APPENDIX A
BASELINES

We used the following approaches as our baselines.

1) **LBT-P** is a bug triage framework using patient knowledge distillation (PKD) approach to compress RoBERTa-large that attempts to mitigate PLM's overthinking problem [1]. As LBT-P's source code was not publicly available, we contacted the authors, who informed us that it was proprietary and therefore could not be shared. Hence, we reproduced the framework by meticulously following the paper's methodologies. Initial discrepancies in results prompted us to communicate with the authors again, who then provided some partial code snippets and recommended a higher learning rate of $1e^{-4}$ for distillation. With these adjustments, we reproduced similar results to the original on the GC dataset, validating our implementation. However, reproducing results on MC and MF was challenging due to our use of *active developers* and significant data distribution shifts. To better evaluate model performance, we maximized developer overlap between training and test sets, reducing data sparsity for a more realistic assessment of generalization. This adjustment impacted all baselines and our approach consistently, ensuring fair comparisons.
2) **DBRNN-A** is a deep bidirectional RNN with Attention that uses LSTM units to capture context in bug reports, addressing the challenge of mixed text, code snippets, and stack traces. Our reproduction of DBRNN-A following this implementation yields similar results to the original paper.
3) **MDN** or Multiple Developer Network first attempts to find similar issues by applying smoothed Unigram Model (UM) and Kullback-Leibler (KL) divergence. Then it generates a network of developers by the number of commits and comments on those bug reports.
4) We also compare our approach with traditional TF-IDF-based SVM classifier and **Large PLMs with FCN and CNN classifiers**. We evaluated high-performing PLM variants for bug triaging, as reported in [2]. All PLMs were sourced from the HuggingFace repository [3] and served as the core text embedding modules. We used the large variants of BERT [4], RoBERTa [5], and DeBERTa [6], given their typical superiority over base versions, along with the base variant of CodeBERT [7], as it is the only publicly available option. For TriagerX CBR, we fine-tuned the base versions of RoBERTa and DeBERTa in an ensemble, showing that with careful tuning and model combination, base models can outperform larger ones in bug triaging.

APPENDIX B
PLM VARIANTS

Pretrained Language Models (PLMs) are available in different sizes, offering trade-offs between performance, efficiency, and deployment requirements. Larger variants capture richer representations but require more memory and computation, making model selection dependent on task-specific needs.

Table I provides an overview of the PLM variants used in this study, highlighting key architectural differences. All of the models were sourced and evaluated from the HuggingFace repository [3] to produce this table.

TABLE I: Overview of different PLMs.

PLM	#Params	Hidden Size	Layers	Attention Heads
BERT-Base	110M	768	12	12
BERT-Large	335M	1024	24	16
RoBERTa-Base	125M	768	12	12
RoBERTa-Large	355M	1024	24	16
DeBERTa-Base	139M	768	12	12
DeBERTa-Large	405M	1024	24	16
CodeBERT	125M	768	12	12

APPENDIX C
BENCHMARK DATASETS

We utilized large-scale Google Chromium (GC), Mozilla Core (MC), and Mozilla Firefox (MF) datasets from the literature [8] and newly prepared benchmark datasets for our analysis as the literature datasets we are currently using do not contain developer interaction information. To create our own benchmarks, we leveraged the GitHub API to collect data from the OpenJ9 and TypeScript (TS) bug repositories. We gathered all reported bugs up to August 2024, dating back to the creation of each repository. These datasets include information such as issue titles, descriptions, assigned developers, and contributors (e.g., those who commented, committed, or created pull requests). Directly assigned developer to an issue is considered the owner of a bug report. In cases where there is no direct assignment on the GitHub issue page, we considered the last person to make a commit or pull request to that bug as the owner. If neither of this information was found on the issue page, we discarded the issue from our dataset.

APPENDIX D
EXPERIMENTAL RESULTS

This section presents additional results comparing TriagerX and its components against all evaluated baselines across all datasets used in this study.

A. Comparison of TriagerX full framework with all baselines

TABLE II: Top-k accuracy of TriagerX framework compared to all considered baselines.

Dataset	Method	K=1	K=3	K=5	K=10	K=20
Openj9	**TriagerX (CBR+IBR)**	**0.327**	**0.533**	**0.633**	**0.807**	**0.918**
	TriagerX CBR	0.272	0.476	0.601	0.780	0.901
	TriagerX IBR	0.284	0.488	0.585	0.699	0.860
	DeBERTa-Large (FCN)	0.178	0.418	0.547	0.698	0.897
	RoBERTa-Large (FCN)	0.191	0.418	0.586	0.743	0.890
	BERT-Large (FCN)	0.168	0.393	0.507	0.694	0.857
	CodeBERT (FCN)	0.129	0.331	0.476	0.689	0.849
	DeBERTa-Large (CNN)	0.170	0.374	0.503	0.675	0.853
	RoBERTa-Large (CNN)	0.206	0.403	0.531	0.670	0.822
	BERT-Large (CNN)	0.181	0.323	0.445	0.652	0.839
	CodeBERT (CNN)	0.100	0.253	0.409	0.595	0.805
	LBT-P	0.211	0.407	0.501	0.631	0.797
	DBRNN-A	0.127	0.300	0.454	0.627	0.775
	MDN	0.100	0.349	0.422	0.606	0.746
	TF-IDF + SVM	0.189	0.357	0.484	0.665	0.828
TS	**TriagerX (CBR+IBR)**	**0.353**	**0.615**	**0.711**	**0.830**	**0.930**
	TriagerX CBR	0.324	0.582	0.682	0.812	0.920
	TriagerX IBR	0.278	0.487	0.564	0.650	0.720
	DeBERTa-Large (FCN)	0.264	0.509	0.618	0.794	0.924
	RoBERTa-Large (FCN)	0.319	0.552	0.669	0.824	0.929
	BERT-Large (FCN)	0.253	0.481	0.614	0.784	0.906
	CodeBERT (FCN)	0.120	0.309	0.458	0.733	0.915
	DeBERTa-Large (CNN)	0.212	0.428	0.580	0.765	0.918
	RoBERTa-Large (CNN)	0.294	0.495	0.602	0.739	0.876
	BERT-Large (CNN)	0.151	0.345	0.502	0.705	0.893
	CodeBERT (CNN)	0.143	0.352	0.506	0.704	0.890
	LBT-P	0.279	0.503	0.627	0.781	0.908
	DBRNN-A	0.231	0.447	0.579	0.729	0.838
	MDN	0.075	0.100	0.275	0.475	0.525
	TF-IDF + SVM	0.272	0.428	0.493	0.663	0.830

B. Comparison of TriagerX CBR with all baselines

TABLE III: Top-k accuracy of all considered baselines on different datasets compared to TriagerX CBR.

Dataset	Method	K=1	K=3	K=5	K=10	K=20
Google Chromium	**TriagerX CBR**	**0.345**	**0.537**	**0.612**	**0.710**	**0.803**
	DeBERTa-Large (FCN)	0.285	0.474	0.567	0.677	0.767
	RoBERTa-Large (FCN)	0.267	0.461	0.551	0.660	0.755
	BERT-Large (FCN)	0.255	0.433	0.520	0.630	0.715
	CodeBERT (FCN)	0.224	0.403	0.493	0.606	0.697
	DeBERTa-Large (CNN)	0.251	0.432	0.525	0.639	0.738
	RoBERTa-Large (CNN)	0.281	0.475	0.564	0.671	0.763
	BERT-Large (CNN)	0.271	0.455	0.549	0.655	0.743
	CodeBERT (CNN)	0.159	0.319	0.399	0.519	0.634
	LBT-P	0.318	0.499	0.578	0.676	0.763
	DBRNN-A	0.183	0.318	0.385	0.482	0.581
	TF-IDF + SVM	0.204	0.310	0.376	0.454	0.529
Mozilla Core	**TriagerX CBR**	**0.340**	**0.521**	**0.598**	**0.700**	**0.805**
	DeBERTa-Large (FCN)	0.257	0.437	0.521	0.639	0.744
	RoBERTa-Large (FCN)	0.276	0.458	0.540	0.650	0.749
	BERT-Large (FCN)	0.215	0.378	0.461	0.571	0.681
	CodeBERT (FCN)	0.206	0.371	0.455	0.570	0.678
	DeBERTa-Large (CNN)	0.269	0.445	0.533	0.639	0.737
	RoBERTa-Large (CNN)	0.306	0.490	0.568	0.668	0.758
	BERT-Large (CNN)	0.258	0.432	0.514	0.624	0.725
	CodeBERT (CNN)	0.268	0.447	0.532	0.640	0.739
	LBT-P	0.279	0.471	0.553	0.655	0.748
	DBRNN-A	0.164	0.290	0.367	0.481	0.594
	TF-IDF + SVM	0.238	0.386	0.454	0.546	0.638
Mozilla Firefox	**TriagerX CBR**	**0.272**	**0.471**	**0.576**	**0.718**	**0.835**
	DeBERTa-Large (FCN)	0.221	0.402	0.488	0.646	0.801
	RoBERTa-Large (FCN)	0.218	0.400	0.505	0.642	0.781
	BERT-Large (FCN)	0.213	0.353	0.445	0.585	0.748
	CodeBERT (FCN)	0.193	0.366	0.454	0.597	0.760
	DeBERTa-Large (CNN)	0.199	0.368	0.473	0.627	0.798
	RoBERTa-Large (CNN)	0.248	0.441	0.534	0.671	0.801
	BERT-Large (CNN)	0.148	0.306	0.389	0.516	0.670
	CodeBERT (CNN)	0.219	0.385	0.483	0.619	0.754
	LBT-P	0.243	0.423	0.524	0.646	0.788
	DBRNN-A	0.135	0.253	0.334	0.441	0.612
	TF-IDF + SVM	0.221	0.388	0.454	0.540	0.623

References

[1] Y. Kaya, S. Hong, and T. Dumitras, "Shallow-deep networks: Understanding and mitigating network overthinking," in *International conference on machine learning*. PMLR, 2019, pp. 3301–3310.

[2] A. K. Dipongkor and K. Moran, "A comparative study of transformer-based neural text representation techniques on bug triaging," in *2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2023, pp. 1012–1023.

[3] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew, "Huggingface's transformers: State-of-the-art natural language processing," *CoRR*, vol. abs/1910.03771, 2019.

[4] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018.

[5] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized BERT pretraining approach," *CoRR*, vol. abs/1907.11692, 2019.

[6] P. He, X. Liu, J. Gao, and W. Chen, "Deberta: Decoding-enhanced BERT with disentangled attention," *CoRR*, vol. abs/2006.03654, 2020.

[7] Z. Feng, D. Guo, D. Tang, N. Duan, X. Feng, M. Gong, L. Shou, B. Qin, T. Liu, D. Jiang, and M. Zhou, "CodeBERT: A pre-trained model for programming and natural languages," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, T. Cohn, Y. He, and Y. Liu, Eds. Online: Association for Computational Linguistics, Nov. 2020, pp. 1536–1547.

[8] S. Mani, A. Sankaran, and R. Aralikatte, "Deeptriage: Exploring the effectiveness of deep learning for bug triaging," *CoRR*, vol. abs/1801.01275, 2018.