Few-Shot Learning for Issue Report Classification

Giuseppe Colavito
University of Bari, Italy
giuseppe.colavito@uniba.it

Filippo Lanubile
University of Bari, Italy
filippo.lanubile@uniba.it

Nicole Novielli
University of Bari, Italy
nicole.novielli@uniba.it

2023 IEEE/ACM 2nd International Workshop on Natural Language-Based Software Engineering (NLBSE) | 979-8-3503-0178-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/NLBSE59153.2023.00011

***Abstract*—We describe our participation in the tool competition in the scope of the 2nd International Workshop on Natural Language-based Software Engineering. We propose a supervised approach relying on SETFIT, a framework for few-shot learning and sentence-BERT (SBERT), a variant of BERT for effective sentence embedding. We experimented with different settings, achieving the best performance by training and testing the SETFIT-based model on a subset of data with manually verified labels (F1-micro = .8321). For the sake of the challenge, we evaluate the SETFIT model on the challenge test set, achieving F1-micro = .7767.**

I. Introduction

Software development is a complex and dynamic process that requires efficient management of issues and bugs. Automatic classification of issue reports plays a key role in achieving effective decision-making and prioritization. Following the experience of the first edition [1], the tool competition at NLBSE'23 invites contributions describing systems for automatic issue report classification. To train and evaluate the systems, the organizers provide participants with a dataset including more than 1.4M GitHub issue reports labeled as either *bug*, *features*, *question*, and *documentation*.

Early studies on automatic issue classification leveraged traditional machine learning, achieving accuracy between 77% and 82% [2]. More recently, researchers started to use deep learning. Kallis et al. [3], [4] proposed Ticket Tagger, a classifier based on *fastText* [5], that leverages the textual content of an issue title and body. During the first edition of the tool competition at NLBSE'22 pre-trained language models, such as BERT [6], emerged as state-of-the-art [7]–[11].

In our contribution to the first edition of the challenge edition, experimented with fine-tuning of BERT [6], a task-agnostic pre-trained language model released by Google, and its variants ALBERT [12] and RoBERTa [13]. All models outperformed the baseline and we observed the best performance with the RoBERTa-based model (F1 = .8591) [7]. The error analysis suggests that one of the main causes of issue misclassification is the presence of inconsistencies in the labeling rationale. Indeed, data quality recently emerged as a crucial issue for the performance of supervised classifiers [14]. Given that the source of the data for the NLBSE'23 [15] dataset is the same as for the NLBSE'22 [1] dataset, it is reasonable to assume that this concern also holds for the projects in the current dataset. As such, in this paper, we investigate the potential impact of data quality improvement through manual verification of labels. Thus, we formulate the following research question:

To what extent does the label consistency impact the performance of supervised issue classification models?

To address our research question, we first improve the label correctness of a subset of the train and test data. Then, we experiment with SETFIT an effective methodology for fine-tuning of transformer-based models using few-shot learning [16]. Our model based on SETFIT achieves F1 = .7767 on the full test set and F1 = .8321 on the relabeled one. The replication material is available on GitHub [17].

II. Background and Related Work

Word embeddings, such as Word2Vec [18], GloVe [19] and FastText [20] are some of the early approaches to representation learning in natural language processing (NLP). These models learn a vector representation for each word in a large training corpus, thus capturing words' contextual semantics. Recent developments in NLP introduced transformer-based models such as BERT [6], which is pre-trained on a large text corpus using masked language modeling and next-sentence prediction objectives. This enables BERT to effectively produce a sentence-level representation that can be fine-tuned for a wide range of NLP tasks. BERT and its variants (such as RoBERTa [13]), achieve state-of-the-art results in several NLP benchmarks. Recently, other transformer-based models were introduced to improve sentence representation by minimizing the distance between semantically similar sentences. Sentence-BERT (SBERT) is designed to address BERT's inefficiency in semantic similarity search and clustering [16], [21]. SBERT employs a siamese and triplet network architecture to create semantically rich sentence representations that can be compared using cosine-similarity.

Few-shot learning addresses the limitations posed by fine-tuning transformer-based language models using small training sets, thus reducing the need for manual labeling [16]. Among others, SETFIT (Sentence Transformer Finetuning) is a framework for few-shot fine-tuning of Sentence Transformers [16].

III. Methodology

The challenge dataset [15] consists of 1.4M issue reports extracted from GH Archive[1] using Google BigQuery[2]. All issues included were closed in 2022 (January - September) and have a non-empty body in English. Each report includes its *id*, *title*, *body*, *author association*, and *label*. The labels are indicative of the type of issue, with the possible classes *bug*, *features*, *question*, and *documentation*. Issues that are labeled

[1]https://www.gharchive.org/
[2]https://cloud.google.com/bigquery/

979-8-3503-0178-6/23/$31.00 ©2023 IEEE
DOI 10.1109/NLBSE59153.2023.00011

with synonyms [22], are mapped to one of these four labels. Issues with multiple labels have been excluded. The dataset is split into a train (90%) and a test set (10%) through stratified sampling (see Table I).

TABLE I
DISTRIBUTION OF ISSUES IN THE TRAINING AND TEST SETS

	Train set		Test set	
Bug	670,951	(53%)	74,781	(53%)
Documentation	56,666	(4%)	6,252	(4%)
Feature	472,216	(37%)	52,797	(37%)
Question	76,048	(6%)	8,490	(6%)
Total	1,275,881		142,320	

The error analysis we conducted in the scope of our previous work presented at NLBSE'22 [7] reveals that one of the main causes for misclassification was the presence of noisy labels. One of the reasons for such noise was the variability in labeling rationale among different projects. Another source of noise is the inherent difficulty in distinguishing between *bugs* and *questions*, as both often contain error traces and an explanation of how to reproduce the error. The distinction between bugs and questions mostly depends on the behavior of the code, i.e. issues reporting errors caused by misuse of the library are typically labeled as questions, which distinguish them from bug reports. However, this distinction is hard to capture based on the textual content only. Often, bugs and questions contain suggestions on how to fix the problem, which might lead to inconsistent labeling of bugs as *feature* requests or vice versa.

The use of noisy data for training is a significant concern, as it may result in a lack of reliability and effectiveness in the model, thus leading to inaccurate results [14], [23], [24]. To mitigate the effects of label noise, further manual analysis and quality checks are necessary to ensure the accuracy and reliability of the labels in the dataset [14]. To this aim, we decide to perform a manual check of the label quality. Given the large number of issues in the NLBSE'23 dataset, this may be a resource-intensive task. Nevertheless, it is important to consider the potential benefits of a more accurate and reliable dataset, such as improved model performance and reduced risk of false predictions [14].

To balance the costs and benefits, we decided to adopt a more focused approach. We randomly sampled 400 instances from the dataset, 200 from the training set and 200 from the test set. Both sets are balanced, having each 50 examples per class. This smaller, hand-labeled dataset serves as a gold standard for training and evaluating a few-shot learner. The distribution of the dataset before and after the manual labeling is depicted in Table II. By leveraging transfer learning, our goal is to understand if a few-shot learner is able to effectively generalize the hand-labeled examples to the entire dataset. The annotation procedure for the dataset involved the following steps. First, three annotators independently labeled each issue based on the textual information only (title and body of the issue). Each issue report was assigned to two of the annotators. We observed a Cohen κ = 0.739, wich corresponds to a substantial interrater agreement [25]. Then, the three annotators discussed and resolved the cases of disagreement during a joint plenary meeting. During this step, we discarded 29 (7.25% of the labeling set) cases for which it was not possible to reach a consensus, i.e. it was not possible to understand the intention of the issue author could not be interpreted due to insufficient/non-informative text, empty issue templates, etc. This procedure ensured the reliability and consistency of the annotations. The manually annotated sample is publicily available [26].

TABLE II
DISTRIBUTION OF LABELS IN THE EXTRACTED SAMPLES

	GitHub labeling				Hand labeling			
Label	Train set		Test set		Train set		Test set	
Bug	50	25%	50	25%	47	24%	53	27%
Documentation	50	25%	50	25%	33	17%	32	16%
Feature	50	25%	50	25%	60	30%	55	28%
Question	50	25%	50	25%	44	22%	47	24%
Discarded	–	–	–	–	16	8%	13	7%
Total	200		200		200		200	

We use this set to train the model submitted for the challenge. As a baseline, the organizers provided the script for training a RoBERTa-based model. To enable a fair comparison, both the models, RoBERTa and SETFIT, are trained using the same preprocessing. As a first pre-processing step, non-textual items, such as images, links, and code snippets, were identified and replaced with tokens (e.g. <img>for images) in the dataset. Subsequently, a text normalization step was performed using the ekphrasis Text Pre-Processor [3] which effectively identified and replaced items such as URLs, email addresses, symbols, phone numbers, mentions, time, date, and numbers with specific tokens.

For the RoBERTa model, we use the RoBERTa tokenizer and then perform padding or truncating to represent the documents with 512 tokens. For the SETFIT model, we used *all-mpnet-base-v2*, one of the top performing pre-trained model for embeddings, according to the Sentence Transformers documentation[4]. This model has an input size of 384 tokens, which enables the full representation of more than 90% of the issues [15].

IV. RESULTS AND DISCUSSION

In Table III, we report the results obtained by training the SETFIT classifier on the hand-labeled gold standard and evaluated on both the hand-labeled test set (a), and on the full test set distributed for the challenge (c). For a fair comparison, we compare the SETFIT model with the performance obtained by RoBERTa on the same test set, when trained on the hand-labeled gold standard set (b1) and on the full train set distributed by the organizers (b2). The challenge baseline is reported in row (d) of the table. Finally, the perfomance of SETFIT model submitted to the challenge is reported in Italic in the table (model c). The SETFIT model is designed to

[3]https://github.com/cbaziotis/ekphrasis

[4]https://www.sbert.net/docs/pretrained_models.html

TABLE III
PERFORMANCE OF THE SETFIT MODEL AND COMPARISON WITH THE ROBERTA BASELINE APPROACH. THE PERFORMANCE OF THE MODEL SUBMITTED TO THE CHALLENGE IS REPORTED IN *Italic*. IN BOLD, WE HIGHLIGHT THE BEST PERFORMANCE OBTAINED WITH SETFIT.

	Model	Train		Test		F1-micro
(a)	SETFIT	Sampled	Manual labels	Sampled	Manual labels	**0.8321**
(b1)	RoBERTa	Sampled	Manual labels	Sampled	Manual labels	0.4348
(b2)	RoBERTa	Full	GitHub labels	Sampled	Manual labels	0.8182
(c)	*SETFIT*	*Sampled*	*Manual labels*	*Full*	*GitHub labeling*	*0.7767*
(d)	RoBERTa (baseline)	Full	GitHub labels	Full	GitHub labels	0.8890

TABLE IV
PERFORMANCE BY CLASS FOR THE ROBERTA BASELINE ON THE FULL CHALLENGE DATASET AND THE SETFIT MODELS TRAINED ON THE MANUALLY LABELED GOLD STANDARD.

	RoBERTa baseline Train and test: full dataset				SETFIT Train: manual gold standard Test: full challenge test set				SETFIT Train and test on the manual gold standard			
Label	Precision	Recall	F1	% Supp	Precision	Recall	F1	% Supp	Precision	Recall	F1	% Supp
Bug	0.9120	0.9360	0.9240	53	0.9151	0.7894	0.8476	53	0.8723	0.8472	0.8590	28
Documentation	0.7480	0.6920	0.7190	4	0.4422	0.5715	0.4986	4	0.9039	0.6594	0.7616	17
Feature	0.8960	0.8910	0.8930	37	0.8049	0.8253	0.8149	37	0.7494	0.9182	0.8251	30
Question	0.7010	0.6050	0.6490	6	0.3518	0.6458	0.4554	6	0.8754	0.8319	0.8528	25
Micro average	0.8890	0.8890	0.8890		0.7846	0.7846	0.7846		0.8321	0.8321	0.8321	
Macro average	0.8140	0.7810	0.7960		0.6285	0.7080	0.6542		0.8502	0.8142	0.8246	

TABLE V
AVERAGE TRAIN AND INFERENCE TIME COMPARISON OVER TEN RUNS

Task	RoBERTa	SETFIT	GPU
Train (HH:MM:SS)	12:24:38	00:03:11	Nvidia A100 40GB
Inference (s)	0.04	0.02	

learn from a few examples, while the RoBERTa baseline is trained on the full set. Other than comparing the performance, in Table V we report the overall training time and the inference time for the single issue report during the classification.

The SETFIT model trained on the manually labeled gold standard achieves a F1-micro = .7767 (see model (c) in Table III). The RoBERTa model still represents the state-of-the-art (F1-micro = .8890) when large training sets are available (see baseline model (d) in Table III). However, when a reduced training set is available, the SETFIT model (model (a), F1-micro = .8321) largely outperforms RoBERTA (model, (b1) with F1-micro = .4348). It is important to note that in, both, settings (a) and (b1) the label consistency between the train and test set is ensured by the manual labeling.

In Table IV, we report the class-based performance, as well as the F1 macro-average. In fact, the ability of a classifier to behave well also on categories with few positive training instances is emphasized by macro-averaging and much less so by micro-averaging [27]. Please, note that the performance of the SETFIT model trained and tested on the manual gold standard is not directly comparable with the first two models in Table IV, as they are evaluated on the full test set. Nevertheless, these results provide a better insight into the models' behavior. The SETFIT model achieves a better F1-macro when trained on the manually annotated gold standard, for which we have a more balanced class distribution. As a consequence, this model also achieves the best performance for *question* and *documentation* classes, which are underrepresented in the full dataset.

This evidence also triggers a question regarding the best practice in absence of training data. The setting (b2) in Table III operationalizes such a scenario, which may occur in real-world applications, i.e. a test set from a given project is available for assessing the performance of 'off-the-shelf' state-of-the-art tools. In this setting, we simulate this situation by training the RoBERTa model on the entire training set with the GitHub labels and testing it on the manually annotated test set. The performance is lower (F1-micro = .8182) compared to the SETFIT model trained on the manually annotated train set (F1-micro = .8321). For this reason, models trained on crowd-sourced data may perform well on the test set when it is drawn from the same source and thus shares the same label rationale of the train set (model (d) in Table III). Conversely, a drop in performance can be observed when the same model is tested on a subset including manually validated labels. This highlights the importance of considering the quality of the training data when evaluating models. Overall, our results suggest that *with small but high-quality datasets a good classification performance can be achieved using the few-shot learning approach* implemented by SETFIT.

V. CONCLUSION

We investigated the impact of data quality improvement through manual verification of labels on issue classification performance. We trained and evaluated a model based on SETFIT using a subset of manually verified data. While not outperforming the RoBERTa baseline on the full test set, the model achieves better performance when trained and tested on data for which label consistency was manually verified. This result supports the idea that a smaller, hand-labeled dataset can be more effective than a larger, noisy dataset, provided that approaches optimized for supervised learning with small training data are used. The proposed approach can be used to effectively fine-tune issue classifiers for small projects with a limited amount of training data. In future work, we plan to experiment with other techniques based on few-shot

learning also leveraging domain-specific sentence embeddings that might better capture the contextual semantics of words.

Acknowledgments

The computational work has been executed on the IT resources made available by two projects, ReCaS and PRISMA, funded by MIUR under the program BPON R&C 2007-2013.

References

[1] R. Kallis, O. Chaparro, A. Di Sorbo, and S. Panichella, "Nlbse'22 tool competition," in *Proceedings of The 1st Intl. Workshop on Natural Language-based Software Engineering (NLBSE'22)*, 2022.

[2] G. Antoniol, K. Ayari, M. Di Penta, F. Khomh, and Y.-G. Guéhéneuc, "Is it a bug or an enhancement? a text-based approach to classify change requests," in *Proceedings of the 2008 Conf. of the center for advanced studies on collaborative research: meeting of minds*, New York, NY, USA: ACM, 2008. DOI: 10.1145/1463788.1463819.

[3] R. Kallis, A. Di Sorbo, G. Canfora, and S. Panichella, "Ticket tagger: Machine learning driven issue classification," in *2019 IEEE Intl. Conf. on Software Maintenance and Evolution, ICSME 2019, Cleveland, OH, USA, September 29 - October 4, 2019*, IEEE, 2019. DOI: 10.1109/ICSME.2019.00070.

[4] R. Kallis, A. Di Sorbo, G. Canfora, and S. Panichella, "Predicting issue types on github," *Science of Computer Programming*, 2021. DOI: https://doi.org/10.1016/j.scico.2020.102598.

[5] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, *Bag of Tricks for Efficient Text Classification*, 2016. DOI: 10.48550/arXiv.1607.01759.

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

[7] G. Colavito, F. Lanubile, and N. Novielli, "Issue Report Classification Using Pre-trained Language Models," in *(NLBSE 2022)*, 2022. DOI: 10.1145/3528588.3528659.

[8] A. Trautsch and S. Herbold, *Predicting Issue Types with seBERT*, 2022. DOI: 10.48550/arXiv.2205.01335.

[9] M. L. Siddiq and J. C. S. Santos, "BERT-Based GitHub Issue Report Classification," in *(NLBSE 2022)*, 2022. DOI: 10.1145/3528588.3528660.

[10] S. Bharadwaj and T. Kadam, "GitHub Issue Classification Using BERT-Style Models," in *(NLBSE 2022)*, 2022. DOI: 10.1145/3528588.3528663.

[11] M. Izadi, "CatIss: An Intelligent Tool for Categorizing Issues Reports using Transformers," in *(NLBSE 2022)*, 2022. DOI: 10.1145/3528588.3528662.

[12] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations*, 2020.

[13] Y. Liu, M. Ott, N. Goyal, *et al.*, *RoBERTa: A Robustly Optimized BERT Pretraining Approach*, 2019. DOI: 10.48550/arXiv.1907.11692.

[14] X. Wu, W. Zheng, X. Xia, and D. Lo, "Data Quality Matters: A Case Study on Data Label Correctness for Security Bug Report Prediction," *IIEEE Trans. Software Eng.*, no. 7, 2022. DOI: 10.1109/TSE.2021.3063727.

[15] R. Kallis, M. Izadi, L. Pascarella, O. Chaparro, and P. Rani, "The nlbse'23 tool competition," in *Proceedings of The 2nd Intl. Workshop on Natural Language-based Software Engineering (NLBSE'23)*, 2023.

[16] L. Tunstall, N. Reimers, U. E. S. Jo, *et al.*, *Efficient Few-Shot Learning Without Prompts*, 2022. DOI: 10.48550/arXiv.2209.11055.

[17] G. Colavito, F. Lanubile, and N. Novielli, *Few-Shot Learning for Issue Report Classification*, version 1.0.0, 2023. [Online]. Available: https://github.com/collab-uniba/Issue-Report-Classification-NLBSE2023.

[18] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient Estimation of Word Representations in Vector Space*, 2013. DOI: 10.48550/arXiv.1301.3781.

[19] J. Pennington, R. Socher, and C. Manning, "GloVe: Global Vectors for Word Representation," in *Proceedings of the 2014 Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar: ACL, 2014. DOI: 10.3115/v1/D14-1162.

[20] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, *Enriching Word Vectors with Subword Information*, 2017. DOI: 10.48550/arXiv.1607.04606.

[21] N. Reimers and I. Gurevych, *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*, 2019. DOI: 10.48550/arXiv.1908.10084.

[22] M. Izadi, K. Akbari, and A. Heydarnoori, "Predicting the objective and priority of issue reports in software repositories," *Empir Software Eng*, no. 2, 2022. DOI: 10.1007/s10664-021-10085-3.

[23] X. Zhu and X. Wu, "Class Noise vs. Attribute Noise: A Quantitative Study," *Artificial Intelligence Review*, no. 3, 2004. DOI: 10.1007/s10462-004-0751-8.

[24] B. Sluban, D. Gamberger, and N. Lavra, "Advances in Class Noise Detection," *ECAI 2010*, 2010. DOI: 10.3233/978-1-60750-606-5-1105.

[25] A. J. Viera and J. M. Garrett, "Understanding interobserver agreement: The kappa statistic," *Family medicine*, 2005.

[26] G. Colavito, F. Lanubile, and N. Novielli, *Few-shot learning for issue report classification*, Zenodo, 2023. DOI: 10.5281/zenodo.7628150. [Online]. Available: https://doi.org/10.5281/zenodo.7628150.

[27] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys*, 2002. DOI: 10.1145/505282.505283.