articles and prepositions from the dataset is a good way to reduce noise because they don't add much to the semantic meaning. For maintaining consistency emojis and whitespace are also excluded.
- Eliminating unnecessary formatting: HTML tags are removed so that the models can focus on the main text that is important for estimating SP.

Dataset A contains 34,484 data items and Dataset B contains 25,854 data items after going through the data filtering and data cleaning phases.

4.2.3 Removing Similar User Stories

A semantic similarity-based data preprocessing approach was used to address the problem of inconsistent SP values assigned to similar user stories. There were two parts to this approach:

1. Sentence-BERT is a robust pre-trained neural network model that was used to extract the rich semantic meaning that is embedded in each user story. Sentence-BERT considers the relationships and context between words in addition to individual words, in contrast to traditional word embedding approaches. This approach provides a more comprehensive representation of the overall meaning of user stories [30].
2. The level of semantic similarity across user stories was measured using cosine similarity which is a well-known metric in text analysis. Measuring the degree to which the embedded representations aligned was done by computing the cosine similarity between the user stories. This approach filtered out the user stories which has similar semantic means with different SP values which led to reduced inconsistencies and noise in the dataset. Furthermore, in previous studies, it was mentioned that cosine similarity was a commonly used metric and it has a high efficiency [31].

3892 data items were eliminated after filtering out similar user stories from the

dataset.

4.2.4 Label Encoding

In this study, we address the issue as a multi-label classification problem. Therefore each SP value must be assigned a particular number. When it comes to transforming the SP values in the Fibonacci sequence into a format that is suitable for the ML models, the SP were encoded from 0 to 4 in the numeric sequence. This encoding uses the Fibonacci sequence's inherent ordering to guarantee that each encoded number appropriately represents the relative complexity of the associated story. The encoding of the SP values is given below.

- SP value 1 gets encoded into the value 0
- SP value 2 gets encoded into the value 1
- SP value 3 gets encoded into the value 2
- SP value 5 gets encoded into the value 3
- SP value 8 gets encoded into the value 4

However, we can also address this as a 4 step binary classification problem. In that situation, the encoding values should only contain values 0 and 1. An example of the first encoding step of SP values is given below. This is further explained in the upcoming sections.

- SP value 1 gets encoded into the value 0
- SP value 2 gets encoded into the value 1
- SP value 3 gets encoded into the value 1
- SP value 5 gets encoded into the value 1
- SP value 8 gets encoded into the value 1

4.2.5 Tokenization and Vectorization

This step of preprocessing involves creating a representation using either tokenization or vectorization. Tokenization is an essential pre-processing step that was applied to the text data handled in this study in accordance with the particular needs of the transformer models that were used. The BERTTokenizer, which can handle sequences with up to 512 distinct tokens was used for the BERT model. The DistilBertTokenizer was selected in order to utilize the DistilBERT model that en-

ables sequences up to 2048 tokens in length. Lastly, the RoBERTa model is used with the RobertaTokenizer, which enables sequences with up to 2048 tokens. Due to the hardware constraints of the ANTPC, the sequence length was limited to 128 tokens on each of the transformer models. While limiting the sequence length to 128 tokens, some descriptive information may inevitably be lost. To overcome this issue, an approach was used which involved the segmentation of the original user story descriptions into smaller and more manageable chunks. This segmentation attempted to preserve the essential elements of the user story descriptions while maintaining the sequence length restriction that had been imposed upon it. An additional step of pre-processing was implemented to overcome the repetition of including the title for every description segment. The titles and descriptions were merged into a single column then segmentation of that column was done using a specific segmentation number.

Text vectorization is essential to the efficient processing and analysis of textual input by traditional ML approaches such as SVM. This study utilizes the TF-IDF vectorization approach, combining n-grams and also uses Word2Vec to capture the semantic context and complex relationships within the textual data.

4.2.6 Handling Class Imbalance Using Data Augmentation and Class Weights

The dataset has a class imbalance across all 5 classes. Class weights were utilized in order to overcome this problem and guarantee that the model gives minority classes sufficient attention. The equation that was used to calculate the class weight for the n^{th} label is given below.

$$\text{class weight for label } n = \frac{\sum_{i=0}^4 \text{samples in class label } i}{\text{samples in class label } n}$$

Before data augmentation is done, the dataset is divided into 3 subsets, 70% for training, 15% for validation and 15% testing. Furthermore, data augmentation is only done on the training set because keeping the validation and testing sets intact

allows for an unbiased evaluation of the model’s performance on real-world data. This approach guarantees that the models are trained on a variety of augmented instances while maintaining the integrity of the unseen testing and validation data for a fair evaluation.

This study utilized NLPAug [32] which is a versatile library that provides 3 data augmentation strategies namely character-level, word-level and sentence-level. Every level of data augmentation includes synonym replacement, random insertion, random deletion and shuffling techniques. Out of all the strategies, replacing synonyms with word embeddings is the most common and efficient method. With this technique, words are strategically replaced with their synonyms to create sentences that have the exact same meaning but a different vocabulary. For synonym replacement, selecting pre-trained embeddings is crucial. For this, we can either use contextual embeddings such as BERT and RoBERTa or non-contextual embeddings such as word2vec and GloVe. For this study, we have utilized sentence-level contextual embeddings using BERT.

4.3 Model Training and Experiments

4.3.1 Environment

All the ML models were trained on the ANTPC server which is a high-performance computing environment with 4 GeForce RTX 2080 Ti GPUs each with an impressive 11GB of dedicated memory. In addition, the ANTPC server has 128GB of RAM and 32 CPU cores which make training procedures more efficient. Additionally, Google Colab was used to increase training capacity. The main programming language used to train the ML model was Python 3.8. The main libraries used were NumPy for numerical computation, Matplotlib for data visualization, PyTorch for deep learning, Pandas for data manipulation and huggingface for implementing the transformer models.

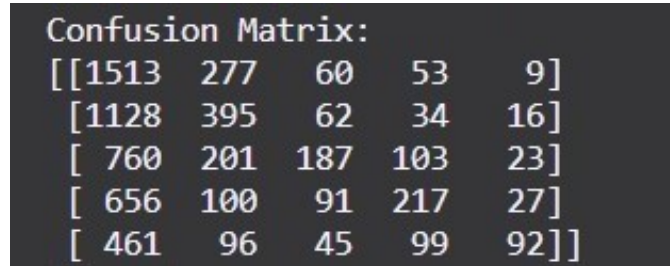
4.3.2 Model Evaluation Metrics

Several metrics were used to evaluate the effectiveness of how each classification model predicted the SP values between 0 and 4.

1. **Confusion Matrix:** This gives the distribution of the model's predictions in relation to the actual labels by providing a summary of the number of True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives (FN). This enables a more in-depth understanding of the model's performance across all classes. This matrix was used in some of the experiments to understand the model's prediction decisions. The confusion matrix is given in the table 4.1. An example of the confusion matrix of an experiment done in this study is given in the figure 4.4.

	Predicted Class	
	Positive	Negative
Positive	TP	FP
Negative	FN	TN

Table 4.1: Confusion Matrix



```
Confusion Matrix:
[[1513 277 60 53 9]
 [1128 395 62 34 16]
 [ 760 201 187 103 23]
 [ 656 100 91 217 27]
 [ 461 96 45 99 92]]
```

Figure 4.4: Confusion Matrix experiment

2. **Accuracy:** This commonly used matrix shows the total percentage of accurate predictions the model makes. In terms of math, it is computed as below. Other than the accuracy, for some experiments the accuracy for each class was also individually measured to see the performance for each class. An example of the class-wise accuracy of an experiment done in this study is given in the figure 4.5.

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

```
Test Class-wise Accuracy:
Class 0 : 0.4605809128630705
Class 1 : 0.36811832374691866
Class 2 : 0.34782608695652173
Class 3 : 0.4287531806615776
Class 4 : 0.4336734693877551
```

Figure 4.5: Class-wise accuracy experiment

3. **MAE**: This matrix evaluates the average size of the gap between actual and predicted SP values. A lower MAE denotes better model performance, implying smaller average prediction mistakes.
4. **MdAE**: The central tendency of prediction errors is the main focus of this matrix. In comparison to MAE, it is less sensitive to outliers and shows the median of the absolute differences between the values that were predicted and the actual values. For the majority of data points, a lower MdAE indicates that the model typically generates predictions that are closer to the true values.

4.3.3 Model Implimentation

Random Forest

The implemented Random Forest model utilized the RandomForestClassifier class from the scikit-learn library. TF-IDF vectorization was used to prepare the textual data for the model. Furthermore, the n-gram range was set to 1-2, indicating that the model considered unigrams and bigrams during vectorization.

RNN

The implemented RNN model employs a sequential architecture that consists of the following layers,

- **Embedding Layer**: This layer converts the text data into numerical vectors. It also captures the semantic relationship between the words.
- **Recurrent Layer**: This layer is implemented using either the LSTM or the BiLSTM. The LSTM processes sequential data which enables it to identify

long-term dependencies. However, unlike the LSTM, the BiLSTM considers both forward and backward directions.

- Dense Layer: This layer maps the processed features to the predicted class.

For training the model, 10 epochs were used with a batch size of 64. For text preprocessing, the Keras tokenizer was utilized and the maximum sequence length was set to 1000 tokens. The following regularization techniques were utilized for overfitting prevention.

- Dropout: The initial value was set to 0.2
- L1 and L2 regularization: The initial values were set to 0.01

SVM

This study explored 2 separate data preprocessing approaches to evaluate how they affected the performance of the SVM model.

1. Vectorization-based Preprocessing: Utilizes the TF-IDF Vectorizer with n-grams and Word2Vec methods for transforming text data into numerical representations. This approach captures the semantic relationships between words. Additionally, n-gram ranges 1-2, 1-3, 1-4, 1-5, and 1-6 were explored to test the effects of considering word combinations of varying lengths.
2. Tokenization-based Preprocessing: Utilizes the BERT tokenizer with the bert-base-uncased pre-trained model. The maximum sequence length was set to 128 tokens.

In order to optimize the SVM model's hyperparameters, a grid search approach was utilized. This rigorous optimization approach determines the combination of hyperparameters that provides the highest efficiency by systematically evaluating a predetermined range of values. Specifically, the grid search explored multiple kernel functions including polynomial, linear, sigmoid and Radial Basis Function (RBF) to capture distinct underlying data relationships. Furthermore, it considered a range of gamma values including 1, 0.1, 0.001, and 0.0001 to manage the impact of data points in the kernel function and C values including 1, 10, 100, 1000, 5000, and 10000 to control the trade-off between training error and model complexity.

The initial implementation phase involved setting a baseline configuration for the

SVM model. This utilized the TF-IDF with the n-gram range of 1-2 for feature representation. The RBF kernel function was utilized due to its flexibility in handling non-linear relationships. Furthermore, an initial gamma value of 0.1 and a C value of 1 were chosen. These values were selected based on common practices and served as a basis for further hyperparameter exploration.

Transformer Models

The implemented transformer models utilized bert-base-uncased, distilBERT-base-uncased and RoBERTa-base as pre-trained models and tokenizers. The maximum sequence length was set to 128 tokens in order to prepare the text data for the models. AdamW optimizer was utilized in the training process due to its stability and effectiveness. A batch size of 16 was utilized and a variety of learning rate values (1e-3, 1e-4, 1e-5, 2e-3, 2e-5, 3e-5, 4e-5, and 5e-5) were evaluated in order to optimize the learning process. Determining the optimal value for learning rate can have significant effects on performance since it regulates the stability and speed of the learning process. Furthermore, a variety of weight decay values were used including 0.0001, 0.001, 0.01, 0.1, 0.2, 0.3 and 0.4. Also, a variety of dropout rates were used including 0.01, 0.05, 0.1, 0.15, 0.2, 0.3 and 0.4 were also used to prevent overfitting.

The initial configuration of the models utilized a batch size of 16, a number of epochs of 4, a dropout rate of 0.1, a weight decay of 0.01 and a learning rate of 2e-5. Following the initial configuration, a systematic hyperparameter tuning process was implemented to optimize the model's performance.

4.4 XAI Approaches

The main goal of this study was to bridge the knowledge gap between the model's prediction decisions and human comprehension by explaining its predicted outputs in a way that is familiar to developers. Out of all the interpretability approaches that were explored, highlighting the words based on their relevance to the final prediction emerged as the most effective method. This approach simplifies comprehension by pointing developers to the exact textual components which influenced the model's predictions.

4.4.1 LIME

LIME was utilized as one of the XAI methods to explain the predictions made by traditional approaches and transformer models. The two main forms of explanation outputs generated by LIME are visualizations which use distinct colours to differentiate the class probabilities and textual explanations in which the words are displayed in either tabular format or with words highlighted directly in the text. Further customization of the LIME method can be done by adjusting the number of features displayed which shows the most relevant words for the prediction and setting the number of labels to show which shows the features contributed for each class.

The initial configuration was done by setting the number of features to 10 and the number of labels to 1. Additionally, the textual explanations are set to display the highlighted words. An example of the explanation given by LIME is illustrated in figure 4.6.

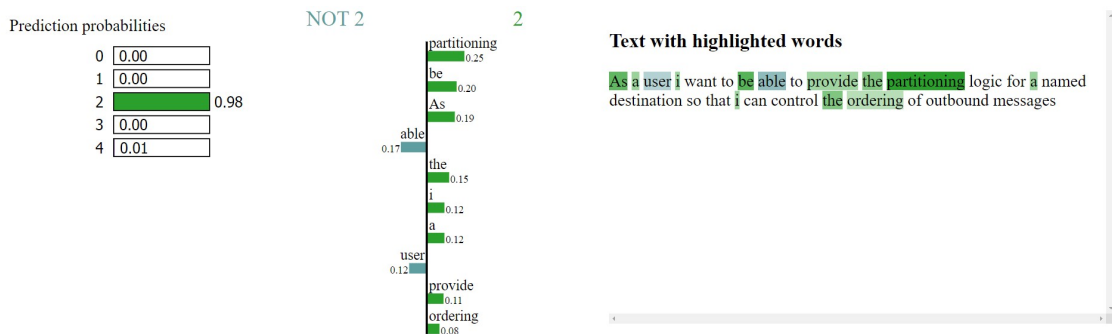


Figure 4.6: Explainability output from LIME

The explanation illustrates 3 main components, prediction probabilities for each class, relevant words for the predicted class and the given text with highlighted words. LIME gives the explanation using 2 different colours with an opacity to indicate their relevance for the prediction. Each class is assigned a particular colour and the 2 colours vary depending on the predicted class.

4.4.2 SHAP

Although SHAP provides several methods of explanation, the "highlighted words based on relevance" method is currently limited to just transformer models. In this

SHAP method 2 parameters should be given. The first parameter is the prediction function which predicts the SP value of the given text and the second parameter is the tokenizer used. Following that, the method then determines the Shapley values for every feature in the given text. Lastly, we can visualize the explanation with the plotting functionalities given by the SHAP library. An example of the explanation given by SHAP is illustrated in figure 4.7.

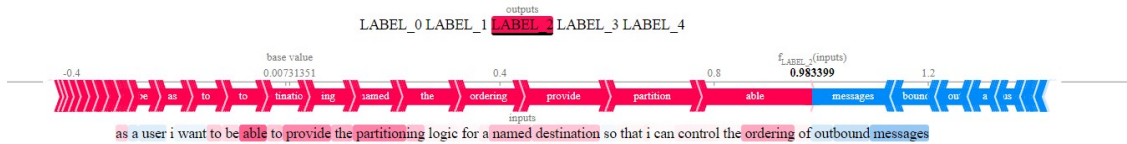


Figure 4.7: Explainability output from SHAP

The explanation visually highlights the predicted label in red colour. By clicking on it, you can see the words with different levels of opacity that had a significant influence on the prediction. A red colour denotes a positive contribution to each related feature’s prediction, while blue denotes a negative contribution. Furthermore, the opacity of the highlighting allows to further identify important words. The opacity of the highlighting distinguishes the relevant words: a highly visible red highlight indicates a highly relevant feature, whereas a highly visible blue indicates a feature that has less of an influence on the prediction.

4.4.3 Transformer-Interpret

Transformer-Interpret is a XAI library which works on every transformer model. This tool provides insightful information about transformer model decision-making processes. Additionally, this approach also has the "highlighted words based on relevance" method. The SequenceClassificationExplainer provided in the Transformers-Interpret library is designed to explain sequence classification problems, like the one used in this work. Therefore as the explainer, the SequenceClassificationExplainer was utilized. The tokenizer and model used for the prediction are passed in as parameters to configure this explainer. Following that, when the text is passed to the explainer, it provides numerical attribution scores for every word. Using these numerical attribution scores, we can plot the explanation. The explanation provides 2 colours for word highlighting. The green colour indicates that it supports

the prediction and the red colour indicates it doesn't support the prediction. An example of the explanation given by Transformer-Interpret is illustrated in figure 4.8.

Legend: ■ Negative □ Neutral ■ Positive				
True Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
2	LABEL_2 (0.97)	LABEL_2	1.99	[CLS] as a user i want to be able to provide the partition ##ing logic for a named destination so that i can control the ordering of out ##bound messages [SEP]

Figure 4.8: Explainability output from Transformer-Interpret

The explanation clearly describes the meaning of the highlighted colours and it also displays the predicted label on the left-hand side of the given explanation. This method gives the explanation using the colours red and green with an opacity to indicate their relevance for the prediction. The green colour indicates a positive contribution to the prediction, while the red colour indicates a negative contribution.

Chapter 5 - Results and Evaluation

The research employed a cross-project evaluation, utilizing dataset A and dataset B. User stories from a variety of 39 projects are included in each dataset and they consist of 25,854 and 34,484 data items in total. This selection of datasets guarantees unbiased and lower noise in the features of the datasets. Both the datasets were divided into 3 parts, 70% for training, 15% for validating and 15% for testing. Furthermore, in order to evaluate the efficacy of the selected models, the comparison of the results with the Choetkiertikul dataset was also done. This comparison provides perspective and aids in determining how effective the proposed approach is in comparison. Also, the cross-project state-of-the-art accuracy for SP estimation is 47.2% which is achieved using the GPT2SP model [2]. Achieving this objective would be a major step forward for the area of study and show the potential of our proposed approach.

5.1 First Phase of Experiments Using the Traditional Approaches

5.1.1 Experiment I - Using Random Forest

The study evaluated a baseline Random Forest model's performance on the 3 datasets. Accuracy was used to evaluate the model's performance. The results showed that there was diversity in the datasets' accuracy, with Dataset A obtaining the highest accuracy of 37.7%. The accuracy for Dataset B was 37.3% and the accuracy for the Choetkiertikul dataset was 37.1%.

5.1.2 Experiment II - Using RNN with LSTM

This study evaluated the 3 datasets using 2 recurrent neural network RNN architectures, LSTM and BiLSTM, in order to provide a baseline performance benchmark. Each dataset was evaluated using both the LSTM and BiLSTM models.

The LSTM model obtained a 35.7% accuracy with an MAE of 1.013 on dataset A, 35.3% accuracy with an MAE of 1.033 on dataset B and 34.4% accuracy with an

MAE of 1.108 on the Choetkiertikul Dataset.

It was observed that the BiLSTM model outperformed the LSTM model across all datasets. The BiLSTM model obtained an accuracy of 36.7% and an MAE of 0.957 on Dataset A. With an MAE of 0.96, the BiLSTM model obtained an accuracy of 35.6% on Dataset B. Lately, with an MAE of 1.1, the BiLSTM model obtained an accuracy of 35.1% on the Choetkiertikul Dataset.

5.1.3 Experiment III - Using SVM

The study utilized an SVM model in a baseline configuration on all 3 datasets to create a baseline performance benchmark. The model obtained an accuracy of 42.5% with a MAE of 0.911 on Dataset A, an accuracy of 40.5% with a MAE of 0.94 on Dataset B and an accuracy of 38.9% with a MAE of 1.01 on the Choetkiertikul dataset.

5.1.4 Conclusion on the First Phase of Experiments Using the Traditional Approaches

In evaluating various ML models across the 3 datasets, SVM model consistently performed well compared to other models. Notably, dataset A was the dataset on which all models achieved the highest accuracy, whereas the Choetkiertikul dataset achieved the lowest accuracy. Given the results, the SVM model has been chosen for additional exploration because of its overall efficiency and promising results. A summary of the results is given in the table 5.1

5.2 Second Phase of Experiments Using the Traditional Approaches

The second phase of experiments involved utilizing the SVM model since it demonstrated promising results compared to other models. This phase included several key components: hyperparameter tuning to optimize model performance, exploring how various embedding techniques affect classification accuracy, analyzing the model's sensitivity to data segmentation by feeding it smaller description chunks and at last, evaluating the model's performance with the augmented dataset.

Model	Dataset	Accuracy
Random forest	Choetkiertikul dataset	37.1%
	Dataset A	37.7%
	Dataset B	37.3%
SVM	Choetkiertikul dataset	38.9%
	Dataset A	42.5%
	Dataset B	40.5%
LSTM	Choetkiertikul dataset	34.2%
	Dataset A	35.7%
	Dataset B	35.3%
Bi-LSTM	Choetkiertikul dataset	35.1%
	Dataset A	36.7%
	Dataset B	35.6%

Table 5.1: A summary of the performance of traditional approaches

5.2.1 Hyper Parameter Tuning

In this phase, a systematic hyperparameter tuning process was undertaken. The model’s performance was evaluated using several kernel functions in the first step. It was observed that the RBF kernel obtained the highest accuracy of 42.5%, followed by the linear kernel at 41.7%, the sigmoid kernel at 41.4% and lastly the polynomial kernel at 23.8%. Since the RBF kernel outperformed the other kernels, it was selected for kernel for subsequent experiments. The best combination of the gamma and C parameters was then found using a grid search. According to the results, an RBF kernel with C set to 1000 and gamma set to 0.1 produced the best results. Furthermore, TF-IDF was utilized with n-grams for vectorization. The selected SVM model was evaluated with different n-gram ranges and it was observed that the 1-4 n-gram range achieved the best result. The model performance for the considered n-gram ranges is given in 5.2.

5.2.2 Experiment I - Using Various Embedding Techniques

This experiment explored the effectiveness of various embedding techniques for SVM classification utilizing Word2Vec, SentenceBERT and BERT embeddings.

n-gram range	Accuracy
1-2	42.5%
1-3	42.8%
1-4	42.9%
1-5	42.3%
1-6	42.2%

Table 5.2: Performance of the SVM model on various n-gram ranges

Utilizing Word2Vec embeddings achieved an accuracy of 35.8%, SentenceBERT embeddings achieved an accuracy of 32% and BERT embeddings achieved an accuracy of 29.7%. Out of the 3 embedding techniques, Word2Vec embeddings showed the highest performance however TF-IDF still outperforms the Word2Vec.

5.2.3 Experiment II - Description Segmentation

This experiment explored the effect of description segmentation on SVM model performance in order to understand how textual word size affects model performance. According to this analysis, the best results were obtained by chunking the description into 15-word segments. A variety of chunk sizes including 5, 10, 15, 20, 30, 40 and 60 words were evaluated in the experiment which demonstrates how sensitive the SVM model is to the level of detail in the input data. The relevant accuracies are given in the table 5.3.

Word size	Accuracy
5	43.2%
10	42.2%
15	42.6%
20	42.2%
30	41.7%
40	41.3%
60	41%

Table 5.3: Performance of the SVM model on various word chunk lengths

5.2.4 Experiment III - Augmented Dataset

An accuracy of 43.21% was obtained from the model's first evaluation using the augmented dataset. Then this study explored the potential advantages of implementing word segmentation as a preprocessing step after seeing this early performance. Positive effects on model performance were observed when word segmentation was utilized in the augmented dataset. With 46.12% accuracy, this achieved a significant improvement. Based on this outcome, it is possible to improve the model's ability to extract significant features and eventually increase prediction accuracy by segmenting words within the text data.

5.2.5 Conclusion on Second Phase of Experiments Using the Traditional Approaches

The SVM model's accuracy progressively improved throughout the experiments. This pattern indicates that the model parameter can be learned and optimized effectively. Out of all the configurations that were explored, the model performed at its best with 46.12% accuracy. The RBF kernel with a C value of 1000, gamma value of 0.1, n-gram range of 1-4 and word segmentation size of 15 was used in this configuration. This outcome shows how well the model can learn complex relationships from the data and highlights the importance of hyperparameter tuning for optimal performance.

Additionally, the obtained accuracy was quite near to the state-of-the-art accuracy in this domain at the time. This result implies the effectiveness of the selected approach in the field as well as its efficacy.

5.3 First Phase of Experiments Using the Transformer Models

To determine the most promising transformer model for further exploration, 2 baseline experiments were conducted for each of the 3 candidate models.

The first experiment is done using single-label multi-class classification. The performance of each model was evaluated utilizing a single multi-class classifier that predicted the SP value directly. Using this method, the SP is supposed to be classified into predefined classes (1, 2, 3, 5, 8).

The second experiment used a novel strategy that utilized 4 binary classifiers. Every classifier was designed to differentiate between particular ranges of SP values.

1. **Classifier 1:** differentiate between SP values of 1 and the combined set of 2, 3, 5 and 8.
2. **Classifier 2:** differentiate between SP values of 2 and the combined set of 3, 5 and 8.
3. **Classifier 3:** differentiate between SP values of 3 and the combined set of 5 and 8.
4. **Classifier 4:** differentiate between SP values of 5 and 8.

The overall architecture of the second experiment is given in the figure 5.1.

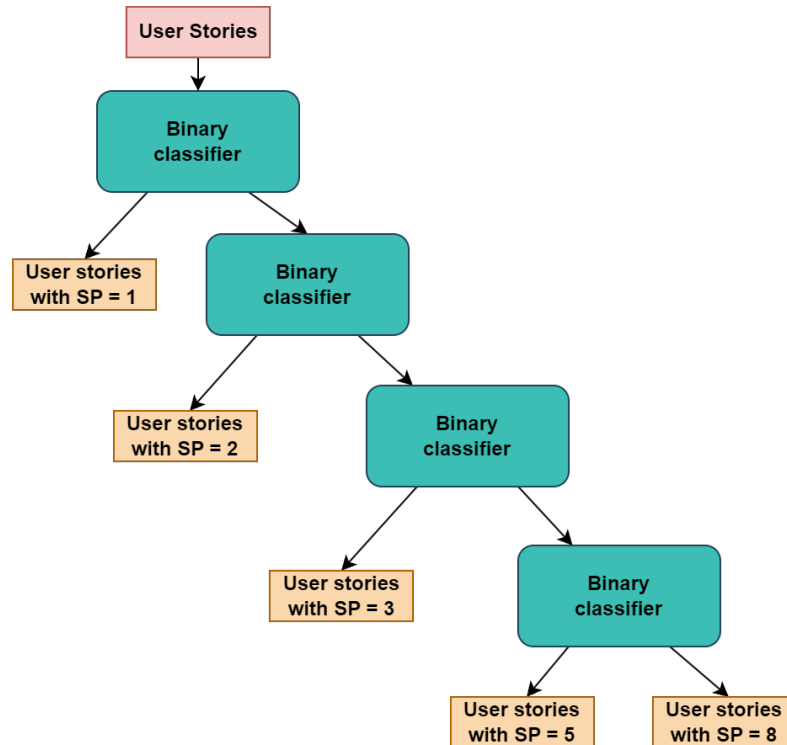


Figure 5.1: The architecture of the second experiment

5.3.1 Experiment I - Using RoBERTa

The Roberta transformer model in a baseline configuration was initially utilized as the first candidate for the experiments. The model obtained an accuracy of 37.41% with a MAE of 0.9543 on Dataset A, an accuracy of 37.46% with a MAE of 0.9691 on Dataset B and an accuracy of 35.24% with a MAE of 1.1077 on the Choetkiertikul dataset. Then for the second experiment, this achieved the below accuracies for the 4 binary classifiers.

1. **Classifier 1:** Accuracy of 70.01% with a MAE of 0.2999
2. **Classifier 2:** Accuracy of 73.38% with a MAE of 0.2662
3. **Classifier 3:** Accuracy of 61.67% with a MAE of 0.3833
4. **Classifier 4:** Accuracy of 65.56% with a MAE of 0.3444

5.3.2 Experiment II - Using DilstilBERT

The DilstilBERT transformer model in a baseline configuration was then utilized as the second candidate for the experiments. This configuration achieved an accuracy of 28.92% and a MAE of 1.1045 on Dataset A, 36.67% accuracy and 0.9976 MAE on Dataset B and 33.81% accuracy with 1.1092 MAE on the Choetkiertikul dataset. Then for the second experiment, this achieved the below accuracies for the 4 binary classifiers.

1. **Classifier 1:** Accuracy of 68.12% with a MAE of 0.3188
2. **Classifier 2:** Accuracy of 76.12% with a MAE of 0.2383
3. **Classifier 3:** Accuracy of 58.93% with a MAE of 0.4107
4. **Classifier 4:** Accuracy of 65.56% with a MAE of 0.3444

5.3.3 Experiment III - Using BERT

The BERT transformer model in a baseline configuration was finally utilized as the last candidate for the experiments. Its performance varied across the datasets. On Dataset A, it achieved an accuracy of 38.29% and a MAE of 0.9537. Dataset B followed closely with an accuracy of 38.96% and MAE of 0.9202. The Choetkiertikul dataset presented the most challenging task, with the model achieving an accuracy of 36.32% and MAE of 1.011. Then for the second experiment, this achieved the

below accuracies for the 4 binary classifiers.

1. **Classifier 1:** Accuracy of 70.35% with a MAE of 0.3188
2. **Classifier 2:** Accuracy of 76.60% with a MAE of 0.2383
3. **Classifier 3:** Accuracy of 62.29% with a MAE of 0.4107
4. **Classifier 4:** Accuracy of 65.82% with a MAE of 0.3444

5.3.4 Conclusion on First Phase of Experiments Using the Transformer Models

In evaluating various transformer models across the 3 datasets and utilizing 4 binary classifiers, the BERT model consistently performed well compared to other models. The DistilBERT model achieved the lowest accuracy on Dataset A. This might be due to the potential trade-off between model complexity and performance, particularly when dealing with Dataset A's features.

A noteworthy observation was identified regarding the performance across different models and datasets. Unlike traditional models, which achieved their highest accuracy on Dataset A, the transformer models achieved their highest accuracy on Dataset B. Furthermore, both BERT and RoBERTa models achieved the lowest accuracy on the Choetkiertikul dataset. However, DistilBERT achieved the lowest accuracy on Dataset A. The results obtained from all models on each dataset and 4 binary classifiers are given in tables 5.4 and 5.5.

Furthermore, while the overall performance of the 4 BERT binary classifiers might initially suggest promising results, a deeper analysis reveals a different picture. When analyzing the class-wise performance, the individual class accuracy scores drastically decrease, indicating that the models might struggle to distinguish between certain classes effectively. Therefore this approach is not taken for further explorations.

Model	Dataset	Accuracy
RoBERTa	Choetkiertikul dataset	35.24%
	Dataset A	37.41%
	Dataset B	37.46%
DilstilBERT	Choetkiertikul dataset	33.81%
	Dataset A	28.92%
	Dataset B	36.67%
BERT	Choetkiertikul dataset	36.32%
	Dataset A	38.29%
	Dataset B	38.96%

Table 5.4: A summary of the performance of transformer models on the 3 datasets

Model	Classifier	Accuracy
RoBERTa	Classifier 1	70.01%
	Classifier 2	73.38%
	Classifier 3	61.67%
	Classifier 4	65.56%
DilstilBERT	Classifier 1	68.12%
	Classifier 2	76.12%
	Classifier 3	58.93%
	Classifier 4	65.56%
BERT	Classifier 1	70.35%
	Classifier 2	76.60%
	Classifier 3	62.29%
	Classifier 4	65.82%

Table 5.5: A summary of the performance of transformer models on the 4 classifiers

5.4 Second Phase of Experiments Using the Transformer Models

The second phase of experiments involved utilizing the BERT model since it achieved better results compared to other transformer models. Due to the close alignment between the accuracy achieved on Dataset A and Dataset B and considering the specific features of each dataset, Dataset A was chosen for further exploration of

the BERT model. This phase included several key components: hyperparameter tuning to optimize model performance, analyzing the model’s sensitivity to data segmentation by feeding it smaller description chunks and evaluating the model’s performance with the augmented dataset.

5.4.1 Hyper Parameter Tuning

This phase focused on optimizing the hyperparameters of the BERT model to improve its performance. A systematic grid search technique was employed to achieve this. The grid search explored hyperparameters including learning rate, weight decay and dropout rates. This aimed to determine the configuration that optimizes the BERT model’s performance by systematically evaluating various combinations of these hyperparameters. The best configuration was found to have a learning rate of $2e-5$, a weight decay of 0.001 and a dropout rate of 0.3 which achieved an accuracy of 39.30%. Therefore this configuration is utilized for further experiments done in this phase.

5.4.2 Experiment I - Description Segmentation

This experiment explored the effect of description segmentation on the BERT model performance in order to understand how textual word size affects model performance. Additionally, this can be used to overcome the issue of information loss. According to this analysis, the best results were obtained by chunking the description into 15-word segments. Various chunk sizes including 5, 10, 15, 20, 30 and 40 words were evaluated in the experiment, demonstrating how sensitive the BERT model is to the level of detail in the input data. The relevant accuracies are given in the table 5.6.

The experiments revealed a pattern with the highest accuracy achieved when the text was segmented into chunks of 15 words. This result suggests that a chunk size of 15 words may be optimal for capturing relevant information within the data while maintaining computational efficiency and avoiding information loss. Therefore the subsequent experiment utilized a chunk size of 15 words.

Word size	Accuracy
5	39.90%
10	40.19%
15	41.65%
20	41.28%
30	40.91%
40	40.49%

Table 5.6: Performance of the BERT model on various word chunk lengths

5.4.3 Experiment II - Augmented Dataset

The initial exploration utilized the augmented dataset which achieved an accuracy of 42.37%. Subsequently, the exploration utilized word segmentation with a chunk size of 15 words. This approach resulted in a notable improvement in accuracy to 44.58%. According to this observation, word segmentation might be crucial for identifying the semantic relationships within the data and improving model performance.

5.4.4 Conclusion on Second Phase of Experiments Using the Transformer Models

Throughout the experiment evaluations, the accuracy of the BERT model showed a pattern of progressive improvement. The observation indicates that the model could efficiently learn and optimize its internal parameters. Across the range of evaluated combinations, the model reached the highest accuracy of 44.58%. The learning rate of $2e-5$, the dropout rate of 0.3, the weight decay of 0.001 and the word segmentation size of 15 were all used in this optimal configuration.

5.5 Human-Based Evaluation

To evaluate the efficacy of the LIME, SHAP and Transformer Interpret methods, a human-centered evaluation [33] was conducted with 7 industry professionals. The group consist of 4 business analysts and 3 software engineers with experience in SP estimation. The study employed a data collection approach through a Google Sheet

and each participant was given 19 identical user stories. Following a structured format, participants were instructed to do the following.

1. Read and understand each user story.
2. Select the most fitting SP from the provided SP list. The SP values include 1,2,3,5,8 and larger than 8
3. Identify the keywords that influenced their SP estimation by entering them into the designated field.
4. Rate their confidence level in each SP estimation and the understanding of the user story on a scale of 1 (not confident) to 5 (highly confident).

This collected data serves as a benchmark for evaluating the effectiveness of the XAI methods of this study in highlighting the key textual features driving the model’s predictions. The answers provided by a participant is given in the figure 5.2.

Issue number	Description	Stroypoint value	Key words which used to decide the storypoint	Confidence level on the description
1	As a user id like to push the custom module via a rest api so that i can install the custom module in cluster	5	Custom module, REST API, cluster	5
2	As a user I like to have an option of aws source module so that I can ingest data from amazon s3 or use the simple email service reference spring integration aws extension	8	AWS Source Module, Ingest data from Amazon S3, Simple email service, reference spring integration	4
3	As a user i want to be able to provide the partitioning logic for a named destination so that i can control the ordering of outbound messages	Larger than 8	Partitioning logic, named destinations, control message ordering	3
4	As a user im trying to load task task deployment and task executions page but im seeing an error instead	5	Error, load, page	5
5	As a user I would like to have a landing page with higherorder links for sources processors sinks and jobs so i can jump to right section from one place	5	Landing page, higher-order links, jump to right section	5
6	As a user I would like to refer to the analytics tab docs so i can understand how to use various widgets from streaming pipeline	3	Analytics tab docs, widgets, understand, streaming pipeline	5
7	As a user I would like to refer to the documentation to configure the properties file so i can use it as recommended to represent the deployment manifest	2	Documentation, configure the properties file, deployment manifest	5

Figure 5.2: The answers provided by a participant

A key challenge arose when comparing SP estimations from the SVM and BERT models, the actual SP in the dataset and human-based estimates. The reason for this challenge is because of the nature of subjectivity in this area of research as participants assigned different SP values to the same user stories depending on their understanding. To address this, the study devised a metric leveraging participants’ confidence scores. This metric functioned similarly to a voting system, but instead of simply counting votes, it takes the summation of the confidence levels assigned to each SP estimate. For instance, if 2 participants assigned a SP of 2 with a confidence level of 4, 3 participants assigned a SP value of 5 with a confidence of 3

and 2 participants assigned a SP value of 8 with a confidence of 5, the confidence score would be 8 (4+4) for SP value 2, 9 (3+3+3) for SP value 5 and 10 (5+5) for SP value 8. Utilizing this approach, the SP with the highest confidence score (in this case 8) was considered as the human-based estimation for comparison with model predictions and actual SP.

The comparison of all the SP predictions done for those 19 user stories using the SVM model, BERT model and human-base estimation is given in the table 5.7.

ID	Actual value	SVM predictions	BERT predictions	human-based
1	8	3	1	3
2	1	5	5	8
3	1	5	2	5
4	1	1	1	2
5	1	5	5	5
6	2	3	5	1
7	1	3	5	1
8	2	5	5	2
9	2	3	5	3
10	8	5	1	1
11	1	1	2	3
12	3	3	5	5
13	8	8	3	1
14	2	5	2	2,3
15	3	3	3	5
16	5	5	8	3
17	8	5	2	3
18	8	3	3	2
19	8	8	2	8

Table 5.7: Comparison of the SP predictions approaches

Table 5.7 reveals comparative performance between the SVM and BERT models in predicting SP. While both models have limitations in accurately matching human-based estimations (SVM: 5/19 correct, BERT: 4/19 correct), the SVM model

demonstrated a slight advantage in closely predicting SP values (SVM: 8/19 close predictions, BERT: 7/19 close predictions). However, when compared to actual story point values, the SVM model outperformed BERT with 7 correct predictions compared to BERT's 3. While both models achieved similar close predictions to actual values (SVM and BERT: 4/19), the SVM model displayed a more consistent ability to accurately estimate SP.

An analysis of the results presented in Table 5.8 provides valuable insights into the alignment between the 3 XAI methods and human decision-making in SP estimation. This alignment reflects how well the explanations generated by these methods correspond to the factors human experts consider when assigning SP values.

The analysis reveals that SHAP consistently identifies keywords most aligned with those used by participants in 14 out of 19 user stories. This suggests that SHAP's explanations effectively capture the aspects of the data that resonate most strongly with human reasoning during SP estimation. In contrast, LIME aligns with participant keywords in 5 user stories, and Transformer Interpret aligns in 3 stories. Notably, in 2 user stories, both SHAP and Transformer Interpret identified the same number of participant-aligned keywords, and in 1 story, SHAP and LIME achieved the same level of alignment. These findings highlight SHAP's effectiveness in generating explanations that closely mirror human decision-making processes for SP estimation tasks.

ID	Human-based	LIME (SVM)	LIME (BERT)	SHAP	Transformer Interpret
1	id, Custom module, REST API, cluster, installation	the, custom, REST	push, id, to, user, a, the	custom, the, can, in, cluster, I, install, like, as, REST API	id, module, custom, like
2	AWS Source Module, Ingest data from Amazon S3, Simple email service, reference spring integration, extension	spring, service, integration, email, aws, reference, extension	user, I, to, option, ingest, aws, an	aws, as, user, module, extension, data, simple, email, service, reference	amazon, option, source module, as, user
3	Partitioning logic, named destinations, control message ordering, provide	the, to, provide, logic, partitioning	as, I, user, a, want, to, control	destination, ordering, messages, user, as	I, want, partitioning, logic, named, out
4	Error, load, page, task, deployment, task executions,	task, instead, seeing, page	I'm, trying, user, to, task	executions, trying, load, task, page, seeing, error	as, user, load, trying, task
5	Landing page, higher-order links, jump to right section, sources, processors, sinks, jump, jobs	and, user, jobs, with, can, one	I, user, like, a, would, to	processors, links, higher, order, place, right, section, sources	as, landing, higher, order, links, sources, processors, sinks
6	Analytics tab docs, widgets, understand, streaming pipeline, refer docs	to, use, refer, docs, widgets, various, like	user, like, I, analytics, a, would, as	user, like, as, understand, docs, pipeline	would, widgets, refer, tab, docs
7	Documentation, configure the properties file, deployment manifest	to, documentation, refer, configure, like, represent	like, user, I, would, a, as, properties	documentations, recommended, as, user, properties, file	as, I, documentation, file, refer

8	Parameterize, import options, eliminate, args, confusing	it, all, user, for, the, can, all	parameterize, like, user, I, options, to	parameterize, options, args, options, confusing, like, user	as, user, I, parameterize, import, args
9	upload, custom, mavengradle targets, automate, installation, module, fragments	custom, to, the. also, automate	user, like, I, capability, to, would, so	custom, module, through, automate, upload	as, user, like, custom, targets, fragments
10	Description, Each module, module purpose & capabilities	the, to, each, of, it, use, have	to, id, like, have, description	description, understand, capabilities, as, for	user, description, modules, understand
11	Configure, threading profile, xml snippet code	threading, profile, code, todo, paste, XML, snippet	I, as, want, a, user, to, the, threading	XML, threading, batch, snippet, code, user, as	user, paste, XML, threading, configure, batch
12	Changing the runtime, better options, automatically upgrade all connectors	the, to, upgrade, all, runtime, better	user, will, I, like, that, changing, the	changing, offer, better, options, upgrade, connectors, like, as	as, like, changing, automatically, upgrade, all
13	Properties and flow variables, stay expanded or collapsed, moving, block to block in studio	block, to, user, studio, from, to, want	want, as, my, a, to, and	variables, stay, when, collapsed, block, moving	user, want, studio, stay
14	Name of the server, launching, version, acceptance criteria	name, version, should, server, be, see, of, criteria	see, as, user, a, I, name, want, of, to	version, should, visible, see, user	as, user, launching

15	First step, adding capability, building multiple configurations, reviewbot, build scripts, docker	to, build, scripts, change, building, step, use	as, adding, a, first, reviewbot, step	adding, capability, change, scripts, multiple, as	adding, capability, building, change, docker
16	First step, address the use cases, propose a design document, requirement design and implementation details	design, and, cases, document, purpose, details	step, address, to, first, the, cases, use, as	propose, implementation, step, as	address, the, use, cases, propose, design, covering
17	Configure, single point, common transport attributes, reusable place	to, the, in, be, Common	as, user, I, a, want, transport	transport, common, transport, attributes, single, reusable, place, config	user, common, transport, attributes, single, reusable, figure, single
18	Review the specs, contribute with my feedback, product owner or community member	be, to, able, member, community, review	community, as, a, owner, product, be	member, owner, community, specs, feedback, contribute	as, owner, member, review, spec, product
19	Tested against a real broker and a real application, transport works on real world usage	real, world, want, broker, works	as, transport, a, I, user, works	broker, transport, work, real, world, usage, sure	transport, world, usage, real, broker

Table 5.8: The keywords that each of the XAI approaches used to determine the SP value

Chapter 6 - Conclusions

This section addresses the research questions that guided this study, discusses its contributions and outlines potential future directions.

The first research question explored the possibility of enriching the existing Choetkiertikul dataset, which consists of 16 open-source projects. This study successfully expanded the dataset to 39 open-source projects, including the 16 projects in the Choetkiertikul dataset. By leveraging this enhanced dataset, the accuracy of the considered ML models in this study considerably improved. Furthermore, one key finding of this study is the positive impact of data augmentation on model accuracy. By leveraging data augmentation techniques, this study was able to enrich the dataset further and enhance the model's ability to learn complex relationships within the data. This ultimately contributed to improved model performance in terms of accuracy. This highlights the importance of data richness and diversity in achieving superior model performance. This research expands upon existing work by providing a collection of valuable datasets for story point estimation. The study offers 3 distinct datasets: Dataset A, Dataset B and an augmented version of Dataset A. This diversity allows researchers to explore different aspects of story point estimation. The augmented Dataset A provides a richer representation of the original data, potentially leading to improved model generalizability. By offering these datasets, this study facilitates further research and development in the field of story point estimation.

The second research question focused on optimizing the dataset through novel pre-processing steps beyond standard approaches like the removal of stop words and punctuations. This study explored and contributed innovative techniques such as removing similar user stories, data augmentation and description segmentation. These steps considerably improved model performance. For instance, the SVM model's accuracy increased from 42.1% to 46.12%, while the BERT model's accuracy rose from 41.1% to 44.58%. These results demonstrate how specific preprocess-

ing techniques can improve the accuracy and generalizability of models.

The third research question explored the potential of transformer models to outperform traditional ML approaches in generating SP values for user stories. While both traditional and transformer-based models were evaluated, the study revealed that SVM and BERT models achieved the highest accuracy among the considered approaches. Within these 2 models, the SVM model achieved a slightly higher accuracy of 46.12% compared to BERT's 44.58%. Furthermore, BERT's pre-trained nature offers advantages in general language understanding, the specific task of story point estimation may align better with the strengths of SVMs.

Additionally, the analysis also revealed that BERT performed better on Dataset B compared to Dataset A. This suggests that the inherent characteristics of the BERT architecture might be better suited to handle the specific data distribution or complexities present in Dataset B. Future research could investigate further into this observation to better understand the strengths and weaknesses of different model architectures for this specific task.

The final research question explored the application of XAI techniques to generate explanations for each SP prediction. 3 XAI approaches namely LIME, SHAP and Transformer Interpret were selected as they focus on highlighting keywords contributing to the model's predictions. Analysing the results revealed that SHAP provided the most insightful explanations, while Transformer Interpret generated the least informative ones. These findings contribute to a deeper understanding of the strengths and limitations of different XAI techniques. Techniques like SHAP, which align more closely with human reasoning processes, can be particularly valuable in tasks requiring interpretable explanations. Future studies can enhance user trust and understanding of the model's decision-making processes by incorporating XAI techniques like SHAP.

This study successfully addressed the research questions demonstrating the effectiveness of the chosen approaches. By enriching the dataset, implementing novel

preprocessing techniques and utilizing specific model architectures, the study made significant contributions to the field of story point estimation. Additionally, it highlighted the importance of XAI techniques in fostering model transparency and interpretability.

While resource constraints limited the use of the BERT-large-uncased model in this study, future research could explore its performance to determine if it surpasses the base BERT model's accuracy. Furthermore, exploring other transformer models such as the GPT-J model could yield valuable insights into the potential of alternative architectures for story point estimation. By continuing to explore these avenues, researchers can further refine and enhance the effectiveness of story point estimation techniques.

Bibliography

- [1] S. Alsaqqa, S. Sawalha, and H. Abdel-Nabi, “Agile software development: Methodologies and trends,” *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 14, no. 11, pp. 246–270, Jul. 2020. DOI: 10.3991/ijim.v14i11.13269. [Online]. Available: <https://online-journals.org/index.php/i-jim/article/view/13269>.
- [2] M. Fu and C. Tantithamthavorn, “Gpt2sp: A transformer-based agile story point estimation approach,” *IEEE Transactions on Software Engineering*, vol. 49, no. 2, pp. 611–625, 2023. DOI: 10.1109/TSE.2022.3158252.
- [3] M. Usman, E. Mendes, F. Neiva, and R. Britto, “Effort estimation in agile software development: A systematic literature review,” Sep. 2014. DOI: 10.1145/2639490.2639503.
- [4] M. Usman, E. Mendes, and J. Börstler, “Effort estimation in agile software development: A survey on the state of the practice,” in *Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering*, ser. EASE ’15, Nanjing, China: Association for Computing Machinery, 2015, ISBN: 9781450333504. DOI: 10.1145/2745802.2745813. [Online]. Available: <https://doi.org/10.1145/2745802.2745813>.
- [5] R. K. Mallidi and M. Sharma, “Study on agile story point estimation techniques and challenges,” *International Journal of Computer Applications*, vol. 174, pp. 9–14, Jan. 2021. DOI: 10.5120/ijca2021921014.
- [6] R. Guidotti, A. Monreale, F. Turini, D. Pedreschi, and F. Giannotti, “A survey of methods for explaining black box models,” *ACM Computing Surveys*, vol. 51, Feb. 2018. DOI: 10.1145/3236009.
- [7] M. Choetkiertikul, H. K. Dam, T. Tran, T. Pham, A. Ghose, and T. Menzies, “A deep learning model for estimating story points,” *IEEE Transactions on Software Engineering*, vol. 45, no. 7, pp. 637–656, 2019. DOI: 10.1109/TSE.2018.2792473.
- [8] M. Abadeer and M. Sabetzadeh, “Machine learning-based estimation of story points in agile development: Industrial experience and lessons learned,” *2021 IEEE 29th International Requirements Engineering Conference Workshops (REW)*, pp. 106–115, 2021.
- [9] B. Marapelli, A. Carie, and S. M. N. Islam, “Rnn-cnn model:a bi-directional long short-term memory deep learning network for story point estimation,” in *2020 5th International Conference on Innovative Technologies in Intelligent Systems and Industrial Applications (CITISIA)*, 2020, pp. 1–7. DOI: 10.1109/CITISIA50690.2020.9371770.
- [10] S. Porru, A. Murgia, S. Demeyer, M. Marchesi, and R. Tonelli, “Estimating story points from issue reports,” Sep. 2016, pp. 1–10. DOI: 10.1145/2972958.2972959.

- [11] P. Abrahamsson, I. Fronza, R. Moser, J. Vlasenko, and W. Pedrycz, “Predicting development effort from user stories,” in *2011 International Symposium on Empirical Software Engineering and Measurement*, 2011, pp. 400–403. DOI: 10.1109/ESEM.2011.58.
- [12] E. Scott and D. Pfahl, “Using developers’ features to estimate story points,” in *Proceedings of the 2018 International Conference on Software and System Process*, ser. ICSSP ’18, Gothenburg, Sweden: Association for Computing Machinery, 2018, pp. 106–110, ISBN: 9781450364591. DOI: 10.1145/3202710.3203160. [Online]. Available: <https://doi.org/10.1145/3202710.3203160>.
- [13] H. Phan and A. Jannesari, *Story point effort estimation by text level graph neural network*, Mar. 2022.
- [14] B. Kumar, U. K. Tiwari, D. C. Dobhal, and H. S. Negi, “User story clustering using k-means algorithm in agile requirement engineering,” in *2022 International Conference on Computational Intelligence and Sustainable Engineering Solutions (CISES)*, 2022, pp. 1–5. DOI: 10.1109/CISES54857.2022.9844390.
- [15] V. Tawosi, A. Al-Subaihin, and F. Sarro, “Investigating the effectiveness of clustering for story point estimation,” in *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2022, pp. 827–838. DOI: 10.1109/SANER53432.2022.00101.
- [16] P. E. Love, W. Fang, J. Matthews, S. Porter, H. Luo, and L. Ding, *Explainable artificial intelligence: Precepts, methods, and opportunities for research in construction*, 2023. arXiv: 2211.06579 [cs.AI].
- [17] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu, “Explainable ai: A brief survey on history, research areas, approaches and challenges,” in Sep. 2019, pp. 563–574, ISBN: 978-3-030-32235-9. DOI: 10.1007/978-3-030-32236-6_51.
- [18] T. Schlippe, Q. Stierstorfer, M. t. Koppel, and P. Libbrecht, “Explainability in automatic short answer grading,” in *International Conference on Artificial Intelligence in Education Technology*, Springer, 2022, pp. 69–87.
- [19] A. Filighera, S. Parihar, T. Steuer, T. Meuser, and S. Ochs, “Your answer is incorrect... would you like to know why? introducing a bilingual short answer feedback dataset,” in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, S. Muresan, P. Nakov, and A. Villavicencio, Eds., Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 8577–8591. DOI: 10.18653/v1/2022.acl-long.587. [Online]. Available: <https://aclanthology.org/2022.acl-long.587>.
- [20] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, “Explainable ai: A review of machine learning interpretability methods,” *Entropy*, vol. 23, no. 1, 2021, ISSN: 1099-4300. DOI: 10.3390/e23010018. [Online]. Available: <https://www.mdpi.com/1099-4300/23/1/18>.
- [21] H. T. T. Nguyen, H. Q. Cao, K. V. T. Nguyen, and N. D. K. Pham, “Evaluation of explainable artificial intelligence: Shap, lime, and cam,” in *Proceedings of the FPT AI Conference*, 2021, pp. 1–6.

- [22] K. Han, A. Xiao, E. Wu, J. Guo, C. Xu, and Y. Wang, “Transformer in transformer,” *Advances in neural information processing systems*, vol. 34, pp. 15 908–15 919, 2021.
- [23] M. Saunders, P. Lewis, A. Thornhill, and A. Bristow, ““research methods for business students” chapter 4: Understanding research philosophy and approaches to theory development,” in Mar. 2019, pp. 128–171, ISBN: 9781292208787.
- [24] A. Dresch, D. P. Lacerda, and J. A. Antunes Jr, *Design science research*, 2015. DOI: 10.1007/978-3-319-07374-3.
- [25] M. Choetkiertikul, H. K. Dam, T. Tran, T. Pham, A. K. Ghose, and T. Menzies, “A deep learning model for estimating story points,” *IEEE Transactions on Software Engineering*, vol. 45, pp. 637–656, 2016. [Online]. Available: <https://api.semanticscholar.org/CorpusID:15300052>.
- [26] V. Tawosi, A. Al-Subaihin, R. Moussa, and F. Sarro, *A versatile dataset of agile open source software projects*, 2022. arXiv: 2202.00979 [cs.SE].
- [27] G. Haixiang, Y. Li, J. Shang, G. Mingyun, H. Yuanyue, and B. Gong, “Learning from class-imbalanced data: Review of methods and applications,” *Expert Systems with Applications*, vol. 73, Dec. 2016. DOI: 10.1016/j.eswa.2016.12.035.
- [28] C. M. Greco, A. Tagarelli, and E. Zumpano, “A comparison of transformer-based language models on nlp benchmarks,” in *Natural Language Processing and Information Systems*, P. Rosso, V. Basile, R. Martínez, E. Métais, and F. Meziane, Eds., Cham: Springer International Publishing, 2022, pp. 490–501, ISBN: 978-3-031-08473-7.
- [29] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, 2019. arXiv: 1810.04805 [cs.CL].
- [30] N. Reimers and I. Gurevych, *Sentence-bert: Sentence embeddings using siamese bert-networks*, 2019. arXiv: 1908.10084 [cs.CL].
- [31] B. Li and L. Han, “Distance weighted cosine similarity measure for text classification,” in *Intelligent Data Engineering and Automated Learning–IDEAL 2013: 14th International Conference, IDEAL 2013, Hefei, China, October 20-23, 2013. Proceedings 14*, Springer, 2013, pp. 611–618.
- [32] J. Praveen Gujjar, H. R. Prasanna Kumar, and M. S. Guru Prasad, “Advanced nlp framework for text processing,” in *2023 6th International Conference on Information Systems and Computer Networks (ISCON)*, 2023, pp. 1–3. DOI: 10.1109/ISCON57294.2023.10112058.
- [33] K. Dawoud, W. Samek, P. Eisert, S. Lopuschkin, and S. Bosse, “Human-centered evaluation of xai methods,” Dec. 2023, pp. 912–921. DOI: 10.1109/ICDMW60847.2023.00122.

Appendix

1. The code for training the models and running the experiments can be found at [this GitHub repository](#).
2. The participant responses of the human-based evaluation are available at [this Google drive link](#).