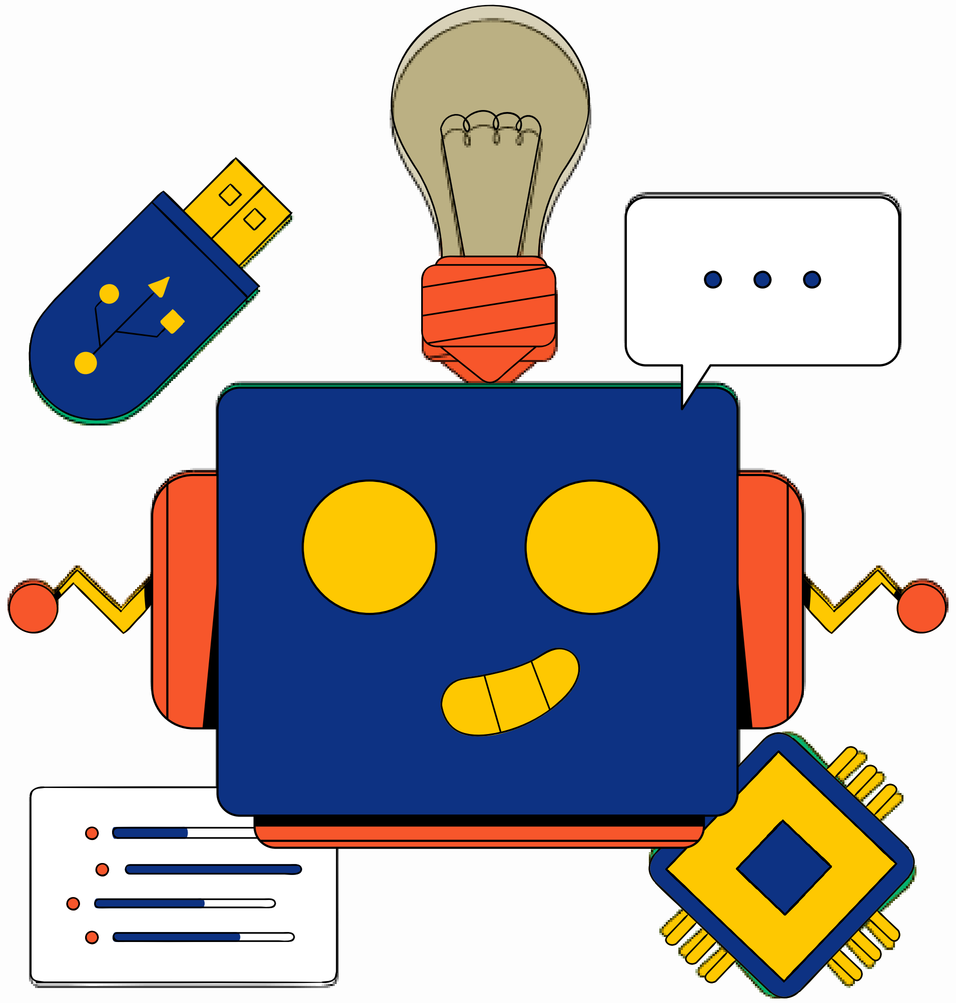
Bug Severity Prediction using Deep Learning

by Asad Muhummad, Asad Ullah, Ahmed Shoaib

Supervisor: Abdul Samad, Sandesh Kumar



Introduction

Bug severity prediction is vital for prioritizing and resolving software issues efficiently. Traditional manual methods are subjective, time-consuming, and error-prone. Leveraging transformer-based models like BERT, which excels in natural language processing tasks, enables automated and accurate classification of bug severity from textual reports. This approach improves decision-making, accelerates debugging, and enhances software quality.

Novelty

- Attention-based architectures like Transformers and attention-enhanced LSTMs have not been widely applied to bug severity prediction.
- Existing methods largely depend on traditional machine learning models such as Random Forests, Naive Bayes, and SVM, or deep learning models like CNNs and RNNs, often without attention mechanisms.
- Explore and leverage advanced transformer-based models (RoBERTa, ALBERT, DistilBERT, DeBERTa) for bug severity prediction.
- Evaluate and compare the performance and accuracies of these models to uncover insights and establish benchmarks for improving predictive capabilities.
- Bridge the gap in research by integrating state-of-the-art models to enhance automated bug severity classification in software development.

Research Question

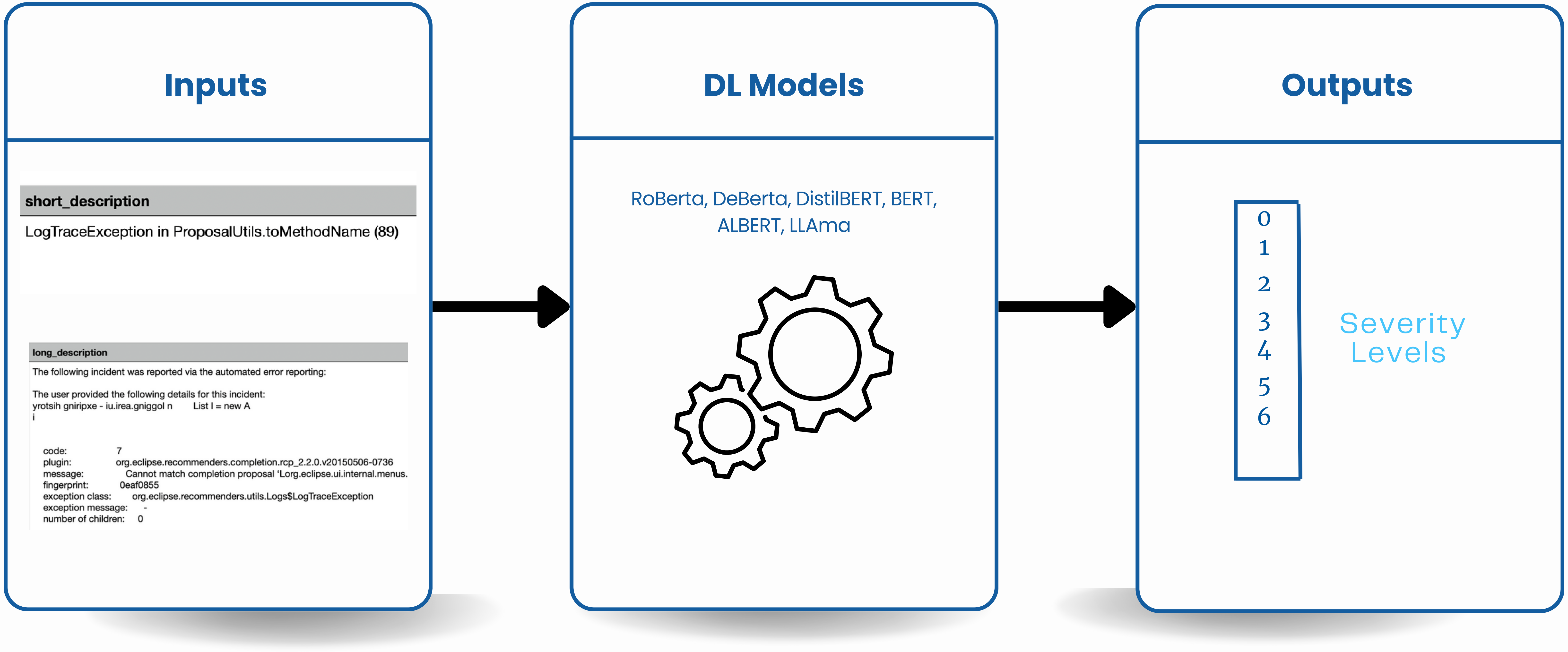
How do the accuracy and performance of deep learning models such as **BERT, ALBERT, DistilBERT, Deberta and RoBerta** compare in assigning severity levels to bugs based on their textual descriptions?

Problem Statement

Given the **short description and long description** of a bug report (where the descriptions were written by a human bug reporter) **predict its severity level** belonging to one of the defined bug severity level classes 0-6

Year	Model	Results	Dataset
2021 [1]	MASP- CNNs	0.7563 - Accuracy 0.7825 - Precision 0.8623 - Recall 0.8169 - F1	Mozilla and Eclipse Projects
2021 [5]	K-Nearest Neighbour (KNN)	0.707-Accuracy	FLOSS Dataset
2022 [2]	CNN-LSTM	F-Score Measure 0.9602 (Eclipse) 0.9322 (Mozilla dataset)	Mozilla and Eclipse Projects
2024 [3]	BERT-SBR	0.9113 - Accuracy , 0.9102 - Precision 0.9113 -Recall 0.9103 - F-Score Measure	HuggingFace, SenitWordNet

Overview



Inputs
short_description
LogTraceException in ProposalUtils.toMethodName (89)
long_description
The following incident was reported via the automated error reporting:
The user provided the following details for this incident:
yrotsih gniripxe - iu.irea.gniggol n List l = new A
i
code: 7
plugin: org.eclipse.recommenders.completion.rcp_2.2.0.v20150506-0736
message: Cannot match completion proposal 'Lorg.eclipse.ui.internal.menus.
fingerprint: 0eaf0855
exception class: org.eclipse.recommenders.utils.Logs$LogTraceException
exception message: -
number of children: 0
DL Models
RoBerta, DeBerta, DistilBERT, BERT, ALBERT, LLAma
Outputs
0
1
2
3
4
5
6
Severity Levels

RESULTS

Results

Model	Evaluation Metrics	Hyperparameters
BERT	**Validation Loss: 0.5622** **Validation Accuracy: 0.8468** **Validation F1 Score: 0.7878**	epochs = 10 batch_size = 32 learning_rate = 1e-6
Alberta	Train Loss: 0.2822710 Validation Loss: 0.7875 Validation Accuracy: 0.7993 Validation F1 Score: 0.7731	epochs = 10 batch_size = 16 learning_rate = le-6 [Note: !6 min per epoch duration]
Roberta	Validation Loss: 0.6632 Validation Accuracy: 0.8331 Validation F1 Score: 0.7573	epochs = 10 batch_size = 16 learning_rate = 1e-6 [Note: 16 min per epoch duration]
DistilBERT	Validation Loss: 0.5989 Validation Accuracy: 0.8420 Validation F1 Score: 0.7747	epochs = 10 batch_size = 16 learning_rate = 1e-6 [Note: 15 min epoch instead of 45min in BERT]
DeBerta	Validation Loss: 0.6305 Validation Accuracy: 84.35% Validation F1 Score: 0.7715	epochs = 10 batch_size = 16 learning_rate = 1e-6 [Note: 9 min avg. epoch]

Issues Experienced

Severe Class Imbalance
Impacting F1 scores

F1 Scores:
Class 2: 99%
Others: 20-30%

How do we solve this?

- **Weighted Random Sampling**
- **Weighted Loss Function**

Code	Count of severity_code	Percentage of Severity Code
0	1	0
1	1121	2
2	42942	84
4	3446	7
5	2801	5
6	1035	2

Results After Weights Supplied

Model	Evaluation Metrics
BERT	Validation Loss: 1.0915 Validation Accuracy: 0.6112 Validation F1 Score: 0.6736
DistilBERT	Validation Loss: 1.4608 Validation Accuracy: 0.2688 Validation F1 Score: 0.3311
Roberta	Validation Loss: 1.0693 Validation Accuracy: 0.6090 Validation F1 Score: 0.6706

All models ran on 10 epochs, 1e-5 Learning Rate and 16 Batch Size.

Discussion and Future Work

01 Unlike our direct tokenization approach, the referenced papers used topic-based feature selection to isolate key features and reduce noise.

02 Transformer models performed well, but CNN-LSTM architectures may better capture spatial and temporal patterns in bug descriptions.

03 Use data augmentation (e.g., SMOTE) or cost-sensitive methods to address class imbalance.

04 Incorporate bug metadata (e.g., timestamps, components) and use topic modeling (e.g., LDA) for richer context.

05 Use external Libraries such as SentiWordNet to incorporate further context in the form of emotion scores to pass as input to the model

References:

[1] A.-H. Dao and C.-Z. Yang, "Severity prediction for bug reports using multi-aspect features: A deep learning approach," Mathematics, vol. 9, no. 14, p. 1644, 2021. [Online]. Available: https://doi.org/10.3390/math9141644.

[2] Gomes, L. A. F., Torres, R. da S., & Côrtes, M. L. (2021). On the prediction of long-lived bugs: An analysis and comparative study using FLOSS projects. Information and Software Technology 132, 106508.

[3] J. Kim and G. Yang, "Bug Severity Prediction Algorithm Using Topic-Based Feature Selection and CNN-LSTM Algorithm," in IEEE Access, vol. 10, pp. 94643-94651, 2022, doi: 10.1109/ACCESS.2022.3204689.

[4] Ali, A., Xia, Y., Umer, Q., & Osman, M. (2024). BERT based severity prediction of bug reports for the maintenance of mobile applications. Journal of Systems and Software, 208, 111898. https://doi.org/10.1016/J.JSS.2023.111898

[5] Wang, R., Ji, X., Xu, S., Tian, Y., Jiang, S., & Huang, R. (2024). An empirical assessment of different word embedding and deep learning models for bug assignment. Journal of Systems and Software, 210, 111961. https://doi.org/10.1016/J.JSS.2024.111961

THANK YOU!