Journal of Theoretical and Applied Information Technology
31st October 2024. Vol.102. No. 20
© Little Lion Scientific

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

MACHINE LEARNING FOR STORY POINT ESTIMATION: DO LARGE LANGUAGE MODELS OUTPERFORM TRADITIONAL METHODS?

Levi Alexander[1] , Riyanto Jayadi[2]

[1] Binus Graduate Program, Bina Nusantara University, Jakarta, Indonesia

[2] Binus Graduate Program, Bina Nusantara University, Jakarta, Indonesia

E-mail: [1]levi.alexander@binus.ac.id, [2]riyanto.jayadi@binus.edu

ABSTRACT

Our research investigates the performance of machine learning models, particularly Large Language Models (LLMs), in automating story point estimation for Agile software development. Traditional estimation methods relying on human judgment can introduce subjectivity and errors. Recent advances in deep learning and LLMs offer potential improvements in accuracy and consistency, especially in handling complex language tasks. We compare traditional machine learning models such as Random Forest, SVM, and Linear SVM with LLMs like BERT and GPT-2, focusing on both within-project and cross-project story point estimation. While traditional models frequently outperform LLMs in project-specific tasks, LLMs show competitive performance in handling more complex and diverse datasets. Our proposed general model, trained on combined datasets, demonstrates competitive results in structured cross-project estimation scenarios, narrowing the performance gap compared to previous models like Deep-SE and GPT2SP. However, project-specific models still outperform the general model in most cases. Our research highlights the trade-offs between model complexity and performance, showing that traditional models are often more efficient and accurate in structured datasets, whereas LLMs excel in tasks requiring deep language understanding. Further refinement of general models could enhance their applicability across diverse projects.

Keywords: *Deep Learning, Large Language Model, Machine Learning, Software Effort Estimation, Story Point Estimation*

1. INTRODUCTION

Agile software development has emerged as a widely adopted methodology in the industry, enabling companies to enhance their software development capabilities and respond more swiftly to dynamic market changes [1], [2]. A crucial aspect of the Agile process is accurate estimation, which helps teams and management identify potential risks, allocate resources effectively, and ensure projects are delivered within budget and on time [3], [4]. Despite these benefits, studies on large-scale IT projects reveal that 66% of projects exceed their budget, and 33% experience delays [5]. A separate study of 1,471 projects found that one in six projects exceeded their budget by up to 200%, with nearly 70% facing delays [6]. Story points have become a popular method for estimation in Agile development, capturing the effort, complexity, risks, dependencies, and unknowns associated with a user story [7]. Traditionally, these points are estimated manually by experienced team members or through collaborative methods like Planning Poker [8], [9]. However, these methods can be subjective, time-consuming, and prone to inaccuracies, potentially increasing project risks [4], [7], [10].

In recent years, researchers have explored Machine Learning (ML) techniques to automate story point estimation, aiming to reduce biases and improve accuracy [3], [4], [8], [10], [11], [12], [13], [14]. Approaches like Deep-SE by Choetkiertikul et al. [3], which employs LSTM-based deep learning, and GPT2SP by Fu and Tantithamthavorn [8], based on GPT-2, have made strides in automating this task. However, they often face challenges in cross-project estimation, where the variability of textual descriptions across different projects leads to inconsistent results, particularly when evaluated using metrics like Mean Absolute Error (MAE), Median Absolute Error (MdAE), and Standardized Accuracy (SA) [3], [8], [13].

The advent of Transformer-based models, such as GPT-2 and BERT, has dramatically advanced the field of Natural Language Processing (NLP), enabling models to excel across multiple disciplines—ranging from legal to finance, and programming—with strong performance in tasks that require understanding and processing complex textual data [15], [16]. These models are highly generalizable and have inspired the development of more general models capable of working across diverse domains. Drawing inspiration from these advancements, this paper explores whether a general model, trained on combined datasets from different projects, can provide consistent and accurate predictions across various Agile software development projects. By leveraging a general model approach, the goal is to address cross-project estimation challenges and enhance performance consistency.

Cross-project estimation remains difficult due to the varying nature of textual descriptions, issue contexts, and project-specific factors, which complicate generalization across datasets. This variability poses a significant hurdle for machine learning models, which often perform well within individual projects but struggle to transfer that performance across different projects.

Additionally, we investigate the effectiveness of LLMs like BERT and GPT-2 in comparison to traditional machine learning approaches. By evaluating their performance on both within-project and cross-project estimation tasks, this paper seeks to assess the feasibility of applying LLMs and general models to automate story point estimation and improve consistency.

1.1 Problem Statement

Accurate story point estimation is a critical aspect of Agile software development, helping teams plan and allocate resources effectively. However, traditional estimation methods are often subjective and prone to errors, introducing risks into the development process. Machine learning (ML) techniques have been explored to automate this process, but a key challenge remains—cross-project estimation, where models struggle to generalize across diverse datasets from different projects. This variability in textual descriptions and project-specific factors makes it difficult for machine learning models to maintain consistent performance across projects.

Furthermore, despite the growing use of Large Language Models (LLMs) in various Natural Language Processing (NLP) tasks, LLMs have not been widely adopted in the field of story point estimation. Traditional models like Support Vector Machines (SVM) continue to dominate research in this area, while the potential of LLMs such as BERT remains largely unexplored. This gap presents an opportunity to investigate whether LLMs, with their ability to process complex language patterns, can address the limitations faced by traditional models, particularly in the context of cross-project estimation.

The goal of this research is to address these challenges by investigating the performance of LLMs in story point estimation and exploring whether a general model trained on combined datasets can offer a viable solution for cross-project estimation. By doing so, this study aims to contribute new insights into the applicability of LLMs and general models in Agile development.

1.2 Research Questions and Results

(RQ1) Are general model approaches better compared to previous research in cross-project estimation?

Results. The general model approach does not outperform previous research in most cross-project estimation tasks, though it shows potential in certain datasets, such as Mesos and Spring XD.

(RQ2) Can a general model trained across multiple datasets achieve competitive accuracy compared to project-specific models in story point estimation?

Results. Project-specific models consistently outperform the general model in within-project estimation, though the general model demonstrates reasonable performance in some cases.

(RQ3) How do LLMs perform, and what is their contribution to story point estimation?

Results. LLMs like BERT and GPT-2 offer advantages in handling domain-specific text, but traditional models often provide better or comparable accuracy with fewer computational resources in structured datasets.

1.3 Contributions

This paper makes the following key contributions:

- A novel approach to story point estimation by training a general model across combined datasets, offering insights into cross-project estimation potential.
- A comprehensive comparison of LLMs (BERT and GPT-2) with traditional models, focusing on their performance in both within-project and cross-project estimation.
- An analysis of the trade-offs between model complexity and performance, with practical recommendations for balancing accuracy and computational resources in real-world software development scenarios.

2. RELATED WORK

Research on story point estimation based on textual descriptions has been ongoing for over a decade [10]. A significant contribution in this area is the work by Choetkiertikul et al. [3], who introduced a model that leverages two deep learning architectures: Long Short-Term Memory (LSTM) and Recurrent Highway Networks (RHWN). In their approach, Choetkiertikul et al. first generated vector representations of each word in the text, which were then fed into an LSTM layer to obtain a vector representation of the entire document. This document vector was subsequently processed through RHWN layers for multiple transformations before being fed into a regression layer that predicted the story point value for the issue. Choetkiertikul et al. model was trained and evaluated on a dataset of 23,313 issues from 16 open-source projects, demonstrating significant performance improvements over baseline estimators [3]. This model achieved an average Mean Absolute Error (MAE) of 2.08, outperforming other machine learning-based approaches such as LSTM+RF, BoW+RF, Doc2Vec+RF, and TFIDF+SVM [3].

This work has served as a state-of-the-art reference and a point of comparison for subsequent research efforts in the domain [4], [8], [10], [11], [13], [17]. Researchers frequently benchmark their models against Choetkiertikul et al. model due to its strong performance and the robustness of its approach, solidifying its position as a leading model in story point estimation based on textual data. For instance, Fu and Tantithamthavorn [8] introduced GPT2SP, a story point estimation model based on the Transformer-based deep learning architecture GPT-2, which was evaluated on the same dataset used by Choetkiertikul et al. model [3]. Following the corrections proposed by Tawosi et al. [13], [17], GPT2SP outperformed the median baseline and the Choetkiertikul et al. model in only 6 out of 16 within-project scenarios. However, these improvements were statistically significant in just three cases: two against the median baseline (with negligible and small effect sizes) and two against the Choetkiertikul et al. model (both with negligible effect sizes) [13], [17]. Despite these mixed results, the continuous comparison with Choetkiertikul et al.'s work highlights the ongoing relevance and impact of their model in the field of story point estimation.

3. APPROACH

3.1 Data Preparation

This study investigates the use of both traditional machine learning models and Large Language Models (LLMs) for story point estimation. The input text consists of the title and description of issues, which are concatenated into a single text document. This combined text is used to train various models, including Support Vector Machines (SVM), Linear SVM, Random Forest, BERT, and GPT-2.

To prepare the data, we concatenate the title and description of each issue into a single column called "text" and convert the "story point" into a numerical "label." Table 1 and Table 2 below provide examples of how the issue data is processed:

Table 1: Example of Issue Data Preprocessing

issue_key	title	description	story_point
XD-1	HDFS ItemWriter	Base integration of core HDFS writer functionality with Spring Batch.	1
XD-2	HDFS Core writing helper classes	Simple file writer that has existed in the spring hadoop samples.	1
XD-6	Channel Registry		3

Table 2: Combined Text and Label Representation for Model Training

text	label

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

HDFS ItemWriter Base integration of core HDFS writer functionality with Spring Batch.	1.0
HDFS Core writing helper classes Simple file writer that has existed in the spring hadoop samples.	1.0
Channel Registry	3.0

3.2 Model Selection

The models selected for this study—Support Vector Machines (SVM), Linear SVM, Random Forest, BERT, and GPT-2—are based on those used in previous research [3], [8]. SVM and GPT-2 were specifically chosen because they have been demonstrated to be effective in prior studies on story point estimation, providing a solid baseline for comparison. Linear SVM was included to offer an additional point of comparison among traditional machine learning models.

To ensure a comprehensive evaluation, we also included BERT, which has shown state-of-the-art performance in various NLP tasks [18], and Random Forest, known for its robustness in traditional machine learning tasks. By incorporating both traditional models and more advanced LLMs, we aimed to compare a diverse range of algorithms while maintaining simplicity. This approach allows us to analyze the performance of newer LLMs against more established models like SVM, Linear SVM, and Random Forest, particularly in the context of story point estimation tasks where model complexity may not always lead to better accuracy.

Unlike prior work that utilized more advanced methods like LSTM-based models [3], our approach emphasizes simplicity, relying on standard techniques such as Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) for feature extraction [19]. This allows us to establish a baseline comparison that is not influenced by enhancement methods or model complexity, thus ensuring unbiased results.

3.3 Feature Extraction and Embedding

For traditional models like SVM, Linear SVM, and Random Forest, text feature extraction is performed using Bag of Words (BoW) or Term Frequency-Inverse Document Frequency (TF-IDF) methods [19]. These numerical representations of words are used to train the models in a regression task.

On the other hand, LLMs such as BERT and GPT-2 use tokenization methods like Byte-Pair Encoding (BPE) for GPT-2 [20] and WordPiece for BERT [18], which break the text into subword units to better capture semantic meaning. The word embeddings generated by these methods are then used as inputs for the transformer-based models.

After preprocessing the text and label columns, the next step involves transforming the text into numerical representations that can be used by the models. The following diagram shows how both traditional and transformer models handle this embedding process:

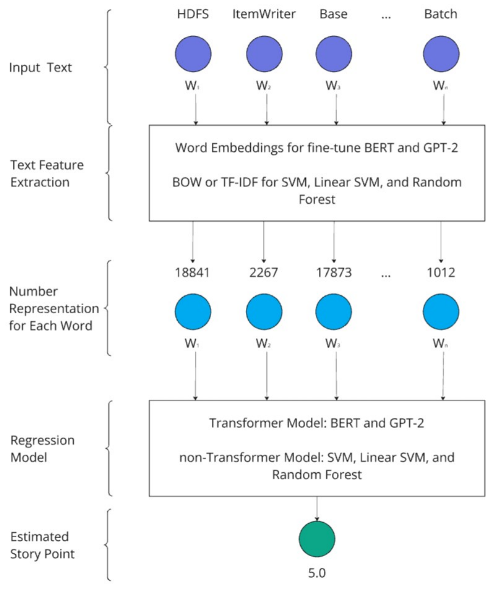
Input Text
HDFS
ItemWriter
Base
...
Batch
W1
W2
W3
Wn
Text Feature Extraction
Word Embeddings for fine-tune BERT and GPT-2
BOW or TF-IDF for SVM, Linear SVM, and Random Forest
Number Representation for Each Word
18841
2267
17873
...
1012
Regression Model
Transformer Model: BERT and GPT-2
non-Transformer Model: SVM, Linear SVM, and Random Forest
Estimated Story Point
5.0

Figure 1: Embedding Process

4. EXPERIMENTS

4.1 Dataset

The dataset used in this study was sourced from Choetkiertikul et al.'s research [3], which included data from 16 open-source projects hosted on 9 major repositories that use the JIRA issue tracking system. These projects, such as Apache Mesos, Apache Usergrid, and JIRA Software, were chosen because they actively use JIRA for issue tracking and assign story points to their issues, making them suitable for story point estimation research [3], [8].

The dataset underwent preprocessing, as previously described in Section 3, to remove issues with zero, negative, or unrealistically high story points (e.g., values greater than 100) [8].

4.2 Data Splitting and Setup

Once preprocessed, the dataset was split into three subsets: 60% for training, 20% for validation, and 20% for evaluation. This 60-20-20 split is commonly used in previous research on story point estimation and similar tasks [3], [8], as it provides a balanced distribution of data for model training and evaluation. Following this well-established practice ensures that our approach is consistent with existing studies and allows for reliable comparisons of model performance.

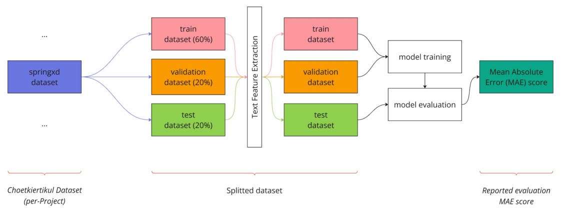

Figure 2: Model Training Process for Each Dataset

4.3 Project-Specific Model Training and Evaluation

For each individual project dataset, we trained models using 60% of the data for training, 20% for validation, and 20% for testing. The dataset split ensures a balanced approach for evaluating model generalizability across unseen data. Figure 2 presents the workflow for training and evaluating the models on each project's dataset.

The algorithms applied to each project dataset include Support Vector Machines (SVM), Linear SVM, Random Forest, BERT, and GPT-2. We employed Bag of Words (BoW) and Term Frequency-Inverse Document Frequency (TF-IDF) techniques to vectorize the text data for traditional machine learning models, while transformer models like BERT and GPT-2 used advanced tokenization techniques such as WordPiece and Byte-Pair Encoding (BPE) to process the text.

For evaluation, we used the Mean Absolute Error (MAE) metric, which is widely recognized for its robustness against outliers and its interpretability in regression tasks [8], [21]. MAE allows us to measure the average magnitude of errors between predicted and actual story points, making it an appropriate metric for story point estimation tasks.

Training times for these models varied between 5 seconds and 20 minutes per project, depending on the dataset size, model complexity, and training hyperparameters. The project-specific models provided valuable insights into how well machine learning algorithms can perform when tailored to specific datasets. In most cases, the traditional models (SVM, Linear SVM, Random Forest) showed competitive performance, while BERT and GPT-2 occasionally demonstrated superior results on larger, more complex datasets.

4.4 General Model Training and Evaluation

In addition to training models on individual project datasets, we also trained a general model using a combined dataset sourced from multiple projects. The general model aims to capture the broader trends in story point estimation by learning from a more diverse range of issues across various projects. This combined dataset was created by merging the preprocessed datasets from different projects, resulting in a larger, more complex dataset for training.

The same algorithms (SVM, Random Forest, Linear SVM, BERT, and GPT-2) were applied to the combined dataset. Training the general model was computationally more intensive, requiring significantly longer training times, with training durations extending up to 50 minutes depending on the model and the size of the combined dataset. Despite the increased computational cost, the general model's performance did not consistently exceed that of project-specific models.

The evaluation of the general model was conducted using the Mean Absolute Error (MAE) metric, as discussed earlier. The general model provides insights into how well machine learning models can generalize across projects. However, a clear trade-off between training time and model performance was observed, with specialized models often outperforming the general model in project-specific tasks.

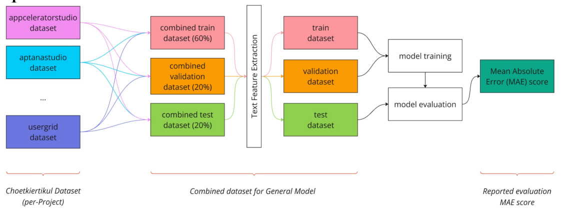
appceleratorstudio dataset
aptanastudio dataset
...
usergrid dataset
combined train dataset (60%)
combined validation dataset (20%)
combined test dataset (20%)
Text Feature Extraction
train dataset
validation dataset
test dataset
model training
model evaluation
Mean Absolute Error (MAE) score
Choetkiertikul Dataset (per-Project)
Combined dataset for General Model
Reported evaluation MAE score

Figure 3: General Model Training Process

5. RESULTS

5.1 Comparison of Large Language Models (LLMs) with Non-Transformer Models

Table 3 provides the Mean Absolute Error (MAE) scores for each model across multiple project datasets and the general model. It highlights the performance of the LLMs (BERT, GPT-2) compared to traditional machine learning models (Random Forest, SVM, Linear SVM).

While BERT consistently performs well across many project-specific datasets, traditional models like SVM and Random Forest also show competitive results. For example, in the Appcelerator Studio (AS) project, BERT achieved an MAE of 0.90167, but SVM outperformed it with an MAE of 0.83396. Similarly, in the Bamboo (BB) project, Random Forest outperformed BERT, achieving an MAE of 0.82604 compared to BERT's 0.86663.

These findings demonstrate that while LLMs such as BERT offer strong performance on some datasets, traditional models still hold their own in many cases, particularly on less complex or more structured datasets. This highlights the robustness of traditional machine learning models in certain scenarios where the complexity of deep learning models may not always translate into better accuracy.

Table 3: Evaluation results for each model (the best results are highlighted in bold). MAE - the lower the better

No	Project	Mean Absolute Error (MAE) Score					
		General Model	BERT	GPT-2	Random Forest	SVM	Linear SVM
1	Combined	-	**1.15622**	1.18158	1.20868	1.21938	1.23672
2	Appcelerator Studio (AS)	2.10492	0.90167	0.90218	0.84241	**0.83396**	0.85715
3	Aptana Studio (AP)	2.66759	1.29021	1.50538	**1.19278**	1.19791	1.19764
4	Bamboo (BB)	0.90021	0.86663	0.99213	**0.82604**	0.86352	0.89092
5	Clover (CV)	1.50203	1.49848	3.34625	1.34066	**1.31851**	1.32157
6	Data Management (DM)	1.37032	1.37559	1.43915	1.32342	1.3147	**1.2987**
7	Dura Cloud (DC)	0.87982	0.75204	6.66794	0.76476	0.75748	**0.74962**
8	Jira Software (JI)	1.58940	1.69684	5.30661	1.15808	1.12001	**1.07344**
9	Mesos (ME)	1.15039	**0.92766**	1.17247	0.95917	0.95807	0.95838
10	Moodle (MD)	3.12658	1.81927	2.01619	1.48347	1.43755	**1.39205**
11	Mule (MU)	1.92389	1.26681	2.80438	1.24435	**1.20406**	1.25505
12	Mule Studio (MS)	2.02571	1.12415	1.21327	1.09643	**1.07578**	1.13761
13	Spring XD (XD)	1.28722	**1.01445**	1.09437	1.02885	1.05952	1.0443
14	Talend Data Quality (TD)	2.00528	1.36755	2.76225	**1.29484**	1.3124	1.34175
15	Talend ESB (TE)	0.86248	**0.72574**	0.91707	0.7779	0.77502	0.80237
16	Titanium (TI)	1.96481	0.97526	1.08559	**0.96789**	1.00033	1.04129
17	Usergrid (UG)	1.07959	0.76829	1.34775	0.78837	**0.70664**	0.83912

5.2 Performance of the General Model

The general model was trained on a combined dataset to provide a solution applicable across multiple projects. However, the results show that the general model's performance is generally lower than project-specific models, with varying degrees of closeness depending on the dataset.

For instance, in the Mesos (ME) project, the general model achieved an MAE of 1.15039, which is relatively close to the best-performing project-specific model (BERT, with an MAE of 0.92766). Similarly, in the Spring XD (XD) project, the general model's MAE of 1.28722 is only slightly higher than BERT's MAE of 1.01445. This suggests that, for certain projects with structured data and fewer outliers, the general model can approximate the performance of specialized models.

However, in other projects, the general model's results are significantly further from the best-performing models. For example, in the Aptana Studio (AP) project, the general model's MAE of 2.66759 is much higher than the best project-specific model (Random Forest, with an MAE of 1.19278). A similar trend is observed in Appcelerator Studio (AS), where the general model scored 2.10492, while SVM achieved 0.83396. This indicates that, for more complex or varied datasets, the general model struggles to capture the nuances that project-specific models can.

The Training Loss and Validation Loss curves for the general model, as shown in Figure 1, suggest that the model was learning effectively from the combined dataset. The training loss consistently decreases, indicating that the model fits well to the training data. However, the validation loss stabilizes at a higher value, which could indicate that the model has learned patterns specific to the training data but struggles to generalize across different projects within the combined dataset. This divergence between training and validation performance may suggest some degree of overfitting, where the model captures noise or project-specific details that do not generalize well to unseen projects.

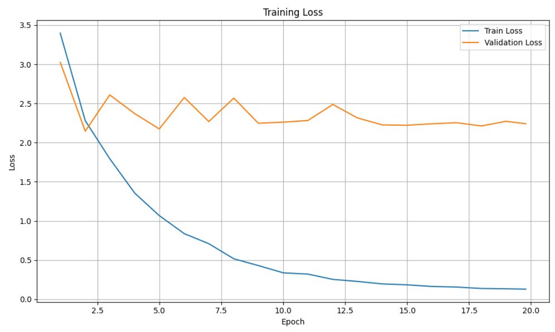

Figure 4: General Model BERT Training and Validation Loss

Similarly, the Mean Absolute Error (MAE) Validation Performance curve for the general model (shown in Figure 2) shows some fluctuation during the training process but ultimately stabilizes. The final validation MAE hovers around 1.15, which is consistent with the general model's overall MAE across different projects, as shown in Table 3. This further confirms that while the general model is capable of capturing some trends across the combined dataset, it cannot consistently match the accuracy of models trained on specific projects.

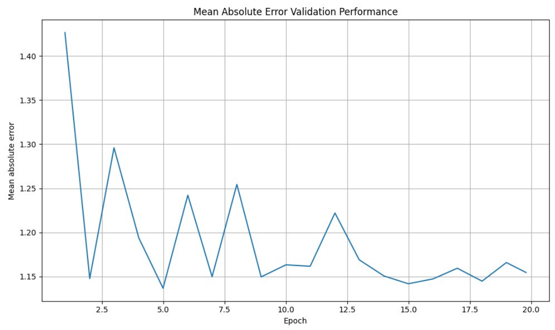

Figure 5: General Model BERT MAE Validation Performance

Overall, the general model demonstrates varying levels of accuracy depending on the characteristics of the dataset. It performs relatively well in more structured datasets, such as Mesos and Spring XD, where the data is consistent and less complex, achieving results that are closer to project-specific models. However, in more diverse or complex datasets, such as Aptana Studio and Appcelerator Studio, the general model shows a significant gap compared to project-specific models. This indicates that while the general model can effectively capture patterns in simpler, more uniform datasets, it struggles to generalize across more complex datasets with greater variability.

The training and validation performance suggest that the general model learns effectively from the combined dataset, but its ability to generalize to different projects within the dataset is limited. In projects with high variability or complexity, the model tends to overfit to the training data, which is reflected in the higher validation loss and MAE scores.

5.3 Comparison with Previous Research for Within-Project Estimation

Table 4 presents a comparison of both the general model and the best-performing models from this study against previous research, including Deep-SE by Choetkiertikul et al. [3] and GPT2SP by Fu and Tantithamthavorn [8]. Due to miscalculations highlighted in GPT2SP original paper, the comparison uses recalculated values from Tawosi et al.'s replication study [17].

When comparing the general model to Deep-SE, GPT2SP, and the mean/median values from Tawosi et al. replication study [17], the results align with the findings from Section 5.2. The general model, while not consistently outperforming project-spesific models, demonstrates competitive results that are often within a reasonable range. For instance, in Appcelerator Studio (AS), the general model's MAE of 2.10 is higher than both Deep-SE (1.42) and GPT2SP (1.53), but it is still within a comparable range when considering the mean (1.91) and median (1.30) values. In Aptana Studio (AP), the general model's MAE of 2.67 is higher than Deep-SE (4.14) and GPT2SP (3.52), but close to the mean (3.59) and median (3.61) values.

In more structured datasets, such as Usergrid (UG), the general model shows even closer results, with an MAE of 1.08, which is very close to Deep-SE's 1.18, GPT2SP's 1.19, and also aligns closely with both the mean (1.19) and median (1.15). This suggests that while the general model does not always surpass project-specific models or previous approaches, it remains competitive across different types of datasets, particularly in more structured environments.

Table 4: Within-Project Estimation Comparison with Previous Researches

No	Project	Mean Absolute Error (MAE) Score					
		General Model	This Paper Best Score	Deep-SE	GPT2SP	Mean	Median
1	Appcelerator Studio (AS)	2.10	0.83	1.42	1.53	1.91	1.30
2	Aptana Studio (AP)	2.67	1.19	4.14	3.52	3.59	3.61
3	Bamboo (BB)	0.90	0.83	0.81	0.77	1.22	0.75
4	Clover (CV)	1.50	1.32	3.39	3.76	4.57	3.71
5	Data Management (DM)	1.37	1.30	5.86	5.39	8.66	6.19
6	Dura Cloud (DC)	0.88	0.75	0.82	0.80	1.00	0.82
7	Jira Software (JI)	1.59	1.07	1.70	1.57	2.40	2.31
8	Mesos (ME)	1.15	0.93	1.12	1.21	1.41	1.22
9	Moodle (MD)	3.13	1.39	7.89	8.38	12.63	6.59
10	Mule (MU)	1.92	1.20	2.59	2.61	2.60	2.47
11	Mule Studio (MS)	2.03	1.08	3.67	3.70	3.74	3.66
12	Spring XD (XD)	1.29	1.01	1.70	1.78	2.05	1.71
13	Talend Data Quality (TD)	2.01	1.29	3.61	3.65	4.56	3.31
14	Talend ESB (TE)	0.86	0.73	0.90	0.86	1.04	0.92
15	Titanium (TI)	1.96	0.97	2.09	2.35	3.02	2.04
16	Usergrid (UG)	1.08	0.71	1.18	1.19	1.19	1.15

Abbreviations: Deep-SE = model from Choetkiertikul et al. study [3], GPT2SP = model from Fu and Tantithamthavorn [8].

The best-performing project-specific models from this study often outperform previous research. For example, in Appcelerator Studio (AS), the best-performing model (SVM) achieved an MAE of 0.83, which outperforms both Deep-SE (1.42) and GPT2SP (1.53). Similarly, in Aptana Studio (AP), the best-performing model (Random Forest) achieved an MAE of 1.19, better than Deep-SE's 4.14 and GPT2SP's 3.52.

In more complex datasets such as Data Management (DM), the best models from this study also showed strong results. The MAE of 1.30 for the best-performing model significantly outperformed both Deep-SE (5.86) and GPT2SP (5.39). This trend of outperforming previous research continues in other challenging datasets like Moodle (MD), where the best model from this study achieved an MAE of 1.39, far better than both Deep-SE (7.89) and GPT2SP (8.38).

5.4 Comparison with Previous Research for Cross-Project Estimation

Cross-project estimation has long posed challenges for machine learning models due to the variability in data across different repositories. Unlike within-project estimation, where models are trained and tested on data from the same project, cross-project estimation evaluates how well a model generalizes across entirely different projects. This task is crucial for real-world applications, where maintaining multiple project-specific models is often impractical. A general model capable of accurate cross-project estimation offers a practical solution to reduce overhead.

The general model consistently outperforms Deep-SE and GPT2SP in several projects, as shown in Table 5, across both within-repository and cross-repository training. For instance, in Mesos (ME), the general model achieves an MAE of 1.15, better than Deep-SE (1.51 and 3.18) and GPT2SP (1.52 and 2.65) across both training scenarios. Similarly, in Usergrid (UG), the general model achieves an MAE of 1.08, surpassing both Deep-SE (1.16 and 3.47) and GPT2SP (1.11 and 2.18). These results demonstrate that the general model performs well even in scenarios where specialized models from previous research were less effective.

Table 5: Cross-Project Estimation Comparison with Previous Researches

No	Project	Mean Absolute Error (MAE) Score

		General Model	Within-Repositories				Cross-Repositories			
			Deep-SE	GPT2 SP	Mean	Median	Deep-SE	GPT2 SP	Mean	Median
1	Appcelerator Studio (AS)	2.10	2.70	2.86	3.38	3.17	2.64	2.16	11.45	3.17
2	Aptana Studio (AP)	2.67	4.37	4.62	4.27	4.38	5.03	4.37	9.84	3.97
3	Mesos (ME)	1.15	1.51	1.52	1.52	1.50	3.18	2.65	3.28	2.58
4	Mule (MU)	1.92	2.77	3.03	3.05	2.60	3.20	3.13	2.89	2.92
5	Mule Studio (MS)	2.03	3.64	3.48	3.34	3.26	3.95	4.02	4.04	3.91
6	Titanium (TI)	1.96	3.28	3.28	3.45	3.17	3.34	3.60	11.19	4.19
7	Titanium (TI)	1.96	3.51	4.09	4.36	4.19	3.81	3.58	5.61	3.46
8	Usergrid (UG)	1.08	1.16	1.11	1.02	0.89	3.47	2.18	3.08	2.30

Abbreviations: Deep-SE = model from Choetkiertikul et al. study [3], GPT2SP = model from Fu and Tantithamthavorn [8].

In some projects, the general model's MAE aligns closely with the mean and median values, reinforcing its competitiveness. For example, in Usergrid (UG), the general model's MAE of 1.08 is close to the mean (1.02) and median (0.89), indicating that its performance is on par with broader estimates. In more complex datasets like Aptana Studio (AP), while the general model performs better than Deep-SE and GPT2SP, its MAE of 2.67 remains close to the median (3.97), indicating that it may still encounter some challenges in highly complex datasets.

6. DISCUSSION

6.1 RQ1: General Models and Cross-Project Estimation

The general model, trained across combined datasets, demonstrated strong performance in cross-project estimation, consistently outperforming previous research models such as Deep-SE and GPT2SP. For example, in the Appcelerator Studio (AS) dataset, the general model achieved an MAE of 2.10, outperforming Deep-SE (2.70), GPT2SP (2.86), and baseline values like the mean (3.38) and median (3.17). Similarly, in Aptana Studio (AP), the general model's performance (MAE of 2.67) was better than Deep-SE (4.37) and GPT2SP (4.62).

In the Mesos (ME) dataset, the general model achieved an MAE of 1.15, better than Deep-SE (1.51) and GPT2SP (1.52). This result shows the general model's capability to handle cross-project estimation challenges effectively, maintaining a competitive edge over previous models. Even in datasets like Mule Studio (MS) and Titanium (TI), where the general model's performance was lower than in other datasets, it still outperformed Deep-SE, GPT2SP, and the baseline mean and median. For example, in Mule Studio (MS), the general model's MAE of 2.03 was better than Deep-SE (3.64), GPT2SP (3.48), and the mean (3.34).

This consistent performance demonstrates that the general model, trained across combined datasets, can handle the variability of cross-project estimation effectively, often outperforming models that were evaluated using both within-repository and cross-repository approaches in previous research. These results directly address RQ1, showing that the general model offers a strong, versatile solution for cross-project estimation compared to previous models.

6.2 RQ2: General Model vs. Project-Specific Models in Story Point Estimation

Previous research consistently showed that project-specific models outperformed cross-project models due to their ability to fine-tune on a single dataset. This trend continues with project-specific models outperforming the general model in nearly all datasets. However, the general model narrows the gap compared to previous research models like Deep-SE and GPT2SP. For example, in the Data Management (DM) dataset, the general model achieved an MAE of 1.37, which is close to the best project-specific model's 1.30, whereas Deep-SE (5.86) and GPT2SP (5.39) had significantly higher errors. Similarly, in Usergrid (UG), the general model's MAE of 1.08 outperformed Deep-SE (1.18) and GPT2SP (1.19), but it still did not reach the project-specific model's 0.71. In Clover (CV), the general model's MAE of 1.50 was closer to the best project-specific score of 1.32, outperforming Deep-SE (3.39) and GPT2SP (3.76).

In some cases, the general model not only did not surpass project-specific models but also did not outperform the baseline metrics such as the mean and median. For instance, in the Appcelerator Studio (AS) dataset, the general model's MAE of 2.10 was higher than the mean (1.91) and median (1.30), despite outperforming Deep-SE (1.42) and GPT2SP (1.53). In Bamboo (BB) and Dura Cloud (DC), the general model similarly outperformed Deep-SE and GPT2SP, but it did not surpass the mean or median values.

A closer look at the best MAE score on project-specific models, which are mostly trained using SVM, Random Forest, and Linear SVM, shows that they often deliver competitive results across multiple datasets. For example, in Appcelerator Studio (AS), the project-specific model trained with SVM achieved an MAE of 0.83, outperforming more complex models like BERT (0.90) and GPT-2 (0.90). In Data Management (DM), the project-specific model trained with Random Forest slightly outperformed the general model with an MAE of 1.32, compared to the general model's 1.37.

These results indicate that while the general model has not surpassed project-specific models in within-project estimation, it demonstrates reasonable performance in several cases. In datasets like Data Management (DM), Usergrid (UG), and Clover (CV), the general model came close to the best project-specific models, although it consistently lags behind them overall. Additionally, the general model sometimes falls behind simple baseline metrics like the mean and median. However, it significantly narrows the performance gap compared to Deep-SE and GPT2SP, showing that the general model can offer competitive accuracy in some cases. These findings support RQ2, demonstrating that while project-specific models still provide the best accuracy, the general model offers a viable alternative and shows improvement over previous cross-project models.

6.3 RQ3: LLM Performance and Contribution to Story Point Estimation

While previous research on story point estimation introduced novel approaches like LSTM+RHW from Choetkiertikul, recent advancements in Large Language Models (LLMs) such as BERT and GPT-2 offer new possibilities for language understanding tasks. However, when evaluated in the specific context of story point estimation, LLMs do not consistently outperform traditional models such as SVM, Linear SVM, and Random Forest across various datasets.

For example, in the Appcelerator Studio (AS) dataset, BERT achieved an MAE of 0.90, but it was outperformed by SVM (0.83), Random Forest (0.84), and Linear SVM (0.86). Similarly, in the Data Management (DM) dataset, BERT had an MAE of 1.37, but traditional models like Random Forest (1.32), SVM (1.31), and Linear SVM (1.30) demonstrated slightly better performance.

However, LLMs like BERT and GPT-2 shine in tasks that require more complex language understanding and retrieval, such as Retrieval-Augmented Generation (RAG) implementations [22]. In RAG, LLMs excel by leveraging their ability to generate meaningful responses based on retrieved knowledge, an area where traditional models cannot compete. This highlights the strengths of LLMs in knowledge-based tasks that require deep text comprehension.

While LLMs offer advantages in specific use cases like RAG, their performance in story point estimation does not exceed that of traditional models. The results of this study suggest that for story point estimation, traditional models like SVM, Linear SVM, and Random Forest continue to provide more effective solutions, with LLMs contributing valuable insights in other areas.

These findings provide a detailed answer to RQ3, demonstrating that while LLMs have potential in handling more complex language tasks, their current application in story point estimation does not outperform traditional models. Therefore, LLMs are better suited for tasks like RAG, while non-transformer models remain the more practical choice for story point estimation.

6.4 Trade-offs Between Model Complexity and Performance

The decision between using LLMs such as BERT and GPT-2 versus traditional models like SVM, Linear SVM, and Random Forest for story point estimation involves both performance and computational costs. LLMs, while offering advanced capabilities, require significantly more resources to train compared to traditional models.

For non-transformer project-specific models, training times are highly efficient, ranging from as little as 1 second (with an approximate cost of \$0.00025) to 5 minutes (costing about \$0.075). Transformer-based project-specific models, such as BERT and GPT-2, typically require between 5 minutes (costing \$0.255) and 30 minutes (costing \$1.53) using GPU-based instances.

When it comes to general models, non-transformer versions can be trained in as little as 5 seconds (costing \$0.00125) up to 1 hour (costing \$0.90). Meanwhile, transformer-based general models require more extensive resources, with training times ranging from 10 minutes (costing \$0.51) to 1 hour (costing \$3.06).

These figures illustrate the stark differences in computational cost between traditional models and LLMs. LLMs require more expensive GPU-based resources and longer training times, while traditional models offer faster and cheaper training for story point estimation.

In tasks like story point estimation, where traditional models provide comparable accuracy, it may be more efficient to use non-transformer models like SVM, Linear SVM, and Random Forest. However, for more complex NLP tasks, such as Retrieval-Augmented Generation (RAG), where LLMs excel, the higher cost and longer training time can be justified.

6.5 Limitations and Assumptions

While this research provides new insights into story point estimation using both traditional models and Large Language Models (LLMs), several limitations and assumptions should be noted:

1. **Model Generalization**: The general model, while showing potential in cross-project estimation, underperforms in certain complex datasets such as Appcelerator Studio (AS) and Aptana Studio (AP). The variability in textual descriptions and project-specific contexts introduces challenges in generalization, leading to higher errors compared to project-specific models. This suggests that further refinement or additional feature engineering may be needed for the general model to handle more diverse datasets effectively.
2. **LLM Training Costs**: Transformer-based models such as BERT and GPT-2 require significantly more computational resources compared to traditional machine learning models. While LLMs performed well on certain datasets, their application to story point estimation did not consistently justify the higher computational costs. This limitation in scalability, particularly for smaller organizations with fewer resources, restricts the broader applicability of LLMs for this task.
3. **Dataset Bias**: The datasets used in this study, though sourced from diverse open-source projects, may not fully represent the variability of real-world Agile development environments. The selected projects may introduce biases that impact the generalizability of the findings to other domains or industries. Further experimentation with more varied datasets from different industries would provide a more comprehensive evaluation of the models' robustness.
4. **Unexplored Model Architectures**: While this study focused on a comparison between traditional machine learning models and transformer-based LLMs, other architectures, such as hybrid models combining LLMs with traditional regressors, were not explored. Such approaches may offer a promising direction for balancing model complexity and accuracy.

7. CONCLUSION

This study addressed the critical challenge of story point estimation in Agile software development, specifically focusing on cross-project estimation, which has historically been difficult due to the variability in project-specific data. By proposing a general model trained across combined datasets, this research sought to mitigate the challenges of cross-project estimation while also exploring the potential of Large Language Models (LLMs) like BERT and GPT-2. The results demonstrate that while the general model did not consistently outperform project-specific models, it significantly narrowed the performance gap compared to previous research in cross-project scenarios, particularly in datasets like Mesos and Spring XD. This finding highlights the general model's potential as a solution to cross-project estimation, though additional refinement is needed for handling more complex datasets.

Traditional machine learning models, such as SVM, Random Forest, and Linear SVM, consistently outperformed LLMs in structured, project-specific tasks, reaffirming their robustness and efficiency in contexts where model complexity does not necessarily translate into better performance. These results suggest that while LLMs offer strong performance in language understanding tasks, their application in story point estimation requires further exploration and optimization, particularly to justify the higher computational costs associated with training transformer-based models. The research also demonstrated that traditional models remain a more cost-effective solution for story point estimation tasks, where comparable or superior accuracy is achievable with significantly lower training times and costs.

In terms of limitations, the general model's underperformance in complex datasets like Appcelerator Studio (AS) and Aptana Studio (AP) reveals challenges in model generalization across highly variable project data. Further research is necessary to improve the model's ability to generalize and adapt to diverse textual descriptions and project-specific contexts. Additionally, the higher computational resources required for training LLMs compared to traditional models pose scalability limitations, particularly for smaller organizations with limited resources.

Future research should explore more modern LLMs, such as Llama 3, to assess their potential in story point estimation. A hybrid approach, utilizing LLMs for feature extraction and traditional models as regressors, may also offer an effective balance between model complexity and accuracy. By addressing these directions, it may be possible to further enhance the performance and applicability of machine learning models in story point estimation.

ACKNOWLEDGMENTS:

This research was supported by funding from the Directorate of Research, Technology, and Community Service, Ministry of Education, Culture, Research, and Technology, Republic of Indonesia, under grant number 0459/E5/PG.02.00/2024 30 May 2024.

REFERENCES:

[1] Digital AI, "16th State of Agile Report," https://digital.ai/. Accessed: Nov. 29, 2023. [Online]. Available: https://digital.ai/resource-center/analyst-reports/state-of-agile-report/

[2] Scrum Alliance, "The State of Scrum: 2017-2018 Report." Accessed: Nov. 29, 2023. [Online]. Available: https://resources.scrumalliance.org/Article/state-scrum-2017-2018-report

[3] M. Choetkiertikul, H. K. Dam, T. Tran, T. Pham, A. Ghose, and T. Menzies, "A Deep Learning Model for Estimating Story Points," *IEEE Trans. Softw. Eng.*, vol. 45, no. 7, pp. 637–656, Jul. 2019, doi: 10.1109/TSE.2018.2792473.

[4] M. Gultekin and O. Kalipsiz, "Story Point-Based Effort Estimation Model with Machine Learning Techniques," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 30, no. 01, pp. 43–66, Jan. 2020, doi: 10.1142/S0218194020500035.

[5] M. Bloch, S. Blumberg, and J. Laartz, "Delivering large-scale IT projects on time, on budget, and on value," 2012.

[6] B. Flyvbjerg and A. Budzier, "Why Your IT Project May Be Riskier than You Think," *SSRN Electron. J.*, 2011, doi: 10.2139/ssrn.2229735.

[7] M. Cohn, *Agile Estimating and Planning*. in Robert C. Martin series. Prentice Hall Professional Technical Reference, 2005. [Online]. Available: https://books.google.co.id/books?id=j0eFmAEACAAJ

[8] M. Fu and C. Tantithamthavorn, "GPT2SP: A Transformer-Based Agile Story Point Estimation Approach," *IEEE Trans. Softw. Eng.*, vol. 49, no. 2, pp. 611–625, Mar. 2022, doi: 10.1109/TSE.2022.3158252.

[9] M. Usman, E. Mendes, F. Weidt, and R. Britto, "Effort estimation in agile software development: a systematic literature review," in *Proceedings of the 10th International Conference on Predictive Models in Software Engineering*, in PROMISE '14. New York, NY, USA: Association for Computing Machinery, Sep. 2014, pp. 82–91. doi: 10.1145/2639490.2639503.

[10] 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)*, Mar. 2022, pp. 827–838. doi: 10.1109/SANER53432.2022.00101.

[11] 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)*, Sydney, Australia: IEEE, Nov. 2020, pp. 1–7. doi: 10.1109/CITISIA50690.2020.9371770.

[12] M. Turic, S. Celar, S. Dragicevic, and L. Vickovic, "Advanced Bayesian Network for Task Effort Estimation in Agile Software Development," *Appl. Sci.*, vol. 13, no. 16, Art. no. 16, Jan. 2023, doi: 10.3390/app13169465.

[13] V. Tawosi, R. Moussa, and F. Sarro, "Agile Effort Estimation: Have We Solved the Problem Yet? Insights From A Replication Study," *IEEE Trans. Softw. Eng.*, vol. 49, no. 4, pp. 2677–2697, Dec. 2022, doi: 10.1109/TSE.2022.3228739.

[14] V. Tawosi, S. Alamir, and X. Liu, *Search-based Optimisation of LLM Learning Shots for Story Point Estimation*. in International Symposium on Search Based Software Engineering. Springer Nature Switzerland, 2023. [Online]. Available: https://www.springerprofessional.de/en/search-based-optimisation-of-llm-learning-shots-for-story-point-/26487178

[15] OpenAI, "GPT-4 Technical Report," Mar. 27, 2023, *arXiv*: arXiv:2303.08774. doi: 10.48550/arXiv.2303.08774.

[16] A. Vaswani *et al.*, "Attention Is All You Need," Aug. 01, 2023, *arXiv*: arXiv:1706.03762. doi: 10.48550/arXiv.1706.03762.

[17] V. Tawosi, R. Moussa, and F. Sarro, "Agile Effort Estimation: Have We Solved the Problem Yet? Insights From A Second Replication Study (GPT2SP Replication Report)," Sep. 01, 2022, *arXiv*: arXiv:2209.00437. doi: 10.48550/arXiv.2209.00437.

[18] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," May 24, 2019, *arXiv*: arXiv:1810.04805. doi: 10.48550/arXiv.1810.04805.

[19] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," Jun. 05, 2018, *arXiv*: arXiv:1201.0490. doi: 10.48550/arXiv.1201.0490.

[20] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language Models are Unsupervised Multitask Learners," 2019.

[21] F. Sarro, A. Petrozziello, and M. Harman, "Multi-objective software effort estimation." Accessed: Aug. 10, 2024. [Online]. Available: https://dl.acm.org/doi/10.1145/2884781.2884830

[22] S. Siriwardhana, R. Weerasekera, E. Wen, T. Kaluarachchi, R. Rana, and S. Nanayakkara, "Improving the Domain Adaptation of Retrieval Augmented Generation (RAG) Models for Open Domain Question Answering," *Trans. Assoc. Comput. Linguist.*, vol. 11, pp. 1–17, Jan. 2023, doi: 10.1162/tacl_a_00530.